Package 'compositions'

January 31, 2024

Version 2.0-8

Date 2024-01-25

Title Compositional Data Analysis

Author K. Gerald van den Boogaart <boogaart@hzdr.de>,
Raimon Tolosana-Delgado, Matevz Bren

Maintainer K. Gerald van den Boogaart <support@boogaart.de>

Depends R (>= 3.6)

Imports methods, utils, grDevices, stats, tensorA, robustbase, bayesm, graphics, MASS

Suggests rgl (>= 1.0.1), combinat, energy, knitr, rmarkdown

Description Provides functions for the consistent analysis of compositional data (e.g. portions of substances) and positive numbers (e.g. concentrations) in the way proposed by J. Aitchison and V. Pawlowsky-Glahn.

License GPL (>= 2)

URL http://www.stat.boogaart.de/compositions/

VignetteBuilder knitr

RoxygenNote 7.1.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2024-01-31 15:30:14 UTC

R topics documented:

`compositions-package` *Compositional Data Analysis*

Description

"compositions" is a package for the analysis of compositional and multivariate positive data (generally called "amounts"), based on several alternative approaches.

Details

The DESCRIPTION file:

Package:	compositions
Version:	2.0-8
Date:	2024-01-25

Title:	Compositional Data Analysis
Author:	K. Gerald van den Boogaart <boogaart@hzdr.de>, Raimon Tolosana-Delgado, Matevz Bren
Maintainer:	K. Gerald van den Boogaart <support@boogaart.de>
Depends:	R (>= 3.6)
Imports:	methods, utils, grDevices, stats, tensorA, robustbase, bayesm, graphics, MASS
Suggests:	rgl (>= 1.0.1), combinat, energy, knitr, rmarkdown
Description:	Provides functions for the consistent analysis of compositional data (e.g. portions of substances) and positi
License:	GPL (>= 2)
URL:	http://www.stat.boogaart.de/compositions/
VignetteBuilder:	knitr
RoxygenNote:	7.1.1

Index of help topics:

	from East Bay
ClamWest	Color-size compositions of 20 clam colonies from West Bay
clo	Closure of a composition
clr	Centered log ratio transform
clr2ilr	Convert between clr and ilr, and between cpt and ipt.
ClusterFinder1	Heuristics to find subpopulations of outliers
CoDaDendrogram	Dendrogram representation of acomp or rcomp objects
coloredBiplot	A biplot providing somewhat easier access to details of the plot.
colorsForOutliers1	Create a color/char palette or for groups of outliers
CompLinModCoReg	Compositional Linear Model of Coregionalisation
compOKriging	Compositional Ordinary Kriging
compositions-package	library(compositions)
ConfRadius	Helper to compute confidence ellipsoids
cor.acomp	Correlations of amounts and compositions
Coxite	Compositions, depths and porosities of 25 specimens of coxite
cpt	Centered planar transform
DiagnosticProb	Diagnostic probabilities
dist	Distances in variouse approaches
ellipses	Draw ellipses
endmemberCoordinates	Recast amounts as mixtures of end-members
Firework	Firework mixtures
geometricmean	The geometric mean
getDetectionlimit	Gets the detection limit stored in the data set
Glacial	Compositions and total pebble counts of 92 glacial tills
groupparts	Group amounts of parts
Hongite	Compositions of 25 specimens of hongite
HouseholdExp	Household Expenditures
Hydrochem	Hydrochemical composition data set of Llobregat river basin water (NE Spain)
idt	Isometric default transform
iit	Isometric identity transform
ilr	Isometric log ratio transform
ilrBase	The canonical basis in the clr plane used for ilr and ipt transforms.
ilt	Isometric log transform
ipt	Isometric planar transform
is.acomp	Check for compositional data type
IsMahalanobisOutlier	Checking for outliers
isoPortionLines	Isoportion- and Isoproportion-lines
juraset	The jura dataset
kingTetrahedron	Ploting composition into rotable tetrahedron

Kongite	Compositions of 25 specimens of kongite
lines.rmult	Draws connected lines from point to point.
logratioVariogram	Empirical variograms for compositions
MahalanobisDist	Compute Mahalanobis distances based von robust Estimations
mean.acomp	Mean amounts and mean compositions
meanRow	The arithmetic mean of rows or columns
Metabolites	Steroid metabolite patterns in adults and children
missingProjector	Returns a projector the the observed space in case of missings.
missingsInCompositions	The policy of treatment of missing values in the "compositions" package
missingSummary	Classify and summarize missing values in a dataset
mix.2aplus	Transformations from 'mixtures' to 'compositions' classes
mix.Read	Reads a data file in a mixR format
mvar	Metric summary statistics of real, amount or compositional data
names.acomp	The names of the parts
normalize	Normalize vectors to norm 1
norm.default	Vector space norm
oneOrDataset	Treating single compositions as one-row datasets
OutlierClassifier1	Detect and classify compositional outliers.
outlierplot	Plot various graphics to analyse outliers.
outliersInCompositions	Analysing outliers in compositions.
pairwisePlot	Creates a paneled plot like pairs for two different datasets.
parametricPosdefMat	Unique parametrisations for matrices.
perturbe	Perturbation of compositions
plot3D	plot in 3D based on rgl
plot3D.acomp	3D-plot of compositional data
plot3D.aplus	3D-plot of positive data
plot3D.rmult	plot in 3D based on rgl
plot3D.rplus	plot in 3D based on rgl
plot.acomp	Ternary diagrams
plot.aplus	Displaying amounts in scatterplots
plot.logratioVariogram	Empirical variograms for compositions
plot.missingSummary	Plot a Missing Summary
pMaxMahalanobis	Compute distributions of empirical Mahalanobis distances based on simulations
PogoJump	Honk Kong Pogo-Jumps Championship
power.acomp	Power transform in the simplex

powerofpsdmatrix	power transform of a matrix
princomp.acomp	Principal component analysis for Aitchison compositions
princomp.aplus	Principal component analysis for amounts in log geometry
princomp.rcomp	Principal component analysis for real compositions
princomp.rmult	Principal component analysis for real data
princomp.rplus	Principal component analysis for real amounts
print.acomp	Printing compositional data.
qHotellingsTsq	Hotellings T square distribution
qqnorm.acomp	Normal quantile plots for compositions and amounts
R2	R square
rAitchison	Aitchison Distribution
+.rcomp	Arithmetic operations for compositions in a real geometry
rcomp	Compositions as elements of the simplex embedded in the D-dimensional real space
rcompmargin	Marginal compositions in real geometry
rDirichlet	Dirichlet distribution
read.geoeas	Reads a data file in a geoeas format
relativeLoadings	Loadings of relations of two amounts
replot	Modify parameters of compositional plots.
rlnorm.rplus	The multivariate lognormal distribution
+.rmult	vectorial arithmetic for datasets in a classical vector scale
rmult	Simple treatment of real vectors
rnorm.acomp	Normal distributions on special spaces
robustnessInCompositions	Handling robustness issues and outliers in compositions.
+.rplus	vectorial arithmetic for data sets with rplus class
rplus	Amounts i.e. positive numbers analysed as objects of the real vector space
runif.acomp	The uniform distribution on the simplex
scalar	Parallel scalar products
scale	Normalizing datasets by centering and scaling
Sediments	Proportions of sand, silt and clay in sediments specimens
segments.rmult	Draws straight lines from point to point.
SerumProtein	Serum Protein compositions of blood samples
ShiftOperators	Shifts of machine operators
simpleMissingSubplot	Ternary diagrams
SimulatedAmounts	Simulated amount datasets
simulateMissings	Artifical simulation of various kinds of

	missings
Skulls	Measurement of skulls
SkyeAFM	AFM compositions of 23 aphyric Skye lavas
split.acomp	Splitting datasets in groups given by factors
straight	Draws straight lines.
summary.acomp	Summarizing a compositional dataset in terms of ratios
summary.aplus	Summaries of amounts
summary.rcomp	Summary of compositions in real geometry
sumMissingProjector	Compute the global projector to the observed subspace.
Supervisor	Proportions of supervisor's statements assigned to different categories
ternaryAxis	Axis for ternary diagrams
totals	Total sum of amounts
tryDebugger	Empirical variograms for compositions
ult	Uncentered log transform
var.acomp	Variances and covariances of amounts and compositions
variation	Variation matrices of amounts and compositions
var.lm	Residual variance of a model
vcovAcomp	Variance covariance matrix of parameters in compositional regression
vgmFit	Compositional variogram model fitting
vgram2lrvgram	vgram2lrvgram
vgram.sph	Variogram functions
WhiteCells	White-cell composition of 30 blood samples by two different methods
Yatquat	Yatquat fruit evaluation
zeroreplace	Zero-replacement routine

To get detailed "getting started" introduction use `help.start()` or `help.start(browser="myfavouritebrowser")` Go to "Packages" then "compositions" and then "overview" and then launch the file "UsingCompositions.pdf" from there. Please also check the web-site: http://www.stat.boogaart.de/compositions/ for improved material and our new book expected to appear spring 2009.

The package is devoted to the analysis of multiple amounts. Amounts have typically non-negative values, and often sum up to 100% or one. These constraints lead to spurious effects on the covariance structure, as pointed out by Chayes (1960). The problem is treated rigorously in the monography by Aitchison (1986), who characterizes compositions as vectors having a relative scale, and identifies its sample space with the D-part simplex. However still (i.e. 2005) most statistical packages do not provided any support for this scale.

The grounding idea of the package exploits the class concept: the analyst gives the data a compositional or amount class, and then all further analysis are (should be) automatically done in a consistent way, e.g. `x <- acomp(X); plot(x)` should plot the data as a composition (in a ternary diagram) directly without any further interaction of the user.

The package provides four different approaches to analyse amounts. These approaches are associated to four R-classes, representing four different geometries of the sampling space of amounts. These geometries depend on two questions: whether the total sum of the amounts is a relevant information, and which is the meaningful measure of difference of the data.

rplus : (Real Plus) The total amount matters, and amounts should be compared on an absolute basis. i.e. the difference between 1g and 2g is the same as the difference between 1kg and 1001g, one gram.
aplus : (Aitchison Plus) The total amount matters, but amounts should be compared relatively, i.e. the difference between 1mg and 2mg is the same as that of 1g and 2g: the double.
acomp : (Aitchison composition) the total amount is constant (or an artifact of the sampling/measurement procedure), and the meaningful difference is a relative one. This class follows the original proposals of Aitchison.
rcomp : (Real composition) the sum is a constant, and the difference in amount from 0% to 1% and from 10% to 11% is regarded as equal. This class represents the raw/naive treatment of compositions as elements of the real simplex based on an absolute geometry. This treatment is implicitly used in most amalgamation problems. However the whole approach suffers from the drawbacks and problems discussed in Chayes (1960) and Aitchison (1986).
The aim of the package is to provide all the functionality to do a consistent analysis in all of these approaches and to make the results obtained with different geometries as easy to compare as possible.

Note

The package compositions has grown a lot in the last year: missings, robust estimations, outlier detection and classification, codadendrogram. This makes everything much more complex especially from the side of programm testing. Thus we would like to urge our users to report all errors and problems of the lastest version (please check first) to support@boogaart.de.

Author(s)

K. Gerald van den Boogaart <boogaart@hzdr.de>, Raimon Tolosana-Delgado, Matevz Bren

Maintainer: K. Gerald van den Boogaart <support@boogaart.de>

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

Aitchison, J, C. Barcel'o-Vidal, J.J. Egozcue, V. Pawlowsky-Glahn (2002) A consise guide to the algebraic geometric structure of the simplex, the sample space for compositional data analysis, *Terra Nostra*, Schriften der Alfred Wegener-Stiftung, 03/2003

Billheimer, D., P. Guttorp, W.F. and Fagan (2001) Statistical interpretation of species composition, *Journal of the American Statistical Association*, **96** (456), 1205-1214

Chayes, F. (1960). On correlation between variables of constant sum. Journal of Geophysical Research 65~(12), 4185–4193.

Pawlowsky-Glahn, V. and J.J. Egozcue (2001) Geometric approach to statistical analysis on the simplex. *SERRA* **15**(5), 384-398

Pawlowsky-Glahn, V. (2003) Statistical modelling on coordinates. In: Thi\'o -Henestrosa, S. and Mart\'in-Fern\'a ndez, J.A. (Eds.) *Proceedings of the 1st International Workshop on Compositional Data Analysis*, Universitat de Girona, ISBN 84-8458-111-X, https://ima.udg.edu/Activitats/CoDaWork03/

Mateu-Figueras, G. and Barcel\'o-Vidal, C. (Eds.) *Proceedings of the 2nd International Workshop on Compositional Data Analysis*, Universitat de Girona, ISBN 84-8458-222-1, https://ima.udg.edu/Activitats/CoDaWork05/

van den Boogaart, K.G. and R. Tolosana-Delgado (2008) "compositions": a unified R package to analyze Compositional Data, *Computers & Geosciences*, 34 (4), pages 320-338, doi: 10.1016/j.cageo.2006.11.017.

See Also

compositions-package, missingsInCompositions, robustnessInCompositions, outliersInCompositions,

Examples

```
library(compositions)      # load library
data(SimulatedAmounts)     # load data sa.lognormals
x <- acomp(sa.lognormals)  # Declare the dataset to be compositional
                           # and use relative geometry
plot(x)                    # plot.acomp : ternary diagram
ellipses(mean(x),var(x),r=2,col="red")  # Simplex 2sigma predictive region
pr <- princomp(x)
straight(mean(x),pr$Loadings)

x <- rcomp(sa.lognormals)  # Declare the dataset to be compositional
                           # and use absolute geometry
plot(x)                    # plot.acomp : ternary diagram
ellipses(mean(x),var(x),r=2,col="red")  # Real 2sigma predictive region
pr <- princomp(x)
straight(mean(x),pr$Loadings)
```

Aar *Composition of glaciar sediments from the Aar massif (Switzerland)*

Description

Geochemical composition of glaciar sediments from the Aar massif region (Switzerland), major oxides and trace elements.

Usage

```
data(Aar)
```

Details

Composition of recent sediments of several morraines and streams from glaciers around the Aar massif, including both major oxides and trace elements. The major oxides are expressed in weight percent (total sum reported in column `SumOxides`), from Silica (`SiO2`, column 3) to total Iron 3 Oxide (`Fe2O3t`, column12, incorporating FeO recasted to Fe2O3). The trace elements are reported in parts per million (ppm, mg/Kg) between columns 14 (`Ba`) and 29 (`Nd`). Partial sum of the trace elements (in ppm) and of all traces and major oxides (in %) are also reported.

Apart of the compositional information, two covariables are included: Sample and GS. The variable `Sample` reports the ID of the sample material. This material was sieved in 11 grain size fractions, and each fraction was analysed separately after drying. The grain size fraction of each subsample is reported in variable `GS`, representing the upper limit of the size fraction reported in ϕ scale, e.g. the binary log transformation of the average diameter $\bar{d}$

$$\phi = -\log_2(\bar{d})$$

The Aar is a granitic-granodioritic-gneissic massif of the Alps, in Switzerland, comprised of several intrusions with different compositions within the range of granitoid lithologies. Details of the region, mineralogy, procedures and study questions behind the data can be found in von Eynatten at al (2012) and references thereon.

Note

Courtesy of H. von Eynatten

Source

von Eynatten H.; Tolosana-Delgado, R.; Karius, V (2012) Sediment generation in modern glacial settings: Grain-size and source-rock control on sediment composition. Sedimentary Geology 280 (1): 80-92 doi: 10.1016/j.sedgeo.2012.03.008

References

von Eynatten H.; Tolosana-Delgado, R.; Karius, V (2012) Sediment generation in modern glacial settings: Grain-size and source-rock control on sediment composition. Sedimentary Geology 280 (1): 80-92 doi: 10.1016/j.sedgeo.2012.03.008

---

acomp *Aitchison compositions*

Description

A class providing the means to analyse compositions in the philosophical framework of the Aitchison Simplex.

Usage

```
acomp(X,parts=1:NCOL(oneOrDataset(X)),total=1,warn.na=FALSE,
        detectionlimit=NULL,BDL=NULL,MAR=NULL,MNAR=NULL,SZ=NULL)
```

Arguments

X	composition or dataset of compositions
parts	vector containing the indices xor names of the columns to be used
total	the total amount to be used, typically 1 or 100
warn.na	should the user be warned in case of NA,NaN or 0 coding different types of missing values?
detectionlimit	a number, vector or matrix of positive numbers giving the detection limit of all values, all columns or each value, respectively
BDL	the code for 'Below Detection Limit' in X
SZ	the code for 'Structural Zero' in X
MAR	the code for 'Missing At Random' in X
MNAR	the code for 'Missing Not At Random' in X

Details

Many multivariate datasets essentially describe amounts of D different parts in a whole. This has some important implications justifying to regard them as a scale for its own, called a composition. This scale was in-depth analysed by Aitchison (1986) and the functions around the class "acomp" follow his approach.
Compositions have some important properties: Amounts are always positive. The amount of every part is limited to the whole. The absolute amount of the whole is noninformative since it is typically due to artifacts on the measurement procedure. Thus only relative changes are relevant. If the relative amount of one part increases, the amounts of other parts must decrease, introducing spurious anticorrelation (Chayes 1960), when analysed directly. Often parts (e.g H2O, Si) are missing in the dataset leaving the total amount unreported and longing for analysis procedures avoiding spurious effects when applied to such subcompositions. Furthermore, the result of an analysis should be indepent of the units (ppm, g/l, vol.%, mass.%, molar fraction) of the dataset.
From these properties Aitchison showed that the analysis should be based on ratios or log-ratios only. He introduced several transformations (e.g. clr,alr), operations (e.g. perturbe, power.acomp), and a distance (dist) which are compatible with these properties. Later it was found that the set of compostions equipped with perturbation as addition and power-transform as scalar multiplication and the dist as distance form a D-1 dimensional euclidean vector space (Billheimer, Fagan and Guttorp, 2001), which can be mapped isometrically to a usual real vector space by ilr (Pawlowsky-Glahn and Egozcue, 2001).
The general approach in analysing acomp objects is thus to perform classical multivariate analysis on clr/alr/ilr-transformed coordinates and to backtransform or display the results in such a way that they can be interpreted in terms of the original compositional parts.
A side effect of the procedure is to force the compositions to sum up to a *total*, which is done by the closure operation clo .

Value

a vector of class "acomp" representing one closed composition or a matrix of class "acomp" representing multiple closed compositions each in one row.

Missing Policy

The policy of treatment of zeroes, missing values and values below detecion limit is explained in depth in compositions.missing.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de, Raimon Tolosana-Delgado

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

Aitchison, J, C. Barcel'o-Vidal, J.J. Egozcue, V. Pawlowsky-Glahn (2002) A consise guide to the algebraic geometric structure of the simplex, the sample space for compositional data analysis, *Terra Nostra*, Schriften der Alfred Wegener-Stiftung, 03/2003

Billheimer, D., P. Guttorp, W.F. and Fagan (2001) Statistical interpretation of species composition, *Journal of the American Statistical Association*, **96** (456), 1205-1214

Chayes, F. (1960). On correlation between variables of constant sum. Journal of Geophysical Research 65~(12), 4185–4193.

Pawlowsky-Glahn, V. and J.J. Egozcue (2001) Geometric approach to statistical analysis on the simplex. *SERRA* **15**(5), 384-398

Pawlowsky-Glahn, V. (2003) Statistical modelling on coordinates. In: Thi\'o-Henestrosa, S. and Mart\'in-Fern\'andez, J.A. (Eds.) *Proceedings of the 1st International Workshop on Compositional Data Analysis*, Universitat de Girona, ISBN 84-8458-111-X, https://ima.udg.edu/Activitats/CoDaWork03/

Mateu-Figueras, G. and Barcel\'o-Vidal, C. (Eds.) *Proceedings of the 2nd International Workshop on Compositional Data Analysis*, Universitat de Girona, ISBN 84-8458-222-1, https://ima.udg.edu/Activitats/CoDaWork05/

van den Boogaart, K.G. and R. Tolosana-Delgado (2008) "compositions": a unified R package to analyze Compositional Data, *Computers & Geosciences*, 34 (4), pages 320-338, doi: 10.1016/j.cageo.2006.11.017.

See Also

clr,rcomp, aplus, princomp.acomp, plot.acomp, boxplot.acomp, barplot.acomp, mean.acomp, var.acomp, variation.acomp, cov.acomp, msd

Examples

```
data(SimulatedAmounts)
plot(acomp(sa.lognormals))
```

`acomp-class` *Class* `"acomp"`

Description

The S4-version of the data container "acomp" for compositional data. More information in `acomp`

Objects from the Class

A virtual Class: No objects may be directly created from it. This is provided to ensure that acomp objects behave as data.frame or structure under certain circumstances. Use `acomp` to create these objects.

Slots

`.Data`: Object of class `"list"` containing the data itself

`names`: Object of class `"character"` with column names

`row.names`: Object of class `"data.frameRowLabels"` with row names

`.S3Class`: Object of class `"character"` with the class string

Extends

Class `"data.frame"`, directly. Class `"compositional"`, directly. Class `"list"`, by class "data.frame", distance 2. Class `"oldClass"`, by class "data.frame", distance 2. Class `"vector"`, by class "data.frame", distance 3.

Methods

coerce `signature(from = "acomp", to = "data.frame")`: to generate a data.frame

coerce `signature(from = "acomp", to = "structure")`: to generate a structure (i.e. a vector, matrix or array)

coerce<- `signature(from = "acomp", to = "data.frame")`: to overwrite a composition with a data.frame

Note

see `acomp`

Author(s)

Raimon Tolosana-Delgado

References

see acomp

See Also

see acomp

Examples

```
showClass("acomp")
```

---

acomparith *Power transform in the simplex*

Description

The Aitchison Simplex with its two operations perturbation as + and power transform as * is a vector space. This vector space is represented by these operations.

Usage

```
power.acomp(x,s)
## Methods for class "acomp"
## x*y
## x/y
```

Arguments

x	an acomp composition or dataset of compositions (or a number or a numeric vector)
y	a numeric vector of size 1 or nrow(x)
s	a numeric vector of size 1 or nrow(x)

Details

The power transform is the basic multiplication operation of the Aitchison simplex seen as a vector space. It is defined as:

$$(x * y)_i := clo((x_i^{y_i})_i)_i$$

The division operation is just the multiplication with $1/y$.

Value

An "acomp" vector or matrix.

Note

For * the arguments x and y can be exchanged. Note that this definition generalizes the power by a scalar, since y or s may be given as a scalar, or as a vector with as many components as the composition in acomp x. The result is then a matrix where each row corresponds to the composition powered by one of the scalars in the vector.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

Aitchison, J, C. Barcel'o-Vidal, J.J. Egozcue, V. Pawlowsky-Glahn (2002) A consise guide to the algebraic geometric structure of the simplex, the sample space for compositional data analysis, *Terra Nostra*, Schriften der Alfred Wegener-Stiftung, 03/2003

Pawlowsky-Glahn, V. and J.J. Egozcue (2001) Geometric approach to statistical analysis on the simplex. *SERRA* **15**(5), 384-398

https://ima.udg.edu/Activitats/CoDaWork03/

https://ima.udg.edu/Activitats/CoDaWork05/

See Also

ilr,clr, alr,

Examples

```
acomp(1:5)* -1 + acomp(1:5)
data(SimulatedAmounts)
cdata <- acomp(sa.lognormals)
plot( tmp <- (cdata-mean(cdata))/msd(cdata) )
class(tmp)
mean(tmp)
msd(tmp)
var(tmp)
```

acompmargin *Marginal compositions in Aitchison Compositions*

Description

Compute marginal compositions of selected parts, by computing the rest as the geometric mean of the non-selected parts.

Usage

```
acompmargin(X,d=c(1,2),name="*",pos=length(d)+1,what="data")
```

Arguments

X	composition or dataset of compositions
d	vector containing the indices xor names of the columns selected
name	The new name of the amalgamation column
pos	The position where the new amalgamation column should be stored. This defaults to the last column.
what	The role of X either "data" for data (or means) to be transformed or "var" for (acomp-clr)-variances to be transformed.

Details

The amalgamation column is simply computed by taking the geometric mean of the non-selected components. This is consistent with the acomp approach and gives clear ternary diagrams. However, this geometric mean is difficult to interpret.

Value

A closed compositions with class "acomp" containing the variables given by d and the the amalgamation column.

Missing Policy

MNAR has the highest priority, MAR afterwards, and WZERO (BDL,SZ) values are considered as 0 and finally reported as BDL.

Author(s)

Raimon Tolosana-Delgado, K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

Vera Pawlowsky-Glahn (2003) personal communication. Universitat de Girona.

van den Boogaart, K.G. and R. Tolosana-Delgado (2008) "compositions": a unified R package to analyze Compositional Data, *Computers & Geosciences*, 34 (4), pages 320-338, doi: 10.1016/j.cageo.2006.11.017.

See Also

rcompmargin, acomp

Examples

```
data(SimulatedAmounts)
plot.acomp(sa.lognormals5,margin="acomp")
plot.acomp(acompmargin(sa.lognormals5,c("Pb","Zn")))
plot.acomp(acompmargin(sa.lognormals5,c(1,2)))
```

---

acompscalarproduct *inner product for datasets with a vector space structure*

Description

acomp and aplus objects are considered as (sets of) vectors. The %*% is considered as the inner multiplication. An inner multiplication with another vector is the scalar product. An inner multiplication with a matrix is a matrix multiplication, where the vectors are either considered as row or as column vector.

Usage

```
## S3 method for class 'acomp'
x %*% y
## S3 method for class 'aplus'
x %*% y
```

Arguments

x	a acomp or aplus object or a matrix interpreted in clr, ilr or ilt coordinates
y	a acomp or aplus object or a matrix interpreted in clr, ilr or ilt coordinates

Details

The operators try to mimic the behavior of %*% on c()-vectors as inner product, applied in parallel to all row-vectors of the dataset. Thus the product of a vector with a vector of the same type results in the scalar product of both. For the multiplication with a matrix each vector is considered as a row or column, whatever is more appropriate. The matrix itself is considered as representing a linear mapping (endomorphism) of the vector space to a space of the same type. The mapping is represented in clr, ilr or ilt coordinates. Which of the aforementioned coordinate systems is used is judged from the type of x and from the dimensions of the A.

Value

Either a numeric vector containing the scalar products, or an object of type acomp or aplus containing the vectors transformed with the given matrix.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

%*%.rmult

Examples

```
x <- acomp(matrix( sqrt(1:12), ncol= 3 ))
x%*%x
A <- matrix( 1:9,nrow=3)
x %*% A %*% x
x %*% A
A %*% x
A <- matrix( 1:4,nrow=2)
x %*% A %*% x
x %*% A
A %*% x
x <- aplus(matrix( sqrt(1:12), ncol= 3 ))
x%*%x
A <- matrix( 1:9,nrow=3)
x %*% A %*% x
x %*% A
A %*% x
```

Activity10 *Activity patterns of a statistician for 20 days*

Description

Proportion of a day in activity teaching, consulting, administrating, research, other wakeful activities and sleep for 20 days are given.

Usage

```
data(Activity10)
```

Details

The activity of an academic statistician were divided into following six categories

teac	teaching
cons	consultation
admi	administration
rese	research
wake	other wakeful activities
slee	sleep

Data show the proportions of the 24 hours devoted to each activity, recorded on each of 20 days, selected randomly from working days in alternate weeks, so as to avoid any possible carry-over effects, such as short-sleep day being compensated by make-up sleep on a succeeding day.

The six activity may be divided into two categories 'work' comprising activities 1,2,3,4: and 'leisure' comprising activities 5 and 6.

All rows sum to one.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name STATDAY.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison: The Statistical Analysis of Compositional Data, 1986, Data 10, pp15.

`Activity31` *Activity patterns of a statistician for 20 days*

Description

Proportion of a day in activity teaching, consulting, administrating, research, other wakeful activities and sleep for 20 days are given.

Usage

```
data(Activity31)
```

Details

The activity of an academic statistician were divided into following six categories

teac	teaching
cons	consultation
admi	administration
rese	research
wake	other wakeful activities
slee	sleep

Data shows the proportions of the 24 hours devoted to each activity, recorded on each of 20 days, selected randomly from working days in alternate weeks, so as to avoid any possible carry-over effects, such as short-sleep day being compensated by make-up sleep on a succeeding day.

The six activity may be divided into two categories 'work' comprising activities 1,2,3,4: and 'leisure' comprising activities 5 and 6.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name ACTIVITY.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison: The Statistical Analysis of Compositional Data, 1986, Data 31.

---

alr *Additive log ratio transform*

Description

Compute the additive log ratio transform of a (dataset of) composition(s), and its inverse.

Usage

```
alr( x ,ivar=ncol(x), ... )
alrInv( z, ...,orig=gsi.orig(z))
```

Arguments

x	a composition, not necessarily closed
z	the alr-transform of a composition, thus a (D-1)-dimensional real vector
...	generic arguments. not used.
orig	a compositional object which should be mimicked by the inverse transformation. It is especially used to reconstruct the names of the parts.
ivar	The column to be used as denominator variable. Unfortunately not yet supported in alrInv. The default works even if x is a vector.

Details

The alr-transform maps a composition in the D-part Aitchison-simplex non-isometrically to a D-1 dimensonal euclidian vector, treating the last part as common denominator of the others. The data can then be analysed in this transformation by all classical multivariate analysis tools not relying on a distance. The interpretation of the results is relatively simple, since the relation to the original D-1 first parts is preserved. However distance is an extremely relevant concept in most types of analysis, where a clr or ilr transformation should be preferred.

The additive logratio transform is given by

$$alr(x)_i := \ln \frac{x_i}{x_D}$$

.

Value

alr gives the additive log ratio transform; accepts a compositional dataset alrInv gives a closed composition with the given alr-transform; accepts a dataset

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

See Also

clr,ilr,apt, https://ima.udg.edu/Activitats/CoDaWork03/

Examples

```
(tmp <- alr(c(1,2,3)))
alrInv(tmp)
unclass(alrInv(tmp)) - clo(c(1,2,3)) # 0
data(Hydrochem)
cdata <- Hydrochem[,6:19]
pairs(alr(cdata),pch=".")
```

amounts-class *Class* "amounts"

Description

Abstract class containing all amounts classes carrying total information: aplus, rplus and ccomp

Objects from the Class

A virtual Class: No objects may be created from it.

Methods

No methods defined with class "amounts" in the signature.

Author(s)

Raimon Tolosana-Delgado

See Also

compositional-class for classes with relative information

Examples

```
showClass("amounts")
```

AnimalVegetation *Animal and vegetation measurement*

Description

Areal compositions by abundance of vegetation and animals for 50 plots in each of regions A and B.

Usage

```
data(AnimalVegetation)
```

Details

In a regional ecology study, plots of land of equal area were inspected and the parts of each plot which were thick or thin in vegetation and dense or sparse in animals were identified. From this field work the areal proportions of each plot were calculated for the four mutually exclusive and exhaustive categories: thick-dense, thick-sparse, thin-dense, thin-sparse. These sets of proportions are recorded for 50 plots from each of two different regions A and B.

All rows sum to 1, except for some rounding errors.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name ANIVEG.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison: The Statistical Analysis of Compositional Data, 1986, Data 25, pp22.

`aplus` *Amounts analysed in log-scale*

Description

A class to analyse positive amounts in a logistic framework.

Usage

```
aplus(X,parts=1:NCOL(oneOrDataset(X)),total=NA,warn.na=FALSE,
        detectionlimit=NULL,BDL=NULL,MAR=NULL,MNAR=NULL,SZ=NULL)
```

Arguments

`X`	vector or dataset of positive numbers
`parts`	vector containing the indices xor names of the columns to be used
`total`	a numeric vectors giving the total amounts of each dataset.
`warn.na`	should the user be warned in case of NA,NaN or 0 coding different types of missing values?
`detectionlimit`	a number, vector or matrix of positive numbers giving the detection limit of all values, all columns or each value, respectively
`BDL`	the code for 'Below Detection Limit' in X
`SZ`	the code for 'Structural Zero' in X
`MAR`	the code for 'Missing At Random' in X
`MNAR`	the code for 'Missing Not At Random' in X

Details

Many multivariate datasets essentially describe amounts of D different parts in a whole. When the whole is large in relation to the considered parts, such that they do not exclude each other, or when the total amount of each componenten is indeed determined by the phenomenon under investigation and not by sampling artifacts (such as dilution or sample preparation), then the parts can be treated as amounts rather than as a composition (cf. `acomp`, `rcomp`).
Like compositions, amounts have some important properties. Amounts are always positive. An amount of exactly zero essentially means that we have a substance of another quality. Different amounts - spanning different orders of magnitude - are often given in different units (ppm, ppb, g/l, vol.%, mass %, molar fraction). Often, these amounts are also taken as indicators of other non-measured components (e.g. K as indicator for potassium feldspar), which might be proportional to the measured amount. However, in contrast to compositions, amounts themselves do matter. Amounts are typically heavily skewed and in many practical cases a log-transform makes their distribution roughly symmetric, even normal.
In full analogy to Aitchison's compositions, vector space operations are introduced for amounts: the perturbation `perturbe.aplus` as a vector space addition (corresponding to change of units), the power transformation `power.aplus` as scalar multiplication describing the law of mass action, and a distance `dist` which is independent of the chosen units. The induced vector space is mapped isometrically to a classical R^D by a simple log-transformation called `ilt`, resembling classical log transform approaches.
The general approach in analysing aplus objects is thus to perform classical multivariate analysis on ilt-transformed coordinates (i.e., logs) and to backtransform or display the results in such a way that they can be interpreted in terms of the original amounts.
The class aplus is complemented by the `rplus`, allowing to analyse amounts directly as real numbers, and by the classes `acomp` and `rcomp` to analyse the same data as compositions disregarding the total amounts, focusing on relative weights only.
The classes rcomp, acomp, aplus, and rplus are designed as similar as possible in order to allow direct comparison between results achieved by the different approaches. Especially the acomp simplex transforms `clr`, `alr`, `ilr` are mirrored in the aplus class by the single bijective isometric transform `ilt`

Value

a vector of class `"aplus"` representing a vector of amounts or a matrix of class `"aplus"` representing multiple vectors of amounts, each vector in one row.

Missing Policy

The policy of treatment of zeroes, missing values and values below detecion limit is explained in depth in compositions.missing.

Author(s)

Raimon Tolosana-Delgado, K.Gerald v.d. Boogaart `http://www.stat.boogaart.de`

References

van den Boogaart, K.G. and R. Tolosana-Delgado (2008) "compositions": a unified R package to analyze Compositional Data, *Computers & Geosciences*, 34 (4), pages 320-338, doi: 10.1016/

j.cageo.2006.11.017.

See Also

ilt,acomp, rplus, princomp.aplus, plot.aplus, boxplot.aplus, barplot.aplus, mean.aplus, var.aplus, variation.aplus, cov.aplus, msd

Examples

```
data(SimulatedAmounts)
plot(aplus(sa.lognormals))
```

---

aplus-class *Class* "aplus"

Description

The S4-version of the data container "aplus" for compositional data. More information in aplus

Objects from the Class

A virtual Class: No objects may be directly created from it. This is provided to ensure that aplus objects behave as data.frame or structure under certain circumstances. Use aplus to create these objects.

Slots

.Data: Object of class "list" containing the data itself

names: Object of class "character" with column names

row.names: Object of class "data.frameRowLabels" with row names

.S3Class: Object of class "character" with the class string

Extends

Class "data.frame", directly. Class "compositional", directly. Class "list", by class "data.frame", distance 2. Class "oldClass", by class "data.frame", distance 2. Class "vector", by class "data.frame", distance 3.

Methods

coerce signature(from = "aplus", to = "data.frame"): to generate a data.frame

coerce signature(from = "aplus", to = "structure"): to generate a structure (i.e. a vector, matrix or array)

coerce<- signature(from = "aplus", to = "data.frame"): to overwrite a composition with a data.frame

Note

see aplus

Author(s)

Raimon Tolosana-Delgado

References

see aplus

See Also

see aplus

Examples

```
showClass("aplus")
```

---

aplusarithm *vectorial arithmetic for data sets with aplus class*

Description

The positive vectors equipped with the perturbation (defined as the element-wise product) as Abelian sum, and powertransform (defined as the element-wise powering with a scalar) as scalar multiplication forms a real vector space. These vector space operations are defined here in a similar way to +.rmult.

Usage

```
perturbe.aplus(x,y)
## S3 method for class 'aplus'
x + y
## S3 method for class 'aplus'
x - y
## S3 method for class 'aplus'
x * y
## S3 method for class 'aplus'
x / y
##  Methods for aplus
##   x+y
##   x-y
##   -x
##   x*r
##   r*x
##   x/r
power.aplus(x,r)
```

Arguments

x	an aplus vector or dataset of vectors
y	an aplus vector or dataset of vectors
r	a numeric vector of size 1 or nrow(x)

Details

The operators try to mimic the parallel operation of R for vectors of real numbers to vectors of amounts, represented as matrices containing the vectors as rows and works like the operators for {rmult}

Value

an object of class "aplus" containing the result of the corresponding operation on the vectors.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

rmult, %*%.rmult

Examples

```
x <- aplus(matrix( sqrt(1:12), ncol= 3 ))
x
x+x
x + aplus(1:3)
x * 1:4
1:4 * x
x / 1:4
x / 10
power.aplus(x,1:4)
```

---

apt *Additive planar transform*

Description

Compute the additive planar transform of a (dataset of) compositions or its inverse.

Usage

```
apt( x ,...)
aptInv( z ,..., orig=gsi.orig(z))
```

Arguments

x	a composition or a matrix of compositions, not necessarily closed
z	the apt-transform of a composition or a matrix of alr-transforms of compositions
...	generic arguments, not used.
orig	a compositional object which should be mimicked by the inverse transformation. It is especially used to reconstruct the names of the parts.

Details

The apt-transform maps a composition in the D-part real-simplex linearly to a D-1 dimensional euclidian vector. Although the transformation does not reach the whole R^{D-1}, resulting covariance matrices are typically of full rank.

The data can then be analysed in this transformation by all classical multivariate analysis tools not relying on distances. See cpt and ipt for alternatives. The interpretation of the results is easy since the relation to the first D-1 original variables is preserved.

The additive planar transform is given by

$$apt(x)_i := clo(x)_i, i = 1, \ldots, D-1$$

Value

apt gives the centered planar transform, aptInv gives closed compositions with the given apt-transforms

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

van den Boogaart, K.G. and R. Tolosana-Delgado (2008) "compositions": a unified R package to analyze Compositional Data, *Computers & Geosciences*, 34 (4), pages 320-338, doi: 10.1016/j.cageo.2006.11.017.

See Also

alr,cpt,ipt

Examples

```
(tmp <- apt(c(1,2,3)))
aptInv(tmp)
aptInv(tmp) - clo(c(1,2,3)) # 0
data(Hydrochem)
cdata <- Hydrochem[,6:19]
pairs(apt(cdata),pch=".")
```

ArcticLake *Artic lake sediment samples of different water depth*

Description

Sand, silt and clay compositions of 39 sediment samples of different water depth in an Arctic lake.

Usage

```
data(ArcticLake)
```

Details

Sand, silt and clay compositions of 39 sediment samples at different water depth (in meters) in an Arctic lake. The additional feature is a concomitant variable or *covariate,* water depth, which may account for some of the variation in the compositions. In statistical terminology we have a multivariate regression problem with sediment composition as regressand and water depth as regressor.

All row percentage sums to 100, except for rounding errors.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name ARCTIC.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison: The Statistical Analysis of Compositional Data, 1986, Data 5, pp5.

arrows3D *arrows in 3D, based on package rgl*

Description

adds 3-dimensional arrows to an rgl plot.

Usage

```
arrows3D(...)
## Default S3 method:
arrows3D(x0,x1,...,length=0.25,
                    angle=30,code=2,col="black",
                    lty=NULL,lwd=2,orth=c(1,0.0001,0.0000001),
                    labs=NULL,size=lwd)
```

Arguments

x0	a matrix or vector giving the starting points of the arrows
x1	a matrix or vector giving the end points of the arrows
...	additional plotting parameters as described in rgl::material3d
length	a number giving the length of the arrowhead
angle	numeric giving the angle of the arrowhead
code	0=no arrowhead,1=arrowhead at x0,2=arrowhead at x1,3=double headed
col	the color of the arrow
lty	Not implemented, here for compatibility reasons with arrows
lwd	line width in pixels
orth	the flat side of the arrow is not unique by x0 and x1. This ambiguity is solved in a way that the arrow seams as wide as possible from the viewing direction orth.
labs	labels to be plotted to the endpoints of the arrows
size	size of the plotting symbol

Details

The function is called to plot arrows into an rgl plot. The size of the arrow head is given in a absolute way. Therefore it is important to give the right scale for the length, to see the arrow head and that it does not fill the whole window.

Value

the 3D plotting coordinates of the tips of the arrows displayed, returned invisibly

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

plot3D, rgl::points3d, graphics::plot

Examples

```
x <- cbind(rnorm(10),rnorm(10),rnorm(10))
if(requireNamespace("rgl", quietly = TRUE)) {
  plot3D(x)
  x0 <- x*0
  arrows3D(x0,x)
} ## this function requires package 'rgl'
```

`as.data.frame` *Convert "compositions" classes to data frames or matrices*

Description

Convert a compositional object to a dataframe

Usage

```
## S3 method for class 'acomp'
as.data.frame(x,...)
## S3 method for class 'rcomp'
as.data.frame(x,...)
## S3 method for class 'aplus'
as.data.frame(x,...)
## S3 method for class 'rplus'
as.data.frame(x,...)
## S3 method for class 'rmult'
as.data.frame(x,...)
## S3 method for class 'ccomp'
as.data.frame(x,...)
## S3 method for class 'rmult'
as.matrix(x,...)
```

Arguments

x	an object to be converted to a dataframe
...	additional arguments are not used

Value

a data frame containing the given data, or (for rmult only) as matrix.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

Examples

```
data(SimulatedAmounts)
as.data.frame(acomp(sa.groups))
# The central perpose of providing this command is that the following
# works properly:
data.frame(acomp(sa.groups),groups=sa.groups.area)
```

`axis3D` *Drawing a 3D coordiante system to a plot, based on package rgl*

Description

Adds a coordinate system to a 3D rgl graphic. In future releases, functionality to add tickmarks will be (hopefully) provided. Now, it is just a system of arrows giving the directions of the three axes.

Usage

```
axis3D(axis.origin=c(0,0,0),axis.scale=1,axis.col="gray",vlabs=c("x","y","z"),
       vlabs.col=axis.col,bbox=FALSE,axis.lwd=2,axis.len=mean(axis.scale)/10,
       axis.angle=30,orth=c(1,0.0001,0.000001),axes=TRUE,...)
```

Arguments

`axis.origin`	The location where to put the origin of the coordinate arrows typicall either 0, the minimum or the mean of the dataset
`axis.scale`	either a number or a 3D vector giving the length of the arrows for the axis in the coordiantes of the plot
`axis.col`	Color to plot the coordinate system
`vlabs`	The names of the axes, plotted at the end
`vlabs.col`	color for the axes labels
`bbox`	boolean, whether to plot a bounding box
`axis.lwd`	line width of the axes
`axis.angle`	angle of the arrow heads
`axis.len`	length of the arrow heads
`orth`	the orth argument of `arrows3D`
`axes`	a boolean, wether to plot the axes
`...`	these arguments are passed to arrows3D as `rgl::material3d` arguments

Details

The function is called to plot a coordiante system consisting of arrows into an rgl plot.

Value

Nothing

Author(s)

K.Gerald v.d. Boogaart `http://www.stat.boogaart.de`

See Also

rgl::points3d, graphics::plot, plot3D,arrows3D

Examples

```
x <- cbind(rnorm(10),rnorm(10),rnorm(10))
if(requireNamespace("rgl", quietly = TRUE)) {
  plot3D(x)
  x0 <- x*0
  axis3D()
} ## this function requires package 'rgl'
```

---

backtransform *Automatic common backtransformation for compositions*

Description

Functions to automatically determine and compute the relevant back-transformation for a rmult object.

Usage

```
backtransform(x, as=x)
backtransform.rmult(x, as=x)
gsi.orig(x,y=NULL)
gsi.getV(x,y=NULL)
```

Arguments

x	an rmult object to be backtransformed; for both gsi.* functions: an rmult object to extract the relevant information from
as	an rmult object previously obtained with any compositional transformation of this package.
y	for both gsi.* functions: an alternative object to extract the relevant information from, in case that x does not include it

Details

The general idea of this package is to analyse the same data with different geometric concepts, in a fashion as similar as possible. For each of the four concepts there exists a family of transforms expressing the geometry in aan appropriate manner. Transformed data can be further analysed, and certain results may be back-transformed to the original scale. These functions take care of tracking, constructing and computing the inverse transformation, whichever was the original geometry and forward transformation used.

Value

For functions backtransform or backtransform.rmult, a corresponding matrix or vector containing the backtransformation of x. Efforts are taken to keep any extra attributes (beyond, "dim", "dimnames" and "class") the argument "x" may have \ For function gsi.orig, the original data with a compositional class, if it exists (or NULL otherwise). \ For function gsi.getV, the transposed, inverse matrix of log-contrasts originally used to forward transform the original composition orig to its coefficients/coordinates. If it does not exists, the output is NULL.

Author(s)

R. Tolosana-Delgado, K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

van den Boogaart, K.G. and R. Tolosana-Delgado (2008) "compositions": a unified R package to analyze Compositional Data, *Computers & Geosciences*, 34 (4), pages 320-338, doi: 10.1016/j.cageo.2006.11.017.

See Also

cdt, idt, clr, cpt, ilt, iit, ilr, ipt, alr, apt

Examples

```
x <- acomp(1:5)
x
backtransform(ilr(x))
backtransform(clr(x))
backtransform(idt(x))
backtransform(cdt(x))
backtransform(alr(x))
```

---

balance *Compute balances for a compositional dataset.*

Description

Compute balances in a compositional dataset.

Usage

```
balance(X,...)
## S3 method for class 'acomp'
balance(X,expr,...)
## S3 method for class 'rcomp'
balance(X,expr,...)
## S3 method for class 'aplus'
balance(X,expr,...)
```

```
## S3 method for class 'rplus'
balance(X,expr,...)
balance01(X,...)
## S3 method for class 'acomp'
balance01(X,expr,...)
## S3 method for class 'rcomp'
balance01(X,expr,...)
balanceBase(X,...)
## S3 method for class 'acomp'
balanceBase(X,expr,...)
## S3 method for class 'rcomp'
balanceBase(X,expr,...)
## S3 method for class 'acomp'
balanceBase(X,expr,...)
## S3 method for class 'rcomp'
balanceBase(X,expr,...)
```

Arguments

X	compositional dataset (or optionally just its column names for balanceBase)
expr	a ~ formula using the column names of X as variables and seperating them by / and organize by paranthesis (). : and * can be used instead of / when the corresponding balance should not be created. - can be used as an synonym to / in the real geometries. 1 can be used in the unclosed geometries to level against a constant.
...	for future perposes

Details

For acomp-compositions balances are defined as orthogonal projections representing the log ratio of the geometric means of subsets of elements. Based on a recursive subdivision (provided by the expr=) this projections provide a (complete or incomplete) basis of the clr-plane. The basis is given by the balanceBase functions. The transform is given by the balance functions. The balance01 functions are a backtransform of the balances to the amount of the first portion if this was the only balance in a 2 element composition, providing an "interpretation" for the values of the balances.

The package tries to give similar concepts for the other scales. For rcomp objects the concept is mainly unchanges but augmented by a virtual component 1, which always has portion 1.

For rcomp objects, we choose not a "orthogonal" transformation since such a concept anyway does not really exist in the given space, but merily use the difference of one subset to the other. The balance01 is than not really a transform of the balance but simply the portion of the first group of parts in all contrasted parts.

For rplus objects we just used an analog to generalisation from the rcomp defintion as aplus is generalized from acomp. However at this time we have no idea wether this has any usefull interpretation.

Value

balance	a matrix (or vector) with the corresponding balances of the dataset.
balance01	a matrix (or vector) with the corresponding balances in the dataset transformed in the given geometry to a value between 0 and 1.
balanceBase	a matrix (or vector) with column vectors giving the transform in the cdt-transform used to achiev the correponding balances.

References

https://ima.udg.edu/Activitats/CoDaWork08/ Papers of Boogaart and Tolosana https://ima.udg.edu/Activitats/CoDaWork05/ Paper of Egozcue

See Also

clr,ilr,ipt, ilrBase

Examples

```
X <- rnorm(100)
Y <- rnorm.acomp(100,acomp(c(A=1,B=1,C=1)),0.1*diag(3))+acomp(t(outer(c(0.2,0.3,0.4),X,"^")))
colnames(Y) <- c("A","B","C")

subComps <- function(X,...,all=list(...)) {
  X <- oneOrDataset(X)
  nams <- sapply(all,function(x) paste(x[[2]],x[[3]],sep=","))
  val  <- sapply(all,function(x){
              a = X[,match(as.character(x[[2]]),colnames(X)) ]
              b = X[,match(as.character(x[[2]]),colnames(X)) ]
              c = X[,match(as.character(x[[3]]),colnames(X)) ]
              return(a/(b+c))
             })
  colnames(val)<-nams
  val
}
pairs(cbind(ilr(Y),X),panel=function(x,y,...) {points(x,y,...);abline(lm(y~x))})
pairs(cbind(balance(Y,~A/B/C),X),panel=function(x,y,...) {points(x,y,...);abline(lm(y~x))})

pairwisePlot(balance(Y,~A/B/C),X)
pairwisePlot(X,balance(Y,~A/B/C),panel=function(x,y,...) {plot(x,y,...);abline(lm(y~x))})
pairwisePlot(X,balance01(Y,~A/B/C))
pairwisePlot(X,subComps(Y,A~B,A~C,B~C))



balance(rcomp(Y),~A/B/C)
balance(aplus(Y),~A/B/C)
balance(rplus(Y),~A/B/C)
```

barplot.acomp *Bar charts of amounts*

Description

Compositions and amounts dispalyed as bar plots.

Usage

```
## S3 method for class 'acomp'
barplot(height,...,legend.text=TRUE,beside=FALSE,total=1,
plotMissings=TRUE,missingColor="red",missingPortion=0.01)
## S3 method for class 'rcomp'
barplot(height,...,legend.text=TRUE,beside=FALSE,total=1,
plotMissings=TRUE,missingColor="red",missingPortion=0.01)
## S3 method for class 'aplus'
barplot(height,...,legend.text=TRUE,beside=TRUE,total=NULL,
plotMissings=TRUE,missingColor="red",missingPortion=0.01)
## S3 method for class 'rplus'
barplot(height,...,legend.text=TRUE,beside=TRUE,total=NULL,
plotMissings=TRUE,missingColor="red",missingPortion=0.01)
## S3 method for class 'ccomp'
barplot(height,...,legend.text=TRUE,beside=FALSE,total=1,
plotMissings=TRUE,missingColor="red",missingPortion=0.01)
```

Arguments

height	an acomp, rcomp, aplus, or rplus object giving amounts to be displayed
...	further graphical parameters as in barplot
legend.text	same as legend.text in barplot
beside	same as beside in barplot
total	The total to be used in displaying the composition, typically 1, 100 or the number of parts. If NULL no normalisation takes place.
plotMissings	logical: shall missings be annotate in the plot
missingColor	color to draw missings
missingPortion	The space portion to be reserved for missings

Details

These functions are essentially light-weighted wrappers for barplot, just adding an adequate default behavior for each of the scales. The missingplot functionality will work well with the default settings.

If plotMissings is true, there will be an additional portion introduced, which is not counted in the total. This might make the plots looking less nice, however they make clear to the viewer that it is

by no means clear how the rest of the plot should be interpreted and that the missing value really casts some unsureness on the rest of the data.

Value

A numeric vector (or matrix, when `beside = TRUE`) giving the coordinates of all the bar midpoints drawn, as in `barplot`

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

`acomp`, `rcomp`, `rplus` `aplus`, `plot.acomp`, `boxplot.acomp`

Examples

```
data(SimulatedAmounts)
barplot(mean(acomp(sa.lognormals[1:10,])))
barplot(mean(rcomp(sa.lognormals[1:10,])))
barplot(mean(aplus(sa.lognormals[1:10,])))
barplot(mean(rplus(sa.lognormals[1:10,])))

barplot(acomp(sa.lognormals[1:10,]))
barplot(rcomp(sa.lognormals[1:10,]))
barplot(aplus(sa.lognormals[1:10,]))
barplot(rplus(sa.lognormals[1:10,]))

barplot(acomp(sa.tnormals))
```

Bayesite *Permeabilities of bayesite*

Description

Relation of mechanical properties of a new fibreboard with its composition

Usage

```
data(Bayesite)
```

Details

In the development of bayesite, a new fibreboard, experiments were conducted to obtain some insight into the nature of the relationship of its permeability to the mix of its four ingredients

A: short fibres,
B: medium fibres,
C: long fibres,
D: binder,

and the pressure of which these are bonded. The results of 21 such experiments are reported. It is required to investigate the dependence of permeability of mixture and bonding pressure.

All 4-part compositions sum to one.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name BAYESITE.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison: The Statistical Analysis of Compositional Data, 1986, Data 21, pp21.

`binary` *Treating binary and g-adic numbers*

Description

Allows the access to individual digits in binary (and general g-adic) numbers.

Usage

```
binary(x,mb=max(maxBit(x,g)),g=2)
unbinary(x,g=2)
bit(x,b,g=2)
## S3 method for class 'numeric'
bit(x,b=0:maxBit(x,g),g=2)
## S3 method for class 'character'
bit(x,b=0:maxBit(x,g),g=2)
bit(x,b,g=2) <- value
## S3 replacement method for class 'numeric'
bit(x,b=0:maxBit(x,g),g=2) <- value
## S3 replacement method for class 'character'
bit(x,b=0:maxBit(x,g),g=2) <- value
maxBit(x,g=2)
## S3 method for class 'numeric'
maxBit(x,g=2)
## S3 method for class 'character'
maxBit(x,g=2)
bitCount(x,mb=max(maxBit(x,g)),g=2)
gsi.orSum(...,g=2)
whichBits(x,mb=max(maxBit(x,g)),g=2,values=c(TRUE))
binary2logical(x,mb=max(maxBit(x,g)),g=2,values=c(TRUE))
```

Arguments

x	a number either represented a g-adic character string or as a integeral numeric value
b	the indicies of the bits to be processes. The least significant bit has index 0.
mb	maximal bit. The index of the most significant bit to be treated
g	the base of the g-adic representation. 2 corresponds to binary numbers, 8 to octal numbers, 16 to hexadecimal numbers. g is limited by 36.
value	a vector of bit values to be selected or setted.
values	a vector of bit values that should be considered as TRUE.
...	some binary numbers

Details

These routines are primerily intended to manipulate g-adic numbers for user interface purposes and condensed representation of information. They are not intended for a long number arithmetic.

Value

binary	returns a standard binary (or g-adic) character representation of the number
unbinary	returns a binary (or g-adic) representation of the number
bit	returns the values of the requested bits. The values are returned as a logical vector for binary numbers an as numeric digit values for other g-adic numbers.
maxBit	returns the most significant bit represented in the number. This is the highest bit set in numeric numbers and the highest actually given character in a character representation.
bitCount	returns the g-adic crossfoot of the number. For a binary number this is the number of bits set
gsi.orSum	Only works for binary numbers and does a parallel or on each of the bits for a list of binary numbers.
whichBits	returns the indices of the bits acutally set (or more precisely of the bits with value in values)
binary2logical	returns the a true false vector of the bits acutally set (or more precisely of the bits with value in values)

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

outlierplot

Examples

```
(x<-unbinary("10101010"))
(y<-binary(x))
bit(x,1:3)
bit(y,0:3)
maxBit(x)
maxBit(y)
whichBits(x)
whichBits(y)
binary2logical(y)
bit(x)
bit(y)
gsi.orSum(y,1)
bitCount(x)
bitCount(y)
bit(x,2)<-1
x
bit(y,2)<-1
y
```

`biplot3D` *Three-dimensional biplots, based on package rgl*

Description

Plots variables and cases in the same plot, based on a principal component analysis.

Usage

```
biplot3D(x,...)
## Default S3 method:
biplot3D(x,y,var.axes=TRUE,col=c("green","red"),cex=c(2,2),
            xlabs = NULL, ylabs = NULL, expand = 1,arrow.len = 0.1,
            ...,add=FALSE)
 ## S3 method for class 'princomp'
biplot3D(x,choices=1:3,scale=1,...,
            comp.col=1,comp.labs=paste("Comp.",1:3),
            scale.scores=lambda[choices]^(1-scale),
            scale.var=scale.comp, scale.comp=sqrt(lambda[choices]),
            scale.disp=1/scale.comp)
```

Arguments

`x`	princomp object or matrix of point locations to be drawn (typically, cases)
`choices`	Which principal components should be used?
`scale`	a scaling parameter like in `biplot`

scale.scores	a vector giving the scaling applied to the scores
scale.var	a vector giving the scaling applied to the variables
scale.comp	a vector giving the scaling applied to the unit length of each component
scale.disp	a vector giving the scaling of the display in the directions of the components
comp.col	color to draw the axes of the components, defaults to black
comp.labs	labels for the components
...	further plotting parameters as defined in rgl::material3d
y	matrix of second point/arrow-head locations (typically, variables)
var.axes	logical, TRUE draws arrows and FALSE points for y
col	vector/list of two elements the first giving the color/colors for the first data set and the second giving color/colors for the second data set.
cex	vector/list of two elements the first giving the size for the first data set and the second giving size for the second data set.
xlabs	labels to be plotted at x-locations
ylabs	labels to be plotted at y-locations
expand	the relative expansion of the y data set with respect to x
arrow.len	The length of the arrows as defined in arrows3D
add	logical, adding to existing plot or making a new one?

Details

This "biplot" is a triplot, relating data, variables and principal components. The relative scaling of the components is still experimental, meant to mimic the behavior of classical biplots.

Value

the 3D plotting coordinates of the tips of the arrows of the variables displayed, returned invisibly

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

gsi

Examples

```
data(SimulatedAmounts)
pc <- princomp(acomp(sa.lognormals5))
pc
summary(pc)
plot(pc)      #plot(pc,type="screeplot")
biplot(pc)
if(requireNamespace("rgl", quietly = TRUE)) {
  biplot3D(pc)
} ## this function requires package 'rgl'
```

Blood23 *Blood samples*

Description

Percentage of different leukocytes in blood samples of ten patients, determined by four different methods.

Usage

```
data(Blood23)
```

Details

In a comparative study of four different methods of assessing leukocytes composition of a blood sample, aliquots of blood samples from ten patients were assessed by the four methods. Data show the percentage in the 40 analyses of:

P: polymorphonuclear leukocytes,
S: small lymphocytes,
L: large mononuclears.

All rows sum to 100.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name BLOOD.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison: The Statistical Analysis of Compositional Data, 1986, Data 23, pp22.

Boxite *Compositions and depth of 25 specimens of boxite*

Description

A mineral compositions of 25 rock specimens of boxite type. Each composition consists of the percentage by weight of five minerals, albite, blandite, cornite, daubite, endite, as well as the depth of location.

Usage

```
data(Boxite)
```

Details

A mineral compositions of 25 rock specimens of boxite type are given. Each composition consists of the percentage by weight of five minerals, albite, blandite, cornite, daubite, endite, and the recorded depth of location of each specimen. We abbreviate the minerals names to A, B, C, D, E.

All row percentage sums to 100, except the 6-th 102.3 and the 10-th 99.9.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name BOXITE.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison: The Statistical Analysis of Compositional Data, 1986, Data 3, pp4.

---

boxplot *Displaying compositions and amounts with box-plots*

Description

For the different interpretations of amounts or compositional data, a different type of boxplot is feasible. Thus different boxplots are drawn.

Usage

```
## S3 method for class 'acomp'
boxplot(x,fak=NULL,...,
                        xlim=NULL,ylim=NULL,log=TRUE,
                        panel=vp.logboxplot,dots=!boxes,boxes=TRUE,
                         notch=FALSE,
                         plotMissings=TRUE,
                         mp=~simpleMissingSubplot(missingPlotRect,
                                                  missingInfo,c("NM","TM",cn))
                         )
## S3 method for class 'rcomp'
boxplot(x,fak=NULL,...,
                        xlim=NULL,ylim=NULL,log=FALSE,
                        panel=vp.boxplot,dots=!boxes,boxes=TRUE,
                         notch=FALSE,
```

```
                          plotMissings=TRUE,
                          mp=~simpleMissingSubplot(missingPlotRect,
                                                   missingInfo,c("NM","TM",cn)))
## S3 method for class 'aplus'
boxplot(x,fak=NULL,...,log=TRUE,
                          plotMissings=TRUE,
                          mp=~simpleMissingSubplot(missingPlotRect,
                                                    missingInfo,
                                                    names(missingInfo)))
## S3 method for class 'rplus'
boxplot(x,fak=NULL,...,ylim=NULL,log=FALSE,
                          plotMissings=TRUE,
                          mp=~simpleMissingSubplot(missingPlotRect,
                                                    missingInfo,
                                                    names(missingInfo)))
vp.boxplot(x,y,...,dots=FALSE,boxes=TRUE,xlim=NULL,ylim=NULL,log=FALSE,
                          notch=FALSE,plotMissings=TRUE,
                          mp=~simpleMissingSubplot(missingPlotRect,
                                                   missingInfo,c("NM","TM",cn)),
                          missingness=attr(y,"missingness") )
vp.logboxplot(x,y,...,dots=FALSE,boxes=TRUE,xlim,ylim,log=TRUE,notch=FALSE,
                          plotMissings=TRUE,
                          mp=~simpleMissingSubplot(missingPlotRect,
                                                   missingInfo,c("NM","TM",cn)),
                          missingness=attr(y,"missingness"))
```

Arguments

x	a data set
fak	a factor to split the data set, not yet implemented in aplus and rplus
xlim	x-limits of the plot.
ylim	y-limits of the plot.
log	logical indicating whether ploting should be done on log scale
panel	the panel function to be used or a list of multiple panel functions
...	further graphical parameters
dots	a logical indicating whether the points should be drawn
boxes	a logical indicating whether the boxes should be drawn
y	used by pairs
notch	logical, should the boxes be notched?
plotMissings	Logical indicating that missings should be displayed.
mp	A formula providing a function call, which will be evaluated within each panel with missings to plot the missingness situation. The call can use the variables missingPlotRect, which provides a rectangle to plot the information to in a par("usr") like specification. In the rX is the current data
missingness	The missingness information as a result from missingType of the full data information the panels could base there missing plots on.

Details

boxplot.aplus and boxplot.rplus are wrappers of bxp, which just take into account the possible logarithmic scale of the data.

boxplot.acomp and boxplot.rcomp generate a matrix of box-plots, where each cell represents the difference between the row and column variables. Such difference is respectively computed as a log-ratio and a rest.

vp.boxplot and vp.logboxplot are only used as panel functions. They should not be directly called.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

plot.acomp, qqnorm.acomp

Examples

```
data(SimulatedAmounts)
boxplot(acomp(sa.lognormals))
boxplot(rcomp(sa.lognormals))
boxplot(aplus(sa.lognormals))
boxplot(rplus(sa.lognormals))
# And now with missing!!!
boxplot(acomp(sa.tnormals))
```

---

ccomp *Count compositions*

Description

A class providing the means to analyse count compositions understood as Poisson or multinomial realisation, where the portions are given by an unkown Aitchison compositions.

Usage

```
ccomp(X,parts=1:NCOL(oneOrDataset(X)),total=NA,warn.na=FALSE,
        detectionlimit=NULL,BDL=NULL,MAR=NULL,MNAR=NULL,SZ=NULL)
```

Arguments

X	composition or dataset of compositions
parts	vector containing the indices xor names of the columns to be used
total	the total amount to be used, typically 1 or 100
warn.na	should the user be warned in case of NA,NaN or 0 coding different types of missing values?
detectionlimit	a number, vector or matrix of positive numbers giving the detection limit of all values, all columns or each value, respectively
BDL	the code for 'Below Detection Limit' in X
SZ	the code for 'Structural Zero' in X
MAR	the code for 'Missing At Random' in X
MNAR	the code for 'Missing Not At Random' in X

Details

A count composition contains an indirect observation of an Aitchison composition by a Poisson or multinomial variable. A count composition can only contain integer counts. It is assumed that the total sum is a an artefact and does not contain information on the actual composition. But it does contain information on the precision of the relative observation.

Value

a vector of class "ccomp" representing count composition or a matrix of class "ccomp" representing multiple count compositions each in one row.

Missing Policy

The policy of treatment of zeroes, missing values and values below detecion limit is explained in depth in compositions.missing.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

barplot.ccomp ccompMultinomialGOF.test ccompPoissonGOF.test cdt.ccomp cdtInv.ccomp fitSameMeanDifferentVarianceModel groupparts.ccomp idt.ccomp idtInv.ccomp is.ccomp mean.ccomp names<-.ccomp names.ccomp plot.ccomp PoissonGOF.test rmultinom.ccomp rnorm.ccomp rpois.ccomp split.ccomp totals.ccomp

Examples

```
data(SimulatedAmounts)
plot(acomp(sa.lognormals))
```

ccomp-class *Class* "ccomp"

Description

The S4-version of the data container "ccomp" for compositional data. More information in ccomp

Objects from the Class

A virtual Class: No objects may be directly created from it. This is provided to ensure that ccomp objects behave as data.frame or structure under certain circumstances. Use ccomp to create these objects.

Slots

.Data: Object of class "list" containing the data itself

names: Object of class "character" with column names

row.names: Object of class "data.frameRowLabels" with row names

.S3Class: Object of class "character" with the class string

Extends

Class "data.frame", directly. Class "compositional", directly. Class "list", by class "data.frame", distance 2. Class "oldClass", by class "data.frame", distance 2. Class "vector", by class "data.frame", distance 3.

Methods

coerce signature(from = "ccomp", to = "data.frame"): to generate a data.frame

coerce signature(from = "ccomp", to = "structure"): to generate a structure (i.e. a vector, matrix or array)

coerce<- signature(from = "ccomp", to = "data.frame"): to overwrite a composition with a data.frame

Note

see ccomp

Author(s)

Raimon Tolosana-Delgado

References

see ccomp

See Also

see ccomp

Examples

```
showClass("ccomp")
```

---

ccompgof *Compositional Goodness of fit test*

Description

Goodness of fit tests for count compositional data.

Usage

```
PoissonGOF.test(x,lambda=mean(x),R=999,estimated=missing(lambda))
ccompPoissonGOF.test(x,simulate.p.value=TRUE,R=1999)
ccompMultinomialGOF.test(x,simulate.p.value=TRUE,R=1999)
```

Arguments

x	a dataset integer numbers (PoissonGOF) or count compositions (compPoissonGOF)
lambda	the expected value to check against
R	The number of replicates to compute the distribution of the test statistic
estimated	states whether the lambda parameter should be considered as estimated for the computation of the p-value.
simulate.p.value	should all p-values be infered by simulation.

Details

The compositional goodness of fit testing problem is essentially a multivariate goodness of fit test. However there is a lack of standardized multivariate goodness of fit tests in R. Some can be found in the energy-package.

In principle there is only one test behind the Goodness of fit tests provided here, a two sample test with test statistic.

$$\frac{\sum_{ij} k(x_i, y_i)}{\sqrt{\sum_{ij} k(x_i, x_i) \sum_{ij} k(y_i, y_i)}}$$

The idea behind that statistic is to measure the cos of an angle between the distributions in a scalar product given by

$$(X, Y) = E[k(X, Y)] = E[\int K(x - X)K(x - Y)dx]$$

where k and K are Gaussian kernels with different spread. The bandwith is actually the standarddeviation of k.
The other goodness of fit tests against a specific distribution are based on estimating the parameters of the distribution, simulating a large dataset of that distribution and apply the two sample goodness of fit test.

Value

A classical "htest" object

data.name	The name of the dataset as specified
method	a name for the test used
alternative	an empty string
replicates	a dataset of p-value distributions under the Null-Hypothesis got from nonparametric bootstrap
p.value	The p.value computed for this test

Missing Policy

Up to now the tests can not handle missings.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

See Also

fitDirichlet,rDirichlet, runif.acomp, rnorm.acomp,

Examples

```
## Not run:
x <- runif.acomp(100,4)
y <- runif.acomp(100,4)

erg <- acompGOF.test(x,y)
#continue
erg
unclass(erg)
erg <- acompGOF.test(x,y)


x <- runif.acomp(100,4)
y <- runif.acomp(100,4)
```

```
dd <- replicate(1000,acompGOF.test(runif.acomp(100,4),runif.acomp(100,4))$p.value)
hist(dd)

dd <- replicate(1000,acompGOF.test(runif.acomp(20,4),runif.acomp(100,4))$p.value)
hist(dd)
dd <- replicate(1000,acompGOF.test(runif.acomp(10,4),runif.acomp(100,4))$p.value)

hist(dd)
dd <- replicate(1000,acompGOF.test(runif.acomp(10,4),runif.acomp(400,4))$p.value)
hist(dd)
dd <- replicate(1000,acompGOF.test(runif.acomp(400,4),runif.acomp(10,4),bandwidth=4)$p.value)
hist(dd)


dd <- replicate(1000,acompGOF.test(runif.acomp(20,4),runif.acomp(100,4)+acomp(c(1,2,3,1)))$p.value)

hist(dd)


x <- runif.acomp(100,4)
acompUniformityGOF.test(x)

dd <- replicate(1000,acompUniformityGOF.test(runif.acomp(10,4))$p.value)

hist(dd)


## End(Not run)
```

---

cdt *Centered default transform*

Description

Compute the centered default transform of a (data set of) compositions or amounts (or its inverse).

Usage

```
          cdt(x,...)
          ## Default S3 method:
cdt( x ,...)
          ## S3 method for class 'acomp'
cdt( x ,...)
          ## S3 method for class 'rcomp'
cdt( x ,...)
          ## S3 method for class 'aplus'
cdt( x ,...)
          ## S3 method for class 'rplus'
cdt( x ,...)
```

```
          ## S3 method for class 'rmult'
cdt( x ,...)
          ## S3 method for class 'ccomp'
cdt( x ,...)
          ## S3 method for class 'factor'
cdt( x ,...)
          ## S3 method for class 'data.frame'
cdt( x ,...)
          cdtInv(x,orig=gsi.orig(x),...)
          ## Default S3 method:
cdtInv( x ,orig=gsi.orig(x),...)
          ## S3 method for class 'acomp'
cdtInv( x ,orig=gsi.orig(x),...)
          ## S3 method for class 'rcomp'
cdtInv( x ,orig=gsi.orig(x),...)
          ## S3 method for class 'aplus'
cdtInv( x ,orig=gsi.orig(x),...)
          ## S3 method for class 'rplus'
cdtInv( x ,orig=gsi.orig(x),...)
          ## S3 method for class 'rmult'
cdtInv( x ,orig=gsi.orig(x),...)
          ## S3 method for class 'ccomp'
cdtInv( x ,orig=gsi.orig(x),...)
          ## S3 method for class 'factor'
cdtInv( x ,orig=gsi.orig(x),...)
          ## S3 method for class 'data.frame'
cdtInv( x ,orig=gsi.orig(x),...)
```

Arguments

x	a classed (matrix of) amount or composition, to be transformed with its centered default transform, or its inverse; in case of the method for `data.frame` objects, the function attempts to track information about a previous class (in an attribute `origClass`, and if found, a cdt/Inv transformation is tried with it; for factors, cdt expands the factor in indicator values for each category, or vice-versa.)
...	generic arguments past to underlying functions.
orig	a compositional object which should be mimicked by the inverse transformation. It is used to determine the backtransform to be used and eventually to reconstruct the names of the parts. It is the generic argument. Typically this argument is extracted from x, but if this fails you can give the data set that has be transformed in the first place.

Details

The general idea of this package is to analyse the same data with different geometric concepts, in a fashion as similar as possible. For each of the four concepts there exists a unique transform expressing the geometry in a linear subspace, keeping the relation to the variables. This unique

transformation is computed by cdt. For acomp the transform is clr, for rcomp it is cpt, for aplus it is ilt, and for rplus it is iit. Each component of the result is identified with a unit vector in the direction of the corresponding component of the original composition or amount. Keep in mind that the transform is not necessarily surjective and thus variances in the image space might be singular.

Value

A corresponding matrix or vector containing the transforms. (Exception: cdt.data.frame can return a data.frame if the input has no "origClass"-attribute)

Author(s)

R. Tolosana-Delgado, K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

van den Boogaart, K.G. and R. Tolosana-Delgado (2008) "compositions": a unified R package to analyze Compositional Data, *Computers & Geosciences*, 34 (4), pages 320-338, doi: 10.1016/j.cageo.2006.11.017.

See Also

backtransform, idt, clr, cpt, ilt, iit

Examples

```
## Not run:
# the cdt is defined by
cdt         <- function(x) UseMethod("cdt",x)
cdt.default <- function(x) x
cdt.acomp   <- clr
cdt.rcomp   <- cpt
cdt.aplus   <- ilt
cdt.rplus   <- iit

## End(Not run)
x <- acomp(1:5)
(ds <- cdt(x))
cdtInv(ds,x)
(ds <- cdt(rcomp(1:5)))
cdtInv(ds,rcomp(x))
  data(Hydrochem)
  x = Hydrochem[,c("Na","K","Mg","Ca")]
  y = acomp(x)
  z = cdt(y)
  y2 = cdtInv(z,y)
  par(mfrow=c(2,2))
  for(i in 1:4){plot(y[,i],y2[,i])}
```

ClamEast — *Color-size compositions of 20 clam colonies from East Bay*

Description

From East Bay, 20 clam colonies were randomly selected and from each a sample of clams were taken. Each sample was sieved into three size ranges, large, medium, and small. Then each size range was sorted by the shale colour, dark or light.

Usage

```
data(ClamEast)
```

Details

The data consist of 20 cases and 2×3 variables denoted:

dl	portion of dark large clams,
dm	portion of dark medium clams,
ds	portion of dark small clams,
ll	portion of light large clams,
lm	portion of light medium clams,
ls	portion of light small clams.

All 6-part compositions sum to one, except for rounding errors.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name CLAMEAST.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison: The Statistical Analysis of Compositional Data, 1986, Data 14, pp18.

ClamWest — *Color-size compositions of 20 clam colonies from West Bay*

Description

From West Bay, 20 clam colonies were randomly selected, and from each a sample of clams were taken. Each sample was sieved into three size ranges, large, medium, and small. Then each size range was sorted by the shale colour, dark or light.

Usage

```
data(ClamWest)
```

Details

The data consist of 20 cases and 2×3 variables denoted:

dl	portion of dark large clams,
dm	portion of dark medium clams,
ds	portion of dark small clams,
ll	portion of light large clams,
lm	portion of light medium clams,
ls	portion of light small clams.

All 6-part compositions sum up to one.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name CLAMWEST.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison: The Statistical Analysis of Compositional Data, 1986, Data 15, pp17.

---

clo *Closure of a composition*

Description

Closes compositions to sum up to one (or an optional total), by dividing each part by the sum.

Usage

```
clo( X, parts=1:NCOL(oneOrDataset(X)),total=1,
     detectionlimit=attr(X,"detectionlimit"),
     BDL=NULL,MAR=NULL,MNAR=NULL,SZ=NULL,
     storelimit=!is.null(attr(X,"detectionlimit"))
     )
```

Arguments

X	composition or dataset of compositions
parts	vector containing the indices xor names of the columns to be used
total	the total amount to which the compositions should be closed; either a single number, or a numeric vector of length gsi.getN(X) specifying a different total for each compositional vector in the dataset.
detectionlimit	a number, vector or matrix of positive numbers giving the detection limit of all values, all variables, or each value
BDL	the code for values below detection limit in X
SZ	the code for structural zeroes in X
MAR	the code for values missed at random in X
MNAR	the code for values missed not at random in X
storelimit	a boolean indicating wether to store the detection limit as an attribute in the data. It defaults to FALSE if the detection limit is not already stored in the dataset. The attribute is only needed for very advanced analysis. Most times, this will not be used.

Details

The closure operation is given by

$$clo(x) := \left(x_i / \sum_{j=1}^{D} x_j \right)$$

clo generates a composition without assigning one of the compositional classes acomp or rcomp. Note that after computing the closed-to-one version, obtaining a version closed to any other value is done by simple multiplication.

Value

a composition or a data matrix of compositions, maybe without compositional class. The individual compositions are forced to sum to 1 (or to the optionally-specified total). The result should have the same shape as the input (vector, row, matrix).

Missing Policy

How missing values are coded in the output always follows the general rules described in compositions.missing. The BDL values are accordingly scaled during the scaling operations but not taken into acount for the calculation of the total sum.

Note

clo can be used to unclass compositions.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de, Raimon Tolosana-Delgado

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

See Also

clr,acomp,rcomp

Examples

```
(tmp <- clo(c(1,2,3)))
clo(tmp,total=100)
data(Hydrochem)
plot( clo(Hydrochem,8:9) ) # Giving points on a line
```

---

clr *Centered log ratio transform*

Description

Compute the centered log ratio transform of a (dataset of) composition(s) and its inverse.

Usage

```
clr( x,... )
clrInv( z,..., orig=gsi.orig(z) )
```

Arguments

x	a composition or a data matrix of compositions, not necessarily closed
z	the clr-transform of a composition or a data matrix of clr-transforms of compositions, not necessarily centered (i.e. summing up to zero)
...	for generic use only
orig	a compositional object which should be mimicked by the inverse transformation. It is especially used to reconstruct the names of the parts.

Details

The clr-transform maps a composition in the D-part Aitchison-simplex isometrically to a D-dimensonal euclidian vector subspace: consequently, the transformation is not injective. Thus resulting covariance matrices are always singular.
The data can then be analysed in this transformation by all classical multivariate analysis tools not relying on a full rank of the covariance. See `ilr` and `alr` for alternatives. The interpretation of the results is relatively easy since the relation between each original part and a transformed variable is preserved.
The centered logratio transform is given by

$$clr(x) := \left(\ln x_i - \frac{1}{D} \sum_{j=1}^{D} \ln x_j \right)_i$$

The image of the `clr` is a vector with entries summing to 0. This hyperplane is also called the clr-plane.

Value

`clr` gives the centered log ratio transform, `clrInv` gives closed compositions with the given clr-transform

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data*, Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

See Also

`ilr`,`alr`,`apt`

Examples

```
(tmp <- clr(c(1,2,3)))
clrInv(tmp)
clrInv(tmp) - clo(c(1,2,3)) # 0
data(Hydrochem)
cdata <- Hydrochem[,6:19]
pairs(clr(cdata),pch=".")
```

clr2ilr *Convert between clr and ilr, and between cpt and ipt.*

Description

Compute the centered log ratio transform of a (dataset of) from isometric log-ratio transform(s) and its inverse. Equivalently, compute centered and isometric planar transforms from each other. Acts in vectors and in bilinear forms. For bilinear forms, transform between variation-form from clr-form.

Usage

```
clr2ilr( x , V=ilrBase(x=x) )
ilr2clr( z , V=ilrBase(z=z), x=gsi.orig(z) )
clrvar2ilr( varx , V=ilrBase(D=ncol(varx)) )
ilrvar2clr( varz , V=ilrBase(D=ncol(varz)+1) ,x=NULL)
clrvar2variation(Sigma)
variation2clrvar(TT)
is.clrvar(M, tol=1e-10)
is.ilrvar(M, tol=1e-10)
```

Arguments

x	the clr/cpt-transform of composition(s) (in the ilr2-routines provided only to give column names.)
z	the ilr/ipt-transform of composition(s)
varx, Sigma	variance or covariance matrix of clr/cpt-transformed compositions
varz	variance or covariance matrix of ilr/ipt-transformed compositions
V	a matrix with columns giving the chosen basis of the clr-plane
TT	variation matrix
M	a matrix, to check if it is a valid variance
tol	tolerance for the check

Details

These functions perform a matrix multiplication with V in an appropriate way.

Value

clr2ilr gives the ilr/ipt transform of the same composition(s),
ilr2clr gives the clr/cpt transform of the same composition(s),
clrvar2ilr gives the variance-/covariance-matrix of the ilr/ipt transform of the same compositional data set,
ilrvar2clr and clrvar2variation give the variance-/covariance-matrix of the clr/cpt transform

of the same compositional data set.
`variation2clrvar` gives the variation matrix from the clr-covariance matrix
`is.*var` check if the given matrix satisfies the conditions to be an ilr-variance resp. a clr-variance

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

Egozcue J.J., V. Pawlowsky-Glahn, G. Mateu-Figueras and C. Barcel'o-Vidal (2003) Isometric logratio transformations for compositional data analysis. *Mathematical Geology*, **35**(3) 279-300
Aitchison, J, C. Barcel'o-Vidal, J.J. Egozcue, V. Pawlowsky-Glahn (2002) A consise guide to the algebraic geometric structure of the simplex, the sample space for compositional data analysis, *Terra Nostra*, Schriften der Alfred Wegener-Stiftung, 03/2003

See Also

`variation`, `ilr`, `ipt`, `clr`, `cpt`

Examples

```
data(SimulatedAmounts)
ilrInv(clr2ilr(clr(sa.lognormals)))-clo(sa.lognormals)
clrInv(ilr2clr(ilr(sa.lognormals)))-clo(sa.lognormals)
ilrvar2clr(var(ilr(sa.lognormals)))-var(clr(sa.lognormals))
clrvar2ilr(var(cpt(sa.lognormals)))-var(ipt(sa.lognormals))
variation(acomp(sa.lognormals))
clrvar2variation(var(acomp(sa.lognormals)))
```

ClusterFinder1 *Heuristics to find subpopulations of outliers*

Description

The ClusterFinder is a heuristic to find subpopulations of outliers essentially by looking for secondary modes in a density estimate.

Usage

```
ClusterFinder1(X,...)
## S3 method for class 'acomp'
ClusterFinder1(X,...,sigma=0.3,radius=1,asig=1,minGrp=3,
                                  robust=TRUE)
```

Arguments

X	the dataset to be clustered
...	Further arguments to MahalanobisDist(X,...,robust=robust,pairwise=TRUE)
sigma	numeric: The Bandwidth of the density estimation kernel in a robustly Mahalanobis transformed space. (i.e. in the transform, where the main group has unit variance)
radius	The minimum size of a cluster in a robustly Mahalanobis transformed space. (i.e. in the transform, where the main group has unit variance)
asig	a scaling factor for the geometry of the robustly Mahalanobis transformed space when computing the likelihood of an observation to belong to group (under a Gaussian assumption). Higher values
minGrp	the minimum size of group to be used. Smaller groups are treated as single outliers
robust	A robustness description for estimating the variance of the main group. FALSE is probably not a usefull value. However later other robustness techniques than mcd might be usefull. TRUE just picks the default method of the package.

Details

See outliersInCompositions for a comprehensive introduction into the outlier treatment in compositions.
The ClusterFinder is labeled with a number to make clear that this is just an implementation of some heuristic and not based on some eternal truth. Other might give better Clusterfinders.
Unlike other Clustering Algorithms the basic model of this algorithm assumes that there is one dominating subpopulation and an unkown number of smaller subpopulations with a similar covariance structure but a different mean. The algorithm thus first estimates the covariance structure of the main population by a robust location scale estimator. Then it uses a simplified (Gaussian) kernel density estimator to find nonrandom secondary modes. The it tries to a assign the different observations according to discrimination analysis model to the different modes. Groups under a given size are considered as single outliers forming a seperate group. In this way the number of clusters is kept low even if there are many erratic measurements in the dataset.
The main use of the clusters is descriptive plotting. The advantage of these cluster against other cluster techniques like k-mean or hclust is that it does not tear appart the central mass of the data, as these methods do to make the clusters as compact as possible.

Value

A list

types	a factor representing the group assignments, when the small groups are ignored
typesTbl	a table giving the number of members in each of these groups
groups	a factor representing the found group assignments
isMax	a logical vector indicating for each observation,whether it represent a local maximum in the density estimate.
prob	the infered probability to belong to the different groups given as an acomp composition.

nmembers	a tabel giving the number of members of each group
density	the density estimated in each observation location
likeli	The infered likelihood see this observation, for each of the groups

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

hclust, kmeans

Examples

```
data(SimulatedAmounts)
  cl <- ClusterFinder1(sa.outliers5,sigma=0.4,radius=1)
  plot(sa.outliers5,col=as.numeric(cl$types),pch=as.numeric(cl$types))
  legend(1,1,legend=levels(cl$types),xjust=1,col=1:length(levels(cl$types)),
                    pch=1:length(levels(cl$types)))
```

CoDaDendrogram *Dendrogram representation of acomp or rcomp objects*

Description

Function for plotting CoDa-dendrograms of acomp or rcomp objects.

Usage

```
CoDaDendrogram(X, V = NULL, expr=NULL, mergetree = NULL, signary = NULL,
    range = c(-4,4), ..., xlim = NULL, ylim = NULL, yaxt = NULL, box.pos = 0,
    box.space = 0.25, col.tree = "black", lty.tree = 1, lwd.tree = 1,
    col.leaf = "black", lty.leaf = 1, lwd.leaf = 1, add = FALSE,border=NULL,
    type = "boxplot")
```

Arguments

X	data set to plot (an rcomp or acomp object)
V	basis to use, described as an ilr matrix
expr	a formula describing the partition basis, as with balanceBase
mergetree	basis to use, described as a merging tree (as in hclust)
signary	basis to use, described as a sign matrix (as in the example below)
range	minimum and maximum value for all coordinates (horizontal axes)

`...`	further parameters to pass to any function, be it a plotting function or one related to the "type" parameter below; likely to produce lots of warnings
`xlim`	minimum and maximum values for the horizontal direction of the plot (related to number of parts)
`ylim`	minimum and maximum values for the vertical direction of the plot (related to variance of coordinates)
`yaxt`	axis type for the vertical direction of the plot (see `par`)
`box.pos`	if type="boxplot", this is the relative position of the box in the vertical direction: 0 means centered on the axis, -1 aligned below the axis and +1 aligned above the axis
`box.space`	if type="boxplot", size of the box in the vertical direction as a portion of the minimal variance of the coordinates
`col.tree`	color for the horizontal axes
`lty.tree`	line type for the horizontal axes
`lwd.tree`	line width for the horizontal axes
`col.leaf`	color for the vertical conections between an axis and a part (leaf)
`lty.leaf`	line type for the leaves
`lwd.leaf`	line width for the leaves
`add`	should a new plot be triggered, or is the material to be added to an existing CoDa-dendrogram?
`border`	the color for drawing the rectangles
`type`	what to represent? one of "boxplot","density","histogram","lines","nothing" or "points", or an univocal abbreviation

Details

The object *and an isometric basis* are represented in a CoDa-dendrogram, as defined by Egozcue and Pawlowsky-Glahn (2005). This is a representation of the following elements:

a a hierarchical partition (which can be specified either through an ilrBase matrix (see `ilrBase`), a merging tree structure (see `hclust`) or a signary matrix (see `gsi.merge2signary`))

b the sample mean of each coordinate of the ilr basis associated to that partition

c the sample variance of each coordinate of the ilr basis associated to that partition

d optionally (potentially!), any graphical representation of each coordinate, as long as this representation is suitable for a univariate data set (box-plot, histogram, dispersion and kernel density are programmed or intended to, but any other may be added with little work).

Each coordinate is represented in a horizontal axis, which limits correspond to the values given in the parameter `range`. The vertical bar going up from each one of these coordinate axes represent the variance of that specific coordinate, and the contact point the coordinate mean. Note that to be able to represent an initial dendrogram, the first call to this function must be given a full data set, as means and variances must be computed. This information is afterwards stored in a global list, to add any sort of new material to all coordinates.
The default option is `type="boxplot"`, which produces a box-plot for each coordinate, customizable using `box.pos` and `box.space`, as well as typical `par` parameters (col, border, lty, lwd, etc.).

To obtain only the first three aspects, the function must be called with `type="lines"`. As extensions, one might represent a single datum/few data (e.g., a mean or a random subsample of the data set) calling the function with `add=TRUE` and `type="points"`. Other options (calling functions `histogram` or `density`, and admitting their parameters) will be also soon available.
Note that the original coda-dendrogram as defined by Egozcue and Pawlowsky-Glahn (2005) works with acomp objects and ilr bases. Functionality is extended to rcomp objects using calls to `idt`.

Author(s)

Raimon Tolosana-Delgado, K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

Egozcue J.J., V. Pawlowsky-Glahn, G. Mateu-Figueras and C. Barcel'o-Vidal (2003) Isometric logratio transformations for compositional data analysis. *Mathematical Geology*, **35**(3) 279-300

Egozcue, J.J. and V. Pawlowsky-Glahn (2005). CoDa-Dendrogram: a new exploratory tool. In: Mateu-Figueras, G. and Barcel\'o-Vidal, C. (Eds.) *Proceedings of the 2nd International Workshop on Compositional Data Analysis*, Universitat de Girona, ISBN 84-8458-222-1, https://ima.udg.edu/Activitats/CoDaWork05/

See Also

`ilrBase`,`balanceBase`, `rcomp`, `acomp`,

Examples

```
# first example: take the data set from the example, select only
# compositional parts
data(Hydrochem)
x = acomp(Hydrochem[,-c(1:5)])
gr = Hydrochem[,4] # river groups (useful afterwards)
# use an ilr basis coming from a clustering of parts
dd = dist(t(clr(x)))
hc1 = hclust(dd,method="ward.D")
plot(hc1)
mergetree=hc1$merge
CoDaDendrogram(X=acomp(x),mergetree=mergetree,col="red",range=c(-8,8),box.space=1)
# add the mean of each river
color=c("green3","red","blue","darkviolet")
aux = sapply(split(x,gr),mean)
aux
CoDaDendrogram(X=acomp(t(aux)),add=TRUE,col=color,type="points",pch=4)

# second example: box-plots by rivers (filled)
CoDaDendrogram(X=acomp(x),mergetree=mergetree,col="black",range=c(-8,8),type="l")
xsplit = split(x,gr)
for(i in 1:4){
 CoDaDendrogram(X=xsplit[[i]],col=color[i],type="box",box.pos=i-2.5,box.space=0.5,add=TRUE)
}
```

```
# third example: fewer parts, partition defined by a signary, and empty box-plots
x = acomp(Hydrochem[,c("Na","K","Mg","Ca","Sr","Ba","NH4")])
signary = t(matrix(  c(1,   1,   1,  1,   1,   1,  -1,
                       1,   1,  -1, -1,  -1,  -1,   0,
                       1,  -1,   0,  0,   0,   0,   0,
                       0,   0,  -1,  1,  -1,  -1,   0,
                       0,   0,   1,  0,  -1,   1,   0,
                       0,   0,   1,  0,   0,  -1,   0),ncol=7,nrow=6,byrow=TRUE))

CoDaDendrogram(X=acomp(x),signary=signary,col="black",range=c(-8,8),type="l")
xsplit = split(x,gr)
for(i in 1:4){
  CoDaDendrogram(X=acomp(xsplit[[i]]),border=color[i],
       type="box",box.pos=i-2.5,box.space=1.5,add=TRUE)
  CoDaDendrogram(X=acomp(xsplit[[i]]),col=color[i],
       type="line",add=TRUE)
}
```

---

coloredBiplot *A biplot providing somewhat easier access to details of the plot.*

Description

This function generates a simple biplot out of various sources and allows to give color and symbol to the x-objects individually.

Usage

```
## Default S3 method:
coloredBiplot(x, y, var.axes = TRUE, col,
         cex = rep(par("cex"), 2), xlabs = NULL, ylabs = NULL, expand=1,
         xlim = NULL, ylim = NULL, arrow.len = 0.1, main = NULL, sub = NULL,
         xlab = NULL, ylab = NULL, xlabs.col = NULL, xlabs.bg = NULL,
         xlabs.pc=NULL, ...)
## S3 method for class 'princomp'
coloredBiplot(x, choices = 1:2, scale = 1,
         pc.biplot=FALSE, ...)
## S3 method for class 'prcomp'
coloredBiplot(x, choices = 1:2, scale = 1,
         pc.biplot=FALSE, ...)
```

Arguments

x	a representation of the the co-information to be plotted, given by a result of princomp or prcomp; or the first set of coordinates to be plotted

y	optional, the second set of coordinates to be potted
var.axes	if 'TRUE' the second set of points have arrows representing them as (unscaled) axes
col	one color (to be used for the y set) or a vector of two colors (to be used for x and y sets respectively, if xlabs.col is NULL)
cex	the usual cex parameter for plotting; can be a length-2 vector to format differently x and y labels/symbols
xlabs	names to write for the points of the first set
ylabs	names to write for the points of the second set
expand	expansion factor to apply when plotting the second set of points relative to the first. This can be used to tweak the scaling of the two sets to a physically comparable scale
xlim	horizontal axis limits
ylim	vertical axis limits
arrow.len	length of the arrow heads on the axes plotted if 'var.axes' is true. The arrow head can be suppressed by 'arrow.len=0'
main	main title
sub	subtitle
xlab	horizontal axis title
ylab	vertical axis title
xlabs.col	the color(s) to draw the points of the first set, if xlabs is null
xlabs.bg	the filling color(s) to draw the points of the first set, if xlabs is null and xlabs.pc is between 21 and 25.
xlabs.pc	the plotting character(s) for the first set, if xlabs is null
scale	the way to distribute the singular values on the right or left singular vectors for princomp and prcomp objects (see biplot)
choices	the components to be plotted (see biplot)
pc.biplot	should be scaled by sqrt(nrow(X))? (see biplot)
...	further parameters for plot

Details

The functions is provided for convenience.

Value

The function is called only for the side effect of plotting. It is a modification of the standard R routine 'biplot'.

Author(s)

Raimon Tolosana-Delgado, K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

biplot, plot.acomp

Examples

```
data(SimulatedAmounts)
coloredBiplot(x=princomp(acomp(sa.outliers5)),pc.biplot=FALSE,
          xlabs.pc=c(1,2,3), xlabs.col=2:4, col="black")
```

---

colorsForOutliers *Create a color/char palette or for groups of outliers*

Description

Conveniance Functions to generate meaningfull color palettes for factors representing different types of outliers.

Usage

```
colorsForOutliers1(outfac, family=rainbow,
                        extreme="cyan",outlier="red",ok="gray40",unknown="blue")
colorsForOutliers2(outfac,use=whichBits(gsi.orSum(levels(outfac))),
                         codes=c(2^outer(c(24,16,8),1:7,"-")),ok="yellow")
pchForOutliers1(outfac,ok='.',outlier='\004',extreme='\003',unknown='\004',...,
 other=c('\001','\002','\026','\027','\010','\011','\012','\013','\014','\015',
  '\016',strsplit("abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ","")[[1]])
   )
```

Arguments

outfac	a factor given by an OutlierClassifier (e.g. OutlierClassifier1). colorsForOutliers1 is used for the types "best","type","outlier","grade". colorsForOutliers2 is used for type all.
family	a function generating a color palette from a numer of colors requested.
extreme	The color/char for extrem but not definitivly outlying observations.
outlier	The color/char for detected outliers.
unknown	The color/char for observation with unclear classification.
other	The character codes for other outlier classes.
ok	The color/char for nonoutlying usual observations.
use	a numerical vector giving the indices of the bits of the output to be represented. The sequence of the bits determins how each bit is represented.
codes	The color influences to be used for each bit.
...	further codings for other factorlevels

Details

This functions are provided for coveniance to quickly generate a palette of reasonable colors or plotting chars for groups of outliers classified by OutlierClassifier1.

Value

a character vector of colors or a numeric vector of plot chars.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

OutlierClassifier1

Examples

```
## Not run:
data(SimulatedAmounts)
data5 <- acomp(sa.outliers5)
olc <- OutlierClassifier1(data5)
plot(data5,col=colorsForOutliers1(olc)[olc])
olc <- OutlierClassifier1(data5,type="all")
plot(data5,col=colorsForOutliers2(olc)[olc])

## End(Not run)
```

CompLinModCoReg *Compositional Linear Model of Coregionalisation*

Description

Creates a Variogram model according to the linear model of spatial corregionalisation for a compositional geostatistical analysis.

Usage

```
CompLinModCoReg(formula,comp,D=ncol(comp),envir=environment(formula))
```

Arguments

formula	A formula without left side providing a formal description of a variogram model.
comp	a compositional dataset, needed to provide the frame size
D	The dimension of the multivariate dataset
envir	The enviroment in which formula should be interpreted.

Details

The linear model of coregionalisation uses the fact that sums of valid variogram models are valid variograms, and that scalar variograms multiplied with a positive definite matrix are valid variograms for vector-valued random functions.

This command computes such a variogram function from a formal description, via a formula without left-hand side. The right-hand side of the formula is a sum. Each summand is either a product of a matrix description and a scalar variogram description or only a scalar variogram description. Scalar variogram descriptions are either formal function calls to

`sph(range)` for spherical variogram with range `range`

`exp(range)` for an exponential variogram with effective range `range`

`gauss(range)` for a Gaussian variogram with effective range `range`

`gauss(range)` for a cardinal sine variogram with range parameter `range`

`pow(range)` for an power variogram with range parameter `range`

`lin(unit)` linear variogram 1 at `unit`.

`nugget()` for adding a nuggeteffect.

Alternatively it can be any expression, which will be evaluated in envir and should depende on a dataset of distantce vectrs `h`. An effective range is that distance at which one reaches the sill (for spherical) of 95% of its values (for all other models). Parametric ranges are given for those models that do not have an effective range formula.
The matrix description always comes first. It can be `R1` for a rank 1 matrix; `PSD` for a Positive Semidefinite matrix; \(S\) for a scalar Sill factor to be multiplied with the identity matrix; or any other construct evaluating to a matrix, like e.g. a function of some parameters with default values, that if called is evaluated to a positive semidefinite matrix. `R1` and `PSD` can also be written as calls providing a vector or respectively a matrix providing the parameter.
The variogram is created with default parameter values. The parameters can later be modified by modifiying the default parameter with assignments like `formals(vg)$sPSD1 = parameterPosdefMat(4*diag(5))`.
We would anyway expect you to fit the model to the data by a command like `fit.lmc(logratioVariogram(...),CompLinMo`

Value

A variogram function, with the extra class "`CompLinModCoReg`".

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

What to cite??

See Also

`vgram2lrvgram`, `vgmFit2lrv`

Examples

```
## Not run:
data(juraset)
X <- with(juraset,cbind(X,Y))
comp <- acomp(juraset,c("Cd","Cu","Pb","Co","Cr"))
CompLinModCoReg(~nugget()+sph(0.5)+R1*exp(0.7),comp)
CompLinModCoReg(~nugget()+R1*sph(0.5)+R1*exp(0.7)+(0.3*diag(5))*gauss(0.3),comp)
CompLinModCoReg(~nugget()+R1*sph(0.5)+R1(c(1,2,3,4,5))*exp(0.7),comp)

## End(Not run)
```

---

compOKriging *Compositional Ordinary Kriging*

Description

Geostatistical prediction for compositional data with missing values.

Usage

```
compOKriging(comp,X,Xnew,vg,err=FALSE)
```

Arguments

comp	an acomp compositional dataset
X	A dataset of locations
Xnew	The locations, where a geostatistical prediction should be computed.
vg	A compositional variogram function.
err	boolean: If true kriging errors are computed additionally. A bug was found here; the argument is currently disabled.

Details

The function performes multivariate ordinary kriging of compositions based on transformes addapted to the missings in every case. The variogram is assumed to be a clr variogram.

Value

A list of class "logratioVariogram"

X	The new locations as given by Xnew
Z	The predicted values as acomp compositions.
err	A bug has been found here. This output is currently disabled (An ncol(Z)xDxD array with the clr kriging errors.)

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

Pawlowsky-Glahn, Vera and Olea, Ricardo A. (2004) Geostatistical Analysis of Compositional Data, Oxford University Press, Studies in Mathematical Geology

Tolosana (2008) ...

Tolosana, van den Boogaart, Pawlowsky-Glahn (2009) Estimating and modeling variograms of compositional data with occasional missing variables in R, StatGis09

See Also

vgram2lrvgram, CompLinModCoReg, vgmFit

Examples

```
## Not run:
# Load data
data(juraset)
X <- with(juraset,cbind(X,Y))
comp <- acomp(juraset,c("Cd","Cu","Pb","Co","Cr"))
lrv <- logratioVariogram(comp,X,maxdist=1,nbins=10)
plot(lrv)

# Fit a variogram model
vgModel <- CompLinModCoReg(~nugget()+sph(0.5)+R1*exp(0.7),comp)
fit <- vgmFit2lrv(lrv,vgModel)
fit
plot(lrv,lrvg=vgram2lrvgram(fit$vg))

# Define A grid
x <- (0:10/10)*6
y <- (0:10/10)*6
Xnew <- cbind(rep(x,length(y)),rep(y,each=length(x)))

# Kriging
erg <- compOKriging(comp,X,Xnew,fit$vg,err=FALSE)
par(mar=c(0,0,1,0))
pairwisePlot(erg$Z,panel=function(a,b,xlab,ylab) {image(x,y,
                    structure(log(a/b),dim=c(length(x),length(y))),
                    main=paste("log(",xlab,"/",ylab,")",sep=""));points(X,pch=".")})


# Check interpolation properties
ergR <- compOKriging(comp,X,X,fit$vg,err=FALSE)
pairwisePlot(ilr(comp),ilr(ergR$Z))
ergR <- compOKriging(comp,X,X+1E-7,fit$vg,err=FALSE)
pairwisePlot(ilr(comp),ilr(ergR$Z))
ergR <- compOKriging(comp,X,X[rev(1:31),],fit$vg,err=FALSE)
pairwisePlot(ilr(comp)[rev(1:31),],ilr(ergR$Z))
```

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

`compositional-class` *Class* `"compositional"`

Description

Abstract class containing all compositional classes with (at least, partly) a closed geometry: `acomp`, `rcomp` and `ccomp`

Objects from the Class

A virtual Class: No objects may be created from it.

Methods

No methods defined with class "compositional" in the signature.

Author(s)

Raimon Tolosana-Delgado

See Also

`amounts-class` for classes with total information

Examples

```
showClass("compositional")
```

`ConfRadius` *Helper to compute confidence ellipsoids*

Description

Computes the quantile of the Mahalanobis distance needed to draw confidence ellipsoids.

Usage

```
ConfRadius(model,prob=1-alpha,alpha)
```

Arguments

`model`	A multivariate linear model
`prob`	The confidence probability
`alpha`	The alpha error allowed, i.e. the complement of the confidence probability

Details

Calculates the radius to be used in confidence ellipses for the parameters based on the Hottelings T^2 distribution.

Value

a scalar

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

lm, mvar, AIC

Examples

```
data(SimulatedAmounts)
model <- lm(ilr(sa.groups)~sa.groups.area)
cf = coef(model)
plot(ilrInv(cf, x=sa.groups))
for(i in 1:nrow(cf)){
  vr = vcovAcomp(model)[,,i,i]
  vr = ilrvar2clr(vr)
  ellipses(ilrInv(cf[i,]), vr, r=ConfRadius(model, alpha=0.05) )
 }
```

---

cor.acomp *Correlations of amounts and compositions*

Description

Computes the correlation matrix in the various approaches of compositional and amount data analysis.

Usage

```
          cor(x,y=NULL,...)
          ## Default S3 method:
cor(x, y=NULL, use="everything",
                        method=c("pearson", "kendall", "spearman"),...)
          ## S3 method for class 'acomp'
cor(x,y=NULL,...,robust=getOption("robust"))
          ## S3 method for class 'rcomp'
cor(x,y=NULL,...,robust=getOption("robust"))
          ## S3 method for class 'aplus'
cor(x,y=NULL,...,robust=getOption("robust"))
```

```
          ## S3 method for class 'rplus'
cor(x,y=NULL,...,robust=getOption("robust"))
          ## S3 method for class 'rmult'
cor(x,y=NULL,...,robust=getOption("robust"))
```

Arguments

x	a data set, eventually of amounts or compositions
y	a second data set, eventually of amounts or compositions
use	see cor
method	see cor
...	further arguments to cor e.g. use
robust	A description of a robust estimator. FALSE for the classical estimators. See mean.acomp for further details.

Details

The correlation matrix does not make much sense for compositions.

In R versions older than v2.0.0, cor was defined in package "base" instead of in "stats". This might produce some misfunction.

Value

The correlation matrix.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

var.acomp

Examples

```
data(SimulatedAmounts)
meanCol(sa.lognormals)
cor(acomp(sa.lognormals5[,1:3]),acomp(sa.lognormals5[,4:5]))
cor(rcomp(sa.lognormals5[,1:3]),rcomp(sa.lognormals5[,4:5]))
cor(aplus(sa.lognormals5[,1:3]),aplus(sa.lognormals5[,4:5]))
cor(rplus(sa.lognormals5[,1:3]),rplus(sa.lognormals5[,4:5]))
cor(acomp(sa.lognormals5[,1:3]),aplus(sa.lognormals5[,4:5]))
```

Coxite *Compositions, depths and porosities of 25 specimens of coxite*

Description

A mineral compositions of 25 rock specimens of coxite type. Each composition consists of the percentage by weight of five minerals, albite, blandite, cornite, daubite, endite, the depth of location, and porosity.

Usage

```
data(Coxite)
```

Details

A mineral compositions of 25 rock specimens of coxite type. Each composition consists of the percentage by weight of five minerals, albite, blandite, cornite, daubite, endite,the recorded depth of location of each specimen, and porosity. Porosity is the percentage of void space that the specimen contains. We abbreviate the minerals names to A, B, C, D, E.

All row percentage sums to 100.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name BOXITE.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison: The Statistical Analysis of Compositional Data, 1986, Data 4, pp4.

cpt *Centered planar transform*

Description

Compute the centered planar transform of a (dataset of) compositions and its inverse.

Usage

```
cpt( x,... )
cptInv( z,...,orig=gsi.orig(z) )
```

Arguments

x	a composition or a data.matrix of compositions, not necessarily closed
z	the cpt-transform of a composition or a data matrix of cpt-transforms of compositions. It is checked that the z sum up to 0.
...	generic arguments. not used.
orig	a compositional object which should be mimicked by the inverse transformation. It is especially used to reconstruct the names of the parts.

Details

The cpt-transform maps a composition in the D-part real-simplex isometrically to a D-1 dimensional euclidian vector space, identified with a plane parallel to the simplex but passing through the origin. However the transformation is not injective and does not even reach the whole plane. Thus resulting covariance matrices are always singular.

The data can then be analysed in this transformed space by all classical multivariate analysis tools not relying on a full rank of the covariance matrix. See ipt and apt for alternatives. The interpretation of the results is relatively easy since the relation of each transformed component to the original parts is preserved.

The centered planar transform is given by

$$cpt(x)_i := clo(x)_i - \frac{1}{D}$$

Value

cpt gives the centered planar transform, cptInv gives closed compositions with the given cpt-transforms.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

van den Boogaart, K.G. and R. Tolosana-Delgado (2008) "compositions": a unified R package to analyze Compositional Data, *Computers & Geosciences*, 34 (4), pages 320-338, doi: 10.1016/j.cageo.2006.11.017.

See Also

clr,apt,ipt

Examples

```
(tmp <- cpt(c(1,2,3)))
cptInv(tmp)
cptInv(tmp) - clo(c(1,2,3)) # 0
data(Hydrochem)
cdata <- Hydrochem[,6:19]
pairs(cpt(cdata),pch=".")
```

`DiagnosticProb` *Diagnostic probabilities*

Description

Data record the probabilities assigned by subjective diagnostic of 15 clinicians and 15 statisticians.

Usage

```
data(DiagnosticProb)
```

Details

The data consist of 30 cases: 15 diagnostics probabilities assigned by clinicians, 15 diagnostics probabilities assigned by statisticians, and 4 variables: probabilities A, B, and C, and type i.e. 1 for clinicians, 2 for statisticians.

In the study of subjective performance in inferential task the subject is faced with the finite set of mutually exclusive and exhaustive hypothesis, and the basis of specific information presented to him/her is required to divide the available unit of probability among these probabilities. In this study the task is presented as a problem of differential diagnosis of three mutually exclusive and exhaustive diseases of students, known under the generic title of 'newmath syndrome',

A - algebritis,
B - bilateral paralexia,
C - calculus deficiency.

The subject, playing the role of diagnostician, is informed that the three diseases types are equally common and is shown the results of 10 diagnostic tests on 60 previous cases of known diagnosis, 20 of each type. The subject is then shown the results of the 10 tests for a new undiagnosed cases and asked to assign diagnostic probabilities to the three possible disease types.

Data record the subjective assessments of 15 clinicians and 15 statisticians for the same case. For this case the objective diagnosis probabilities are known to be $(.08, .05, .87).$

All row probabilities sum to 1, except for some rounding errors.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name DIAGPROB.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison: The Statistical Analysis of Compositional Data, 1986, Data 17, pp20.

`dist` *Distances in variouse approaches*

Description

Calculates a distance matrix from a data set.

Usage

```
dist(x,...)
## Default S3 method:
dist(x,...)
```

Arguments

`x`	a dataset
`...`	further arguments to `dist`

Details

The distance is computed based on `cdt`

Value

a distance matrix

Author(s)

K.Gerald v.d. Boogaart `http://www.stat.boogaart.de`

See Also

`aplus`

Examples

```
data(SimulatedAmounts)
phc <- function(d) { plot(hclust(d))}
phc(dist(iris[,1:4]))
phc(dist(acomp(sa.lognormals),method="manhattan"))
phc(dist(rcomp(sa.lognormals)))
phc(dist(aplus(sa.lognormals)))
phc(dist(rplus(sa.lognormals)))
```

`ellipses` *Draw ellipses*

Description

Draws ellipses from a mean and a variance into a plot.

Usage

```
  ## S3 method for class 'acomp'
ellipses(mean,var,r=1,...,steps=72,
                                        thinRatio=NULL,aspanel=FALSE)
  ## S3 method for class 'rcomp'
ellipses(mean,var,r=1,...,steps=72,
                                        thinRatio=NULL,aspanel=FALSE)
  ## S3 method for class 'aplus'
ellipses(mean,var,r=1,...,steps=72,thinRatio=NULL)
  ## S3 method for class 'rplus'
ellipses(mean,var,r=1,...,steps=72,thinRatio=NULL)
  ## S3 method for class 'rmult'
ellipses(mean,var,r=1,...,steps=72,thinRatio=NULL)
```

Arguments

`mean`	a compositional dataset or value of means or midpoints of the ellipses
`var`	a variance matrix or a set of variance matrices given by `var[i,,]` (multiple covariance matrices are not consitently implemented as of today). The principal axis of the variance give the axis of the ellipses, whereas the square-root of the eigenvalues times r give the half-diameters of the ellipse.
`r`	a scaling of the half-diameters
`...`	further graphical parameters
`steps`	the number of discretisation points to draw the ellipses.

`thinRatio` The ellipse function now be default plots the whole ellipsiod by giving its principle circumferences. However this is not reasonable for the thinner directions. If a direction other than the first two eigendirections has an eigenvalue not bigger than thinRatio*rmax it is not plotted. Thus thinRatio=1 reinstantiates the old behavior of the function. Later thinratio=NULL will become the default, in which case the projection of the ellipse is plotted. However this is not implemented yet.

`aspanel` Is the function called as slave to draw in a panel of a gsi.pairs plot, or as a user function setting up the plots.

Details

The ellipsoid/ellipse drawn is given by the solutions of

$$(x - mean)^t var^{-1}(x - mean) = r^2$$

in the respective geometry of the parameter space. Note that these ellipses can be added to panel plots (by means of orthogonal projections in the corresponding geometry).

There are actually three possibilities of drawing a a hyperdimensional ellipsoid or ellipse and non of them is perfect.

thinRatio=1.1 This works like, what was implemented in the older versions of compositons, but never correctly documented. It draws an ellipse with main axes given by the two largest Eigendirections of the `var`-Matrix given.

thinRatio=0 Draws all the ellipses given by every pair of eigendirections. In this way we get a visual impression of the high dimensional ellipsoid represend by the variance matrix. However the plots gets fastly cluttered in dimensions, when D>4. A 0<thinRatio<1 can avoid using eigendirection with small extend (i.e. smaller than thinRatio*largest Eigenvalue.

thinRatio=NULL Draws in each Panel a two dimensional ellipse representing the marginal variance in the projection of the plot, if var was to be interpreted as a variance matrix. This can be seen as some sort of projection of the high dimensional ellipsoid, but is not necessarily its visual outline.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

`plot.acomp`,

Examples

```
data(SimulatedAmounts)

plot(acomp(sa.lognormals))
tt<-acomp(sa.lognormals); ellipses(mean(tt),var(tt),r=2,col="red")
tt<-rcomp(sa.lognormals); ellipses(mean(tt),var(tt),r=2,col="blue")
```

```
plot(aplus(sa.lognormals[,1:2]))
tt<-aplus(sa.lognormals[,1:2]); ellipses(mean(tt),var(tt),r=2,col="red")
tt<-rplus(sa.lognormals[,1:2]); ellipses(mean(tt),var(tt),r=2,col="blue")

plot(rplus(sa.lognormals[,1:2]))
tt<-aplus(sa.lognormals[,1:2]); ellipses(mean(tt),var(tt),r=2,col="red")
tt<-rplus(sa.lognormals[,1:2]); ellipses(mean(tt),var(tt),r=2,col="blue")
tt<-rmult(sa.lognormals[,1:2]); ellipses(mean(tt),var(tt),r=2,col="green")
```

---

endmemberCoordinates *Recast amounts as mixtures of end-members*

Description

Computes the convex combination of amounts as mixtures of endmembers to explain X as well as possible.

Usage

```
    endmemberCoordinates(X,...)
    endmemberCoordinatesInv(K,endmembers,...)
    ## Default S3 method:
endmemberCoordinates(X,
                        endmembers=diag(gsi.getD(X)), ...)
    ## S3 method for class 'acomp'
endmemberCoordinates(X,
                    endmembers=clrInv(diag(gsi.getD(X))),...)
    ## S3 method for class 'aplus'
endmemberCoordinates(X,endmembers,...)
    ## S3 method for class 'rplus'
endmemberCoordinates(X,endmembers,...)
    ## S3 method for class 'rmult'
endmemberCoordinatesInv(K,endmembers,...)
    ## S3 method for class 'acomp'
endmemberCoordinatesInv(K,endmembers,...)
    ## S3 method for class 'rcomp'
endmemberCoordinatesInv(K,endmembers,...)
    ## S3 method for class 'aplus'
endmemberCoordinatesInv(K,endmembers,...)
    ## S3 method for class 'rplus'
endmemberCoordinatesInv(K,endmembers,...)
```

Arguments

X	a data set of amounts or compositions, to be represented in as convex combination of the endmembers in the given geometry

K	weights of the endmembers in the convex combination
endmembers	a dataset of compositions of the same class as X. The number of endmembers given must not exceed the dimension of the space plus one.
...	currently unused

Details

The convex combination is performed in the respective geometry. This means that, for rcomp objects, positivity of the result is only guaranteed with endmembers corresponding to extremal individuals of the sample, or completely outside its hull. Note also that, in acomp geometry, the endmembers must necessarily be outside the hull.
The main idea behind this functions is that the composition actually observed came from a convex combination of some extremal compositions, specified by endmembers. Up to now, this is considered as meaningful only in rplus geometry, and under some special circumstances, in rcomp geometry. It is not meaningful in terms of mass conservation in acomp and aplus geometries, because these geometries do not preserve mass: whether such an operation has an interpretation is still a matter of debate. In rcomp geometry, the convex combination is dependent on the units of measurements, and will be completely different for volume and mass %. Even more, it is valid only if the whole composition is observed (!).

Value

The endmemberCoordinates functions give a rmult data set with the weights (a.k.a. barycentric coordinates) allowing to build X as good as possible as a convex combination (a mixture) from endmembers. The result is of class rmult because there is no guarantee that the resulting weights are positive (although they sum up to one).
The endmemberCoordinatesInv functions reconstruct the convex combination from the weights K and the given endmembers. The class of endmembers determines the geometry chosen and the class of the result.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de, Raimon Tolosana-Delgado

References

Shurtz, Robert F., 2003. Compositional geometry and mass conservation. Mathematical Geology 35 (8), 972–937.

Examples

```
data(SimulatedAmounts)
ep <- aplus(rbind(c(2,1,2),c(2,2,1),c(1,2,2)))
# mix the endmembers in "ep" with weights given by "sa.lognormals"
dat <- endmemberCoordinatesInv(acomp(sa.lognormals),acomp(ep))
par(mfrow=c(1,2))
plot(dat)
  plot(acomp(ep),add=TRUE,col="red",pch=19)
# compute the barycentric coordinates of the mixture in the "end-member simplex"
plot( acomp(endmemberCoordinates(dat,acomp(ep))))
```

```
dat <- endmemberCoordinatesInv(rcomp(sa.lognormals),rcomp(ep))
plot(dat)
plot( rcomp(endmemberCoordinates(dat,rcomp(ep))))

dat <- endmemberCoordinatesInv(aplus(sa.lognormals),aplus(ep))
plot(dat)
plot( endmemberCoordinates(dat,aplus(ep)))

dat <- endmemberCoordinatesInv(rplus(sa.lognormals),rplus(ep))
plot(dat)
plot(endmemberCoordinates(rplus(dat),rplus(ep)))
```

`Firework` *Firework mixtures*

Description

Data show two measured properties, brilliance and vorticity, of 81 girandoles composed of different mixtures of five ingredients: a – e. Of these ingredients, a and b are the primary light-producting, c is principal propellant, and d and e are binding agents for c.

Usage

```
data(Firework)
```

Details

The data consist of 81 cases and 7 variables: ingredients a, b, c, d, and e, and the two measured properties *brilliance* and *vorticity*. The 81 different mixtures form a special experiment design. First the 81 possible quadruples formed from the three values -1, 0, 1 were arranged in ascending order. Then for each such quadruple z, the corresponding mixture x(z)=(a,b,c,d,e)=alrInv(z) is computed. Thus the No. 4 girandole corresponds to z=(-1,-1,0,-1) and so is composed of a mixture x=(.12,.12,.32,.12,.32) of the five ingredients. All 5-part mixtures sum up to one.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name YATQUAD.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison: The Statistical Analysis of Compositional Data, 1986, Data 13, pp17.

`fitdirichlet` *Fitting a Dirichlet distribution*

Description

Fits a Dirichtlet Distribution to a dataset by maximum likelihood.

Usage

```
fitDirichlet(x,elog=mean(ult(x)),alpha0=rep(1,length(elog)),maxIter=20,n=nrow(x))
```

Arguments

`x`	a dataset of compositions (acomp)
`elog`	the expected log can provided instead of the dataset itself.
`alpha0`	the start value for alpha parameter in the iteration
`maxIter`	The maximum number of iterations in the Fischer scoring method.
`n`	the number of datapoints used to estimate elog

Details

The fitting is done using a modified version of the Fisher-Scoring method using analytiscal expressions for log mean and log variance. The modification is introducted to prevent the algorithm from leaving the admissible parameter set. It reduced the stepsize to at most have of distance to the limit of the admissible parameter set.

Value

`alpha`	the estimated parameter
`loglikelihood`	the likelihood
`df`	The dimension of the dataset minus the dimension of the parameter

Missing Policy

Up to now the fitting can not handle missings.

Author(s)

K.Gerald v.d. Boogaart `http://www.stat.boogaart.de`

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

See Also

rDirichlet, acompDirichletGOF.test, runif.acomp, rnorm.acomp,

Examples

```
x <- rDirichlet.acomp(100,c(1,2,3,4))
fitDirichlet(x)
```

---

fitSameMeanDifferentVarianceModel

Fit Same Mean Different Variance Model

Description

Fits a model of the same mean, but different variances model to a set of several multivariate normal groups by maximum likelihood.

Usage

```
fitSameMeanDifferentVarianceModel(x)
```

Arguments

x	list of rmult type datasets

Details

The function tries to fit a normal model with different variances but the same mean between different groups.

Value

mean	the estimated mean
vars	a list of estimated variance-covariance matrices
N	a vector containing the sizes of the groups

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

See Also

acompNormalLocation.test

Examples

```
fitSameMeanDifferentVarianceModel
```

---

gausstest *Classical Gauss Test*

Description

One and two sample Gauss test for equal mean of normal random variates with known variance.

Usage

```
Gauss.test(x,y=NULL,mean=0,sd=1,alternative = c("two.sided", "less", "greater"))
```

Arguments

x	a numeric vector providing the first dataset
y	optional second dataset
mean	the mean to compare with
sd	the known standard deviation
alternative	the alternative to be used in the test

Details

The Gauss test is in every Text-Book, but not in R, because it is nearly never used. However it is included here for educational purposes.

Value

A classical "htest" object

data.name	The name of the dataset as specified
method	a name for the test used
parameter	the mean and variance provided to the test
alternative	an empty string
p.value	The p.value computed for this test

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

t.test

Examples

```
x <- rnorm(100)
y <- rnorm(100)
Gauss.test(x,y)
```

---

geometricmean *The geometric mean*

Description

Computes the geometric mean.

Usage

```
geometricmean(x,...)
geometricmeanRow(x,...)
geometricmeanCol(x,...)
gsi.geometricmean(x,...)
gsi.geometricmeanRow(x,...)
gsi.geometricmeanCol(x,...)
```

Arguments

x	a numeric vector or matrix of data
...	further arguments to compute the mean

Details

The geometric mean is defined as:

$$geometricmean(x) := \left(\prod_{i=1}^{n} x_i\right)^{1/n}$$

The geometric mean is actually computed by `exp(mean(log(c(unclass(x))),...))`.

Value

The geometric means of x as a whole (geometricmean), its rows (geometricmeanRow) or its columns (geometricmeanCol).

Missing Policy

The the first three functions take the geometric mean of all non-missing values. This is because they should yield a result in term of data analysis.

Contrarily, the gsi.* functions inherit the arithmetic IEEE policy of R through `exp(mean(log(c(unclass(x))),...))`. Thus, NA codes a not available i.e. not measured, NaN codes a below detection limit, and 0.0 codes a structural zero. If any of the elements involved is 0, NA or NaN the result is of the same type. Here 0 takes precedence over NA, and NA takes precedence over NaN. For example, if a structural 0 appears, the geometric mean is 0 regardless of the presence of NaN's or NA's in the rest. Values below detection limit become NaN's if they are coded as negative values.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

mean.rplus

Examples

```
geometricmean(1:10)
geometricmean(c(1,0,NA,NaN))  # 0
X <- matrix(c(1,NA,NaN,0,1,2,3,4),nrow=4)
X
geometricmeanRow(X)
geometricmeanCol(X)
```

---

getdetectionlimit *Gets the detection limit stored in the data set*

Description

The detection limit of those values below-detection-limit are stored as negative values in compositional dataset. This function extracts that information.

Usage

```
getDetectionlimit(x,dl=attr(x,"detectionlimit"))
```

Arguments

x	a data set
dl	a default to replace the information in the dataset

Details

For a proper treatment of truncated data it would be necessary to know the detection limit even for observed data. Unfortunately, there is no clear way to encode this information without annoying the user.

Value

a matrix in the same shape as x, with a positive value (the detection limit) where available, and NA in the other cells.

Author(s)

K.Gerald van den Boogaart

References

Boogaart, K.G. v.d., R. Tolosana-Delgado, M. Bren (2006) Concepts for handling of zeros and missing values in compositional data, in E. Pirard (ed.) (2006)Proceedings of the IAMG'2006 Annual Conference on "Quantitative Geology from multiple sources", September 2006, Liege, Belgium, S07-01, 4pages, ISBN 978-2-9600644-0-7, http://www.stat.boogaart.de/Publications/iamg06_s07_01.pdf

See Also

compositions.missings,zeroreplace

Examples

```
x <- c(2,-0.5,4,3,-0.5,5,BDLvalue,MARvalue,MNARvalue)
getDetectionlimit(x)
```

Glacial *Compositions and total pebble counts of 92 glacial tills*

Description

In a pebble analysis of glacial tills, the total number of pebbles in each of 92 samples was counted and the pebbles were sorted into four categories red sandstone, gray sandstone, crystalline and miscellaneous. The percentages of these four categories and the total pebble counts are recorded.

Usage

```
data(Glacial)
```

Details

Percentages by weight in 92 samples of pebbles of glacial tills sorted into four categories, red sandstone, gray sandstone, crystalline and miscellaneous. The percentages of these four categories and the total pebbles counts are recorded. The glaciologist is interested in describing the pattern of variability of his data and whether the compositions are in any way related to abundance. All rows sum to 100, except for some rounding errors.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name GLACIAL.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison: The Statistical Analysis of Compositional Data, 1986, Data 18, pp21.

---

gof *Compositional Goodness of fit test*

Description

Goodness of fit tests for compositional data.

Usage

```
acompGOF.test(x,...)
acompNormalGOF.test(x,...,method="etest")
## S3 method for class 'formula'
acompGOF.test(formula, data,...,method="etest")
## S3 method for class 'list'
acompGOF.test(x,...,method="etest")
gsi.acompUniformityGOF.test(x,samplesize=nrow(x)*20,R=999)
acompTwoSampleGOF.test(x,y,...,method="etest",data=NULL)
```

Arguments

x	a dataset of compositions (acomp)
y	a dataset of compositions (acomp)
samplesize	number of observations in a reference sample specifying the distribution to compare with. Typically substantially larger than the sample under investigation
R	The number of replicates to compute the distribution of the test statistic
method	Selecting a method to be used. Currently only "etest" for using an energy test is supported.
...	further arguments to the methods
formula	an anova model formula defining groups in the dataset
data	unused

Details

The compositional goodness of fit testing problem is essentially a multivariate goodness of fit test. However there is a lack of standardized multivariate goodness of fit tests in R. Some can be found in the energy-package.

In principle there is only one test behind the Goodness of fit tests provided here, a two sample test with test statistic.

$$\frac{\sum_{ij} k(x_i, y_i)}{\sqrt{\sum_{ij} k(x_i, x_i) \sum_{ij} k(y_i, y_i)}}$$

The idea behind that statistic is to measure the cos of an angle between the distributions in a scalar product given by

$$(X, Y) = E[k(X, Y)] = E[\int K(x - X)K(x - Y)dx]$$

where k and K are Gaussian kernels with different spread. The bandwith is actually the standarddeviation of k.
The other goodness of fit tests against a specific distribution are based on estimating the parameters of the distribution, simulating a large dataset of that distribution and apply the two sample goodness of fit test.

For the moment, this function covers: two-sample tests, uniformity tests and additive logistic normality tests. Dirichlet distribution tests will be included soon.

Value

A classical "htest" object

data.name	The name of the dataset as specified
method	a name for the test used
alternative	an empty string
replicates	a dataset of p-value distributions under the Null-Hypothesis got from nonparametric bootstrap
p.value	The p.value computed for this test

Missing Policy

Up to now the tests can not handle missings.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

See Also

fitDirichlet,rDirichlet, runif.acomp, rnorm.acomp,

Examples

```
## Not run:
x <- runif.acomp(100,4)
y <- runif.acomp(100,4)

erg <- acompTwoSampleGOF.test(x,y)
#continue
erg
unclass(erg)
erg <- acompGOF.test(x,y)


x <- runif.acomp(100,4)
y <- runif.acomp(100,4)
dd <- replicate(1000,acompGOF.test(runif.acomp(100,4),runif.acomp(100,4))$p.value)
hist(dd)

dd <- replicate(1000,acompGOF.test(runif.acomp(20,4),runif.acomp(100,4))$p.value)
hist(dd)
dd <- replicate(1000,acompGOF.test(runif.acomp(10,4),runif.acomp(100,4))$p.value)

hist(dd)
dd <- replicate(1000,acompGOF.test(runif.acomp(10,4),runif.acomp(400,4))$p.value)
hist(dd)
dd <- replicate(1000,acompGOF.test(runif.acomp(400,4),runif.acomp(10,4),bandwidth=4)$p.value)
hist(dd)


dd <- replicate(1000,acompGOF.test(runif.acomp(20,4),runif.acomp(100,4)+acomp(c(1,2,3,1)))$p.value)

hist(dd)

# test uniformity

attach("gsi") # the uniformity test is only available as an internal function
x <- runif.acomp(100,4)
gsi.acompUniformityGOF.test.test(x)

dd <- replicate(1000,gsi.acompUniformityGOF.test.test(runif.acomp(10,4))$p.value)
hist(dd)
detach("gsi")


## End(Not run)
```

groupparts *Group amounts of parts*

Description

Groups parts by amalgamation or balancing of their amounts or proportions.

Usage

```
groupparts(x,...)
## S3 method for class 'acomp'
groupparts(x,...,groups=list(...))
## S3 method for class 'rcomp'
groupparts(x,...,groups=list(...))
## S3 method for class 'aplus'
groupparts(x,...,groups=list(...))
## S3 method for class 'rplus'
groupparts(x,...,groups=list(...))
## S3 method for class 'ccomp'
groupparts(x,...,groups=list(...))
```

Arguments

x	an amount/compositional dataset
...	further parameters to use (actually ignored)
groups	a list of numeric xor character vectors, each giving a group of parts

Details

In the real geometry grouping is done by amalgamation (i.e. adding the parts). In the Aitchison-geometry grouping is done by taking geometric means. The new parts are named by named formal arguments. Not-mentioned parts remain ungrouped.

Value

a new dataset of the same type with each group represented by a single column

Missing Policy

For the real geometries, SZ and BDL are considered as 0, and MAR and MNAR are kept as missing of the same type. For the relative geometries, a BDL is a special kind of MNAR, whereas a SZ is qualitatively different (thus a balance with a SZ has no sense). MAR values transfer their MAR property to the resulting new variable.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de, Raimon Tolosana-Delgado

References

Egozcue, J.J. and V. Pawlowsky-Glahn (2005) Groups of Parts and their Balances in Compositional Data Analysis, Mathematical Geology, in press

See Also

aplus

Examples

```
data(SimulatedAmounts)
plot(groupparts(acomp(sa.lognormals5),A=c(1,2),B=c(3,4),C=5))
plot(groupparts(aplus(sa.lognormals5),B=c(3,4),C=5))
plot(groupparts(rcomp(sa.lognormals5),A=c("Cu","Pb"),B=c(2,5)))
hist(groupparts(rplus(sa.lognormals5),1:5))
```

Hongite *Compositions of 25 specimens of hongite*

Description

A mineral compositions of 25 rock specimens of hongite type. Each composition consists of the percentage by weight of five minerals, albite, blandite, cornite, daubite, endite.

Usage

```
data(Hongite)
```

Details

A mineral compositions of 25 rock specimens of hongite type. Each composition consists of the percentage by weight of five minerals, albite, blandite, cornite, daubite, endite, which we conveniently abbreviate to A, B, C, D, E. All row sums are equal to 100, except for rounding errors.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name HONGITE.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison (1986): The Statistical Analysis of Compositional Data; (Data 1), pp2, 9.

HotellingsTsq *Hotellings T square distribution*

Description

The Hotellings T square distribution is the distribution of the squared Mahalanobis distances with respected to estimated variance covariance matrices.

Usage

```
qHotellingsTsq(p,n,m)
pHotellingsTsq(q,n,m)
```

Arguments

p	a (vector of) probabilities
q	a vector of quantils
n	number of parameters, the p parameter of Hotellings T^2 distribution
m	number of dimensions, the m parameter of the Hotellings T^2 distribution

Details

The Hotellings T^2 with paramter p and m is the distribution empirical squared Mahalanobis distances of a m dimensional vector with respect to a variance covariance matrix estimated based on np degrees of freedom.

Value

qHotellingsT2	a vector of quantils
pHotellingsT2	a vector of probabilities

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

ellipses, ConfRadius, pf

Examples

```
(q <- qHotellingsTsq(seq(0,0.9,by=0.1),3,25))
pHotellingsTsq(q,3,25)
```

HouseholdExp *Household Expenditures*

Description

Household budget survey data on month expenditures of twenty men and twenty women for four commodity groups: housing, foodstuffs, other, and services. Amounts in HK-Dollar are given. There are 40 cases and 5 variables for 4 commodity groups and sex.

Usage

```
data(HouseholdExp)
```

Details

In a sample survey of people living alone in a rented accommodation, twenty men and twenty women were randomly selected and asked to record over a period of one month their expenditures on the following four mutually exclusive and exhaustive commodity groups: Housing, including fuel and lights, Foodstuffs, including alcohol and tobacco, Other goods, including clothing, footwear and durable goods, and Services, including transport and vehicles. Amounts in HK-Dollar are given. There are 40 cases, 20 men and 20 women and 5 variables: 4 for commodity groups Housing, Food, Other, Services and the fifth sex, $+1$ for men, -1 for women. Note that the data has no sum constraint.

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name HEMF.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison (1986): The Statistical Analysis of Compositional Data, (Data 08), pp13.

Hydrochem *Hydrochemical composition data set of Llobregat river basin water (NE Spain)*

Description

Contains a hydrochemical amount/compositional data set obtained from several rivers in the Llobregat river basin, in northeastern Spain.

Usage

```
data(Hydrochem)
```

Format

Data matrix with 485 cases and 19 variables.

Details

This hydrochemical data set contains measurements of 14 components, H, Na, K, Ca, Mg, Sr, Ba, NH_4, Cl, HCO_3, NO_3, SO_4, PO_4, TOC. From them, hydrogen was derived by inverting the relationship between its stable form in water, H_3O^+, and pH. Details can be found in Otero et al. (2005). Each of these parameters is measured approximately once each month during 2 years in 31 stations, placed along the rivers and main tributaries of the Llobregat river, one of the medium rivers in northeastern Spain.

The Llobregat river drains an area of 4948.2 km^2, and it is 156.6 km long, with two main tributaries, Cardener and Anoia. The headwaters of Llobregat and Cardener are in a rather unpolluted area of the Eastern Pyrenees. Mid-waters these rivers flow through a densely populated and industrialized area, where potash mining activity occurs and there are large salt mine tailings stored with no water proofing. There, the main land use is agriculture and stockbreeding. The lower course flows through one of the most densely populated areas of the Mediterranean region (around the city of Barcelona) and the river receives large inputs from industry and urban origin, while intensive agriculture activity is again present in the Llobregat delta. Anoia is quite different. Its headwaters are in an agricultural area, downwaters it flows through an industrialized zone (paper mills, tannery and textile industries), and near the confluence with Llobregat the main land use is agriculture again, mainly vineyards, with a decrease in industry and urban contribution. Given this variety in geological background and human activities, the sample has been splitted in four groups (higher Llobregat course, Cardener, Anoia and lower Llobregat course), which in turn are splitted into main river and tributaries (Otero et al, 2005). Information on these groupings, the sampling locations and sampling time is included in 5 complementary variables.

Author(s)

Raimon Tolosana-Delgado

Source

The dataset is also accessible in Otero et al. (2005), and are here included under the GNU Public Library Licence Version 2 or newer.

References

Otero, N.; R. Tolosana-Delgado, A. Soler, V. Pawlowsky-Glahn and A. Canals (2005). Relative vs. absolute statistical analysis of compositions: A comparative study of surface waters of a Mediterranean river. Water Research, 39(7): 1404-1414. doi: 10.1016/j.watres.2005.01.012.

Tolosana-Delgado, R.; Otero, N.; Pawlowsky-Glahn, V.; Soler, A. (2005). Latent Compositional Factors in The Llobregat River Basin (Spain) Hydrogeochemistry. Mathematical Geology 37(7): 681-702.

Examples

```
data(Hydrochem)
cHydrochem=Hydrochem[, 6:19]
```

```
biplot(princomp(rplus(cHydrochem)))
biplot(princomp(rcomp(cHydrochem)))

biplot(princomp(aplus(cHydrochem)))
biplot(princomp(acomp(cHydrochem)))
```

---

idt *Isometric default transform*

Description

Compute the isometric default transform of a vector (or dataset) of compositions or amounts in the selected class.

Usage

```
          idt(x,...)
          ## Default S3 method:
idt( x,... )
          ## S3 method for class 'acomp'
idt( x ,...)
          ## S3 method for class 'rcomp'
idt( x ,...)
          ## S3 method for class 'aplus'
idt( x ,...)
          ## S3 method for class 'rplus'
idt( x ,...)
          ## S3 method for class 'rmult'
idt( x ,...)
          ## S3 method for class 'ccomp'
idt( x ,...)
          ## S3 method for class 'factor'
idt( x ,...)
          ## S3 method for class 'data.frame'
idt( x ,...)
          idtInv(x,orig=gsi.orig(x),...)
          ## Default S3 method:
idtInv( x ,orig=gsi.orig(x),...)
          ## S3 method for class 'acomp'
idtInv( x ,orig=gsi.orig(x), V=gsi.getV(x),...)
          ## S3 method for class 'rcomp'
idtInv( x ,orig=gsi.orig(x), V=gsi.getV(x),...)
          ## S3 method for class 'aplus'
idtInv( x ,orig=gsi.orig(x),...)
          ## S3 method for class 'rplus'
idtInv( x ,orig=gsi.orig(x),...)
          ## S3 method for class 'ccomp'
```

```
idtInv( x ,orig=gsi.orig(x),...)
          ## S3 method for class 'rmult'
idtInv( x ,orig=gsi.orig(x),...)
          ## S3 method for class 'factor'
idtInv( x ,orig=gsi.orig(x), V=gsi.getV(x),...)
          ## S3 method for class 'data.frame'
idtInv( x , orig=gsi.orig(x), ...)
```

Arguments

x	a classed amount or composition, to be transformed with its isometric default transform, or its inverse; in case of the method for data.frame objects, the function attempts to track information about a previous class (in an attribute origClass, and if found, a transformation is tried with it; for factors, idt expands the factor according to the contrasts represented by V, or vice-versa.)
...	generic arguments past to underlying functions
orig	a compositional object which should be mimicked by the inverse transformation. It is the generic argument. Typically the orig argument is stored as an attribute in x and will be extracted automatically by this function; if this fails, orig can be set equal to the dataset that was transformed in the first place.
V	matrix of (*transposed, inverted*) logcontrasts; together with orig, it defines the back-transformation. Typically the V argument is stored as an attribute in x and will be extracted automatically by this function; if this fails, V must be manually set to the matrix V used in the idt/ilr/ipt calculations. Argument not used in amounts or counts geometries.

Details

The general idea of this package is to analyse the same data with different geometric concepts, in a fashion as similar as possible. For each of the four concepts there exists an isometric transform expressing the geometry in a full-rank euclidean vector space. Such a transformation is computed by idt. For acomp the transform is ilr, for rcomp it is ipt, for aplus it is ilt, and for rplus it is iit. Keep in mind that the transform does not keep the variable names, since there is no guaranteed one-to-one relation between the original parts and each transformed variable.
The inverse idtInv is intended to allow for an "easy" and automatic back-transformation, without intervention of the user. The argument orig (the one determining the behaviour of idtInv as a generic function) tells the function which back-transformation should be applied, and gives the column names of orig to the back-transformed values of x. Therefore, it is very conventient to give the original classed data set used in the analysis as orig.

Value

A corresponding matrix of row-vectors containing the transforms. (Exception: idt.data.frame can return a data.frame if the input has no "origClass"-attribute)

Author(s)

R. Tolosana-Delgado, K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

van den Boogaart, K.G. and R. Tolosana-Delgado (2008) "compositions": a unified R package to analyze Compositional Data, *Computers & Geosciences*, 34 (4), pages 320-338, doi: 10.1016/j.cageo.2006.11.017.

See Also

backtransform, cdt, ilr, ipt, ilt, cdtInv, ilrInv, iptInv, iltInv, iitInv

Examples

```
## Not run:
# the idt is defined by
idt         <- function(x) UseMethod("idt",x)
idt.default <- function(x) x
idt.acomp   <- function(x) ilr(x)
idt.rcomp   <- function(x) ipt(x)
idt.aplus   <- ilt
idt.rplus   <- iit

## End(Not run)
idt(acomp(1:5))
idt(rcomp(1:5))
  data(Hydrochem)
  x = Hydrochem[,c("Na","K","Mg","Ca")]
  y = acomp(x)
  z = idt(y)
  y2 = idtInv(z,y)
  par(mfrow=c(2,2))
  for(i in 1:4){plot(y[,i],y2[,i])}
```

---

iit *Isometric identity transform*

Description

Compute the isometric identity transform of a vector (dataset) of amounts and its inverse.

Usage

```
iit( x ,...)
iitInv( z ,... )
```

Arguments

x	a vector or data matrix of amounts
z	the iit-transform of a vector or data.matrix of iit-transforms of amounts
...	generic arguments, to pass to other functions.

Details

The iit-transform maps D amounts (considered in a real geometry) isometrically to a D dimensonal euclidian vector. The `iit` is part of the `rplus` framework. Despite its trivial operation, it is present to achieve maximal analogy between the `aplus` and the `rplus` framework.
The data can then be analysed in this transformated space by all classical multivariate analysis tools. The interpretation of the results is easy since the relation to the original variables is preserved. However results may be inconsistent, since the multivariate analysis tools disregard the positivity condition and the inner laws of amounts.

The isometric identity transform is a simple identity given by

$$iit(x)_i := x_i$$

Value

`ilt` gives the isometric identity transform, i.e. simply the input stripped of the "rplus" class attribute, `iptInv` gives amounts with class "rplus" with the given iit, i.e. simply the argument checked to be a valid "rplus" object, and with this class attribute.

Note

`iit` can be used to unclass amounts.

Author(s)

K.Gerald v.d. Boogaart `http://www.stat.boogaart.de`

References

van den Boogaart, K.G. and R. Tolosana-Delgado (2008) "compositions": a unified R package to analyze Compositional Data, *Computers & Geosciences*, 34 (4), pages 320-338, doi: 10.1016/j.cageo.2006.11.017.

See Also

`ilt`, `ilr`, `rplus`

Examples

```
(tmp <- iit(c(1,2,3)))
iitInv(tmp)
iitInv(tmp) - c(1,2,3) # 0
data(Hydrochem)
```

```
cdata <- Hydrochem[,6:19]
pairs(iit(cdata))
```

`ilr` *Isometric log ratio transform*

Description

Compute the isometric log ratio transform of a (dataset of) composition(s), and its inverse.

Usage

```
ilr( x , V = ilrBase(x) ,...)
ilrInv( z , V = ilrBase(z=z),..., orig=gsi.orig(z))
```

Arguments

`x`	a composition, not necessarily closed
`z`	the ilr-transform of a composition
`V`	a matrix, with columns giving the chosen basis of the clr-plane
`...`	generic arguments. not used.
`orig`	a compositional object which should be mimicked by the inverse transformation. It is especially used to reconstruct the names of the parts.

Details

The ilr-transform maps a composition in the D-part Aitchison-simplex isometrically to a D-1 dimensonal euclidian vector. The data can then be analysed in this transformation by all classical multivariate analysis tools. However the interpretation of the results may be difficult, since there is no one-to-one relation between the original parts and the transformed variables.

The isometric logratio transform is given by

$$ilr(x) := V^t clr(x)$$

with `clr`(x) the centred log ratio transform and $V \in R^{d\times(d-1)}$ a matrix which columns form an orthonormal basis of the clr-plane. A default matrix V is given by `ilrBase(`D`)`.

Value

`ilr` gives the isometric log ratio transform, `ilrInv` gives closed compositions with the given ilr-transforms

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de, Raimon Tolosana-Delgado

References

Egozcue J.J., V. Pawlowsky-Glahn, G. Mateu-Figueras and C. Barcel'o-Vidal (2003) Isometric logratio transformations for compositional data analysis. *Mathematical Geology*, **35**(3) 279-300
Aitchison, J, C. Barcel'o-Vidal, J.J. Egozcue, V. Pawlowsky-Glahn (2002) A consise guide to the algebraic geometric structure of the simplex, the sample space for compositional data analysis, *Terra Nostra*, Schriften der Alfred Wegener-Stiftung, 03/2003
https://ima.udg.edu/Activitats/CoDaWork03/

See Also

clr,alr,apt, ilrBase

Examples

```
(tmp <- ilr(c(1,2,3)))
ilrInv(tmp)
ilrInv(tmp) - clo(c(1,2,3)) # 0
data(Hydrochem)
cdata <- Hydrochem[,6:19]
pairs(ilr(cdata))
ilrBase(D=3)
```

---

ilrBase *The canonical basis in the clr plane used for ilr and ipt transforms.*

Description

Compute the basis of a clr-plane, to use with isometric log-ratio or planar transform of a (dataset of) compositions.

Usage

```
ilrBase( x=NULL , z=NULL , D = NULL, method = "basic" )
```

Arguments

x	optional dataset or vector of compositions
z	optional dataset or vector containing ilr or ipt coordinates
D	number of parts of the simplex
method	method to build the basis, one of "basic", "balanced", "optimal" "PBhclust", "PBmaxvar" or "PBangprox"

Details

Method "basic" computes a triangular Helmert matrix (corresponding to the original ilr transformation defined by Egozcue et al, 2003). In this case, `ilrBase` is a wrapper catching the answers of `gsi.ilrBase` and is to be used as the more convenient function.

Method "balanced" returns an ilr matrix associated with a balanced partition, splitting the parts in groups as equal as possible. Transforms `ilr` and `ipt` computed with this basis are less affected by any component (as happens with "basic").

The following methods are all data-driven and will fail if `x` is not given. Some of these methods are extended to non-acomp datasets via the `cpt` general functionality. Use with care with non-acomp objects!

Method "optimal" is a wrapper to `gsi.optimalilrBase`, providing the ilr basis with less influence of missing values. It is computed as a hierarchical cluster of variables, with parts previously transformed to 1 (if the value is lost) or 0 (if it is recorded).

Methods "PBhclust", "PBmaxvar" and "PBangprox" are principal balance methods (i.e. balances approximating principal components in different ways). These are all resolved by calls to `gsi.PrinBal`. Principal balances functionality should be considered beta!

Value

All methods give a matrix containing by columns the basis elements for the canonical basis of the clr-plane used for the ilr and ipt transform. Only one of the arguments `x`, `z` or `D` is needed to determine the dimension of the simplex.

If you provide transformed data `z`, the function attempts to extract the basis information from it with `gsi.getV`. Otherwise, the default compatible ilr base matrix is created.

References

Egozcue J.J., V. Pawlowsky-Glahn, G. Mateu-Figueras and C. Barcel'o-Vidal (2003) Isometric logratio transformations for compositional data analysis. *Mathematical Geology*, **35**(3) 279-300

https://ima.udg.edu/Activitats/CoDaWork03/

See Also

`clr`,`ilr`,`ipt`

Examples

```
ilr(c(1,2,3))
ilrBase(D=2)
ilrBase(c(1,2,3))
ilrBase(z= ilr(c(1,2,3)) )
round(ilrBase(D=7),digits= 3)
ilrBase(D=7,method="basic")
ilrBase(D=7,method="balanced")
```

`ilt` *Isometric log transform*

Description

Compute the isometric log transform of a vector (dataset) of amounts and its inverse.

Usage

```
ilt( x ,...)
iltInv( z ,... )
```

Arguments

`x`	a vector or data matrix of amounts
`z`	the ilt-transform of a vector or data matrix of ilt-transforms of amounts
`...`	generic arguments, not used.

Details

The ilt-transform maps D amounts (considered in log geometry) isometrically to a D dimensional euclidean vector. The `ilt` is part of the `aplus` framework.
The data can then be analysed in this transformation by all classical multivariate analysis tools. The interpretation of the results is easy since the relation to the original variables is preserved.

The isometric log transform is given by

$$ilt(x)_i := \ln x_i$$

Value

`ilt` gives the isometric log transform, i.e. simply the log of the argument, whereas `iltInv` gives amounts with the given ilt, i.e. simply the exp of the argument.

Author(s)

K.Gerald v.d. Boogaart `http://www.stat.boogaart.de`

References

van den Boogaart, K.G. and R. Tolosana-Delgado (2008) "compositions": a unified R package to analyze Compositional Data, *Computers & Geosciences*, 34 (4), pages 320-338, doi: 10.1016/j.cageo.2006.11.017.

See Also

`ilr`, `iit`, `aplus`

Examples

```
(tmp <- ilt(c(1,2,3)))
iltInv(tmp)
iltInv(tmp) - c(1,2,3) # 0
data(Hydrochem)
cdata <- Hydrochem[,6:19]
pairs(ilt(cdata))
```

`ipt` *Isometric planar transform*

Description

Compute the isometric planar transform of a (dataset of) composition(s) and its inverse.

Usage

```
ipt( x , V = ilrBase(x),... )
iptInv( z , V = ilrBase(z=z),...,orig=gsi.orig(z))
uciptInv( z , V = ilrBase(z=z),...,orig=gsi.orig(z) )
```

Arguments

`x`	a composition or a data matrix of compositions, not necessarily closed
`z`	the ipt-transform of a composition or a data matrix of ipt-transforms of compositions
`V`	a matrix with columns giving the chosen basis of the clr-plane
`...`	generic arguments. not used.
`orig`	a compositional object which should be mimicked by the inverse transformation. It is especially used to reconstruct the names of the parts.

Details

The ipt-transform maps a composition in the D-part real-simplex isometrically to a D-1 dimensonal euclidian vector. Although the transformation does not reach the whole R^{D-1}, resulting covariance matrices are typically of full rank.
The data can then be analysed in this transformation by all classical multivariate analysis tools. However, interpretation of results may be difficult, since the transform does not keep the variable names, given that there is no one-to-one relation between the original parts and each transformed variables. See `cpt` and `apt` for alternatives.

The isometric planar transform is given by

$$ipt(x) := V^t cpt(x)$$

with `cpt`(x) the centred planar transform and $V \in R^{d\times(d-1)}$ a matrix which columns form an orthonormal basis of the clr-plane. A default matrix V is given by `ilrBase(`D`)`

Value

ipt gives the centered planar transform, iptInv gives closed compositions with with the given ipt-transforms, uciptInv unconstrained iptInv does the same as iptInv but sets illegal values to NA rather than giving an error. This is a workaround to allow procedures not honoring the constraints of the space.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

van den Boogaart, K.G. and R. Tolosana-Delgado (2008) "compositions": a unified R package to analyze Compositional Data, *Computers & Geosciences*, 34 (4), pages 320-338, doi: 10.1016/j.cageo.2006.11.017.

See Also

ilr,ilrBase, cpt

Examples

```
(tmp <- ipt(c(1,2,3)))
iptInv(tmp)
iptInv(tmp) - clo(c(1,2,3)) # 0
data(Hydrochem)
cdata <- Hydrochem[,6:19]
pairs(ipt(cdata))
```

---

is.acomp *Check for compositional data type*

Description

is.*XXX* returns TRUE if and only if its argument is of type *XXX*

Usage

```
is.acomp(x)
is.rcomp(x)
is.aplus(x)
is.rplus(x)
is.rmult(x)
is.ccomp(x)
```

Arguments

x	any object to be checked

Details

These functions only check for the class of the object.

Value

TRUE or FALSE

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

acomp, rcomp aplus, rplus

Examples

```
is.acomp(1:3)
is.acomp(acomp(1:3))
is.rcomp(acomp(1:3))
is.acomp(acomp(1:3)+acomp(1:3))
```

IsMahalanobisOutlier *Checking for outliers*

Description

Detect outliers with respect to a normal distribution model.

Usage

```
IsMahalanobisOutlier(X,...,alpha=0.05,goodOnly=NULL,
                     replicates=1000,corrected=TRUE,robust=TRUE,crit=NULL)
```

Arguments

X	a dataset (e.g. given as acomp, rcomp, aplus, rplus or rmult) object to which idt and MahalanobisDist apply.
...	further arguments to MahalanobisDist/gsi.mahOutlier
alpha	The confidence level for identifying outliers.
goodOnly	an integer vector. Only the specified index of the dataset should be used for estimation of the outlier criteria. This parameter if only a small portion of the dataset is reliable.
replicates	The number of replicates to be used in the Monte Carlo simulations for determination of the quantiles. The replicates not given a minimum is computed from the alpha level to ensure reasonable precission.

corrected	logical. Literatur often proposed to compare the Mahalanobis distances with Chisq-Approximations of there distributions. However this does not correct for multiple testing. If corrected is true a correction for multiple testing is used. In any case we do not use the chisq-approximation, but a simulation based procedure to compute confidence bounds.
robust	A robustness description as define in robustnessInCompositions
crit	The critical value to be used. Typically the routine is called mainly for the purpose of finding this value, which it does, when crit is NULL, however sometimes we might want to specifiy a value used by someone else to reproduce the results.

Details

See outliersInCompositions and robustnessInCompositions for a comprehensive introduction into the outlier treatment in compositions.

See OutlierClassifier1 for a highlevel method to classify observations in the context of outliers.

Value

A logical vector giving for each element the result of the alpha-level test for beeing an outlier. TRUE corresponds to a significant result.

Note

For some unkown reasons the computation sometimes produces NaN's. In this case a warning is issued and a recomputation is tried.

The package **robustbase** is required for using the robust estimations.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

OutlierClassifier1 , outlierplot, ClusterFinder1

Examples

```
## Not run:
data(SimulatedAmounts)

datas <- list(data1=sa.outliers1,data2=sa.outliers2,data3=sa.outliers3,
              data4=sa.outliers4,data5=sa.outliers5,data6=sa.outliers6)

opar<-par(mfrow=c(2,3),pch=19,mar=c(3,2,2,1))
tmp<-mapply(function(x,y){
plot(x,col=ifelse(IsMahalanobisOutlier(x),"red","gray"))
  title(y)
},datas,names(datas))

## End(Not run)
```

isoPortionLines *Isoportion- and Isoproportion-lines*

Description

Add lines of equal portion and proportion to ternary diagrams, to serve as reference axis.

Usage

```
isoPortionLines(...)
## S3 method for class 'acomp'
isoPortionLines(by=0.2,at=seq(0,1,by=by),...,
                      parts=1:3,total=1,labs=TRUE,lines=TRUE,unit="")
## S3 method for class 'rcomp'
isoPortionLines(by=0.2,at=seq(0,1,by=by),...,
                      parts=1:3,total=1,labs=TRUE,lines=TRUE,unit="")
isoProportionLines(...)
## S3 method for class 'acomp'
isoProportionLines(by=0.2,at=seq(0,1,by=by),...,
                      parts=1:3,labs=TRUE,lines=TRUE)
## S3 method for class 'rcomp'
isoProportionLines(by=0.2,at=seq(0,1,by=by),...,
                      parts=1:3,labs=TRUE,lines=TRUE)
```

Arguments

...	graphical arguments
at	numeric in [0,1]: which portions/proportions should be marked?
by	numeric in (0,1]: steps between protions/proportions
parts	numeric vector subset of {1,2,3}: the variables to be marked
total	the total amount to be used in labeling
labs	logical: plot the labels?
lines	logical: plot the lines?
unit	mark of the units e.g. "%"

Details

Isoportion lines give lines of the same portion of one of the parts, while isoproportion line gives lines of the same ratio between two parts. The isoproportion lines are straight lines in both the Aitchison and the real geometries of the simplex, while the isoportion lines are not straight in an Aitchison sense (only in the real one). However, note that both types of lines remain straight in the real sense when perturbed (von Eynatten et al., 2002).

Note

Currently IsoLines only works with individual plots. This is mainly due to the fact that I have no idea, what the user interface of this function should look like for multipanel plots. This includes philosophical problems with the meaning of isoportions in case of marginal plots.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

von Eynatten, H., V. Pawlowsky-Glahn, and J.J. Egozcue (2002) Understanding perturbation on the simplex: a simple method to better visualize and interpret compositional data in ternary diagrams. *Mathematical Geology* **34**, 249-257

See Also

plot.acomp

Examples

```
data(SimulatedAmounts)
plot(acomp(sa.lognormals))
isoPortionLines()
plot(acomp(sa.lognormals),center=TRUE)
isoPortionLines()
plot(rcomp(sa.lognormals))
isoPortionLines()
plot(acomp(sa.lognormals))
isoProportionLines()
plot(acomp(sa.lognormals),center=TRUE)
isoProportionLines()
plot(rcomp(sa.lognormals))
isoProportionLines()
```

---

jura *The jura dataset*

Description

A geochemical dataset from the Swiss Jura.

Usage

```
data(juraset)
data(jura259)
```

Format

A 359x11 or 259x11 dataframe

Details

The JURA data set provided by J.-P. Dubois, IATE-Paedologie, Ecole Polytechnique Federale de Lausanne, 1015 Lausanne, Switzerland. Spatial coordinates and values of categorial and continuous attributes at the 359 sampled sites. The 100 test locartions are denoted with a star. Rock Types: 1: Argovian, 2: Kimmeridgian, 3: Sequanian, 4: Portlandian, 5: Quaternary. Land uses: 1: Forest, 2: Pasture, 3: Meadow , 4: Tillage

X	X location coordinate
Y	Y location coordinate
Rock	Categorical: rocktype,
Land	Categorical: land usage
Cd	element amount,
Cu	element amount,
Pb	element amount,
Co	element amount,
Cr	element amount,
Ni	element amount,

All 3-part compositions sum to one.

Source

AI-Geostats

References

Atteia, O., Dubois, J.-P., Webster, R., 1994, Geostatistical analysis of soil contamination in the Swiss Jura: Environmental Pollution 86, 315-327

Webster, R., Atteia, O., Dubois, J.-P., 1994, Coregionalization of trace metals in the soil in the Swiss Jura: European Journal of Soil Science 45, 205-218

Examples

```
## Not run:
data(juraset)
X <- with(juraset,cbind(X,Y))
comp <- acomp(juraset,c("Cd","Cu","Pb","Co","Cr"))
lrv <- logratioVariogram(comp,X,maxdist=1,nbins=10)
plot(lrv)

## End(Not run)
```

`kdeDirichlet` *Density estimation on the simplex with Dirichlet kernel*

Description

Function to compute the kernel density estimation on a grid of the simplex

Usage

```
kdeDirichlet(x, adj = 1, n = 200, kdegrid = NULL, delta = FALSE)
```

Arguments

`x`	data set of (complete) compositional data, i.e. data summing to 1 by columns
`adj`	accessory scaling factor, for modifying the bandwith in analogy to function [MASS::kde()]
`n`	integer, number of grid nodes on each component, where to estimate the density; ignored if 'kdegrid' is given
`kdegrid`	data frame, set of locations where to estimate the density; either specify 'n' or 'kdegrid'
`delta`	logical or real controlling if/how zeroes in 'x' are treated; logical works only for 'kdegrid=NULL' and uses correction by half the grid cell size, otherwise give a real value; this value will be added to the whole composition, including the non-zero values.

Details

This function computes the kde (kernel density estimation) of the probability density function of a random composition on the simplex, by using Dirichlet kernels. The method was proposed by Aitchison and Lauder (1985).

Value

A list of two or three elements, depending on the value of 'kdegrid'. If 'kdegrid' is null, the function is assumed to be used for two-dimensional plotting, and the output is one compatible with the function [image()], i.e. a list of three elements (vector of x-values, vector of y-values and matrix of density values computed). If 'kdegrid' is a grid, then the output has two elements: the input grid and a vector of computed densities.

NOTE: no effort is made to check that 'kdegrid' has the right class, dimension or content.

References

Aitchison J., Lauder I.J. (1985) Kernel density estimation for compositional data; _J. Roy. Statist. Soc. Ser. C_, 34 (2): 129-137.

Ouimet, F. and Tolosana-Delgado, R. (2022) Asymptotic properties of Dirichlet kernel density estimators; _Journal of Multivariate Analysis_ 187: 104832, doi: 10.1016/j.jmva.2021.104832

kingTetrahedron *Ploting composition into rotable tetrahedron*

Description

Plots acomp/rcomp objects into tetrahedron exported in kinemage format.

Usage

```
kingTetrahedron(X, parts=1:4, file="tmptetrahedron.kin",
clu=NULL,vec=NULL, king=TRUE, scale=0.2, col=1,
title="Compositional Tetrahedron")
```

Arguments

X	a compositional acomp or rcomp object of 4 or more parts
parts	a numeric or character vector specifying the 4 parts to be used.
file	file.kin for 3D display with the KiNG (Kinemage, Next Generation) interactive system for three-dimensional vector graphics.
clu	partition determining the colors of points
vec	vector of values determining points sizes
king	FALSE for Mage; TRUE for King (described below)
scale	relative size of points
col	color of points if clu=NULL
title	The title of the plot

Details

The routine transforms a 4 parts mixture m quadrays into 3-dimensional XYZ coordinates and writes them as file.kin. For this transformation we apply K. Urner: Quadrays and XYZ at http://www.grunch.net/synergetics/quadxyz.html. The kin file we display as 3-D animation with KiNG viewer. A kinemage is a dynamic, 3-D illustration. The best way to take advantage of that is by rotating it and twisting it around with the mouse click near the center of the graphics window and slowly draging right or left, up or down. Furthermore by clicking on points with the mouse (left button again), the label associated with each point will appear in the bottom left of the graphics area and also the distance from this point to the last will be displayed. With the right button drag we can zoom in and out of the picture. This animation supports coloring and different sizing of points.

We can display the kin file as 3-D animation also with MAGE viewer a previous version of KiNG,

more information (and links to the software) can be found at https://en.wikipedia.org/wiki/Kinemage. For this one has to put king=FALSE as a parameter.

Value

The function is called for its side effect of generating a file for 3D display with the KiNG (Kinemage, Next Generation) interactive system for three-dimensional vector graphics. Works only with KiNG viewer. More information (and links to the actual viewers) can be found at https://en.wikipedia.org/wiki/Kinemage

Note

This routine and the documentation is based on mix.Quad2net from the MixeR-package of Vladimir Batagelj and Matevz Bren, and has been contributed by Matevz Bren to this package. Only slight modifications have been applied to make function compatible with the philosophy and objects of the compositions package.

Author(s)

Vladimir Batagelj and Matevz Bren, with slight modifications of K.Gerald van den Boogaart

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

http://vlado.fmf.uni-lj.si/pub/MixeR/

http://www.grunch.net/synergetics/quadxyz.html

See Also

plot.acomp

Examples

```
## Not run:
data(SimulatedAmounts)
dat <- acomp(sa.groups5)
hc <- hclust(dist(dat), method = "complete")  # data are clustered
kingTetrahedron(dat,parts=1:4, file="myfirst.kin", clu=cutree(hc,7),
scale=0.2)
# the 3-D plot is written into Glac1.kin file to be displayed with KiNG viewer.
# The seven clusters partition is notated with different colors of points.

## End(Not run)
```

Kongite *Compositions of 25 specimens of kongite*

Description

A mineral compositions of 25 rock specimens of kongite type. Each composition consists of the percentage by weight of five minerals, albite, blandite, cornite, daubite, endite.

Usage

```
data(Kongite)
```

Details

A mineral compositions of 25 rock specimens of hongite type. Each composition consists of the percentage by weight of five minerals, albite, blandite, cornite, daubite, endite, which we conveniently abbreviate to A, B, C, D, E. All row percentage sums to 100.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name HONGITE.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison, J. (1986) The Statistical Analysis of Compositional Data, (Data 2), pp2.

lines *Draws connected lines from point to point.*

Description

Functions taking coordinates given in various ways and joining the corresponding points with line segments.

Usage

```
          ## S3 method for class 'acomp'
lines(x,...,steps=30,aspanel=FALSE)
          ## S3 method for class 'rcomp'
lines(x,...,steps=30,aspanel=FALSE)
          ## S3 method for class 'aplus'
lines(x,...,steps=30,aspanel=FALSE)
          ## S3 method for class 'rplus'
lines(x,...,steps=30,aspanel=FALSE)
          ## S3 method for class 'rmult'
lines(x,...,steps=30,aspanel=FALSE)
```

Arguments

x	a dataset of the given type
...	further graphical parameters
steps	the number of discretisation points to draw the segments, which might be not visually straight.
aspanel	Logical, indicates use as slave to do acutal drawing only.

Details

The functions add lines to the graphics generated with the corresponding plot functions.
Adding to multipaneled plots, redraws the plot completely and is only possible, when the plot has been created with the plotting routines from this library.
For the rcomp/rplus geometries the main problem is providing a function that reasonably works with lines leaving the area. We tried to use a policy of cuting the line at the actual borders of the (high dimensional) simplex. That can lead to very strange visual impression showing lines ending somewhere in the middle of the plot. However these lines actually hit some border of the simplex that is not shown in the plot. A hyper dimensional tetrahedron is even more difficult to imagin than a hyperdimensional cube.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de, Raimon Tolosana-Delgado

See Also

plot.acomp, straight

Examples

```
data(SimulatedAmounts)

plot(acomp(sa.lognormals))
lines(acomp(sa.lognormals),col="red")
lines(rcomp(sa.lognormals),col="blue")
```

```
plot(aplus(sa.lognormals[,1:2]))
lines(aplus(sa.lognormals[,1:2]),col="red")
lines(rplus(sa.lognormals)[,1:2],col="blue")

plot(rplus(sa.lognormals[,1:2]))
tt<-aplus(sa.lognormals[,1:2]); ellipses(mean(tt),var(tt),r=2,col="red")
tt<-rplus(sa.lognormals[,1:2]); ellipses(mean(tt),var(tt),r=2,col="blue")
tt<-rmult(sa.lognormals[,1:2]); ellipses(mean(tt),var(tt),r=2,col="green")
```

`logratioVariogram` *Empirical variograms for compositions*

Description

Computes the matrix of logratio variograms.

Usage

```
  logratioVariogram(data, ...)

  ## S3 method for class 'acomp'
logratioVariogram(data,
                          loc,
                          maxdist=max(dist(loc))/2,
                          nbins=20,
                          dists=seq(0,maxdist,length.out=nbins+1),
                          bins=cbind(dists[-length(dists)],dists[-1]),
                          azimuth=0,
                          azimuth.tol=180,
                          comp=data,
                          ...
                          )
```

Arguments

`data`	an acomp compositional dataset
`...`	arguments for generic functionality
`loc`	a matrix or dataframe providing the observation locations of the compositions. Any number of dimension >= 2 is supported.
`maxdist`	the maximum distance to compute the variogram for.
`nbins`	The number of distance bins to compute the variogram for
`dists`	The distances seperating the bins
`bins`	a matrix with lower and upper limit for the distances of each bin. A pair is counted if min<h<=max. min and max are provided as columns. bins is computed from maxdist,nbins and dists. If it is provided, it is used directly.

azimuth	For directional variograms the direction, either as an azimuth angle (i.e. a single real number) for 2D datasets or a unit vector pointing of the same dimension as the locations. The angle is clockwise from North in degree.
azimuth.tol	The angular tolerance it should be below 90 if a directional variogram is intended.
comp	do not use, only provided for backwards compatibility. Use data instead

Details

The logratio-variogram is the set of variograms of each of the pairwise logratios. It can be proven that it carries the same information as a usual multivariate variogram. The great advantage is that all the funcitions have a direct interpreation and can be estimated even with (MAR) missings in the dataset.

Value

A list of class "logratioVariogram".

vg	A nbins x D x D array containing the logratio variograms
h	A nbins x D x D array containing the mean distance the value is computed on.
n	A nbins x D x D array containing the number of nonmissing pairs used for the corresponding value.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

Tolosana, van den Boogaart, Pawlowsky-Glahn (2009) Estimating and modeling variograms of compositional data with occasional missing variables in R, StatGis09

Pawlowsky-Glahn, Vera and Olea, Ricardo A. (2004) Geostatistical Analysis of Compositional Data, Oxford University Press, Studies in Mathematical Geology

See Also

vgram2lrvgram, CompLinModCoReg, vgmFit

Examples

```
## Not run:
data(juraset)
X <- with(juraset,cbind(X,Y))
comp <- acomp(juraset,c("Cd","Cu","Pb","Co","Cr"))
lrv <- logratioVariogram(comp,X,maxdist=1,nbins=10)
plot(lrv)

## End(Not run)
```

`lrvgram` *vgram2lrvgram*

Description

Transforms model functions for different types of compositional (logratio)(co)variograms.

Usage

```
cgram2vgram(cgram)
vgram2lrvgram(vgram)
```

Arguments

`cgram`	A (matrix valued) covariance function.
`vgram`	A (matrix valued) variogram functions.

Details

The variogram is given by `cgram(0)-cgram(h)` and `lrvgram(h)[,i,j]==vgram(h)[,i,i]+vgram(h)[,i,j]-2*vgram(h)`

The logratio-variogram is the set of variograms of each of the pairwise logratios. It can be proven that it carries the same information as a usual multivariate variogram. The great advantage is that all the funcitions have a direct interpreation and can be estimated even with (MAR) missings in the dataset.

Value

A function that takes the same parameters as the input function (through a ... parameterlist), but provides the correponding variogram values (cgram2vgram) or logratio Variogram (vgram2lrvgram) values.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

Tolosana, van den Boogaart, Pawlowsky-Glahn (2009) Estimating and modeling variograms of compositional data with occasional missing variables in R, StatGis09

See Also

`logratioVariogram`, `CompLinModCoReg`, `vgmFit`

Examples

```
data(juraset)
comp <- acomp(juraset,c("Cd","Cu","Pb","Co","Cr"))
vg <- CompLinModCoReg(~nugget()+sph(0.5)+R1*exp(0.7),comp)
vg(1:3)
vgram2lrvgram(vg)(1:3)
```

`MahalanobisDist` *Compute Mahalanobis distances based von robust Estimations*

Description

MahalanobisDist computes the Mahalanobis distances to the center or to other observations.

Usage

```
MahalanobisDist(x,center=NULL,cov=NULL,inverted=FALSE,...)
## S3 method for class 'rmult'
MahalanobisDist(x,center=NULL,cov=NULL,inverted=FALSE,...,
           goodOnly=NULL,pairwise=FALSE,pow=1,
robust=FALSE,giveGeometry=FALSE)
## S3 method for class 'acomp'
MahalanobisDist(x,center=NULL,cov=NULL,inverted=FALSE,...,
           goodOnly=NULL, pairwise=FALSE,pow=1,robust=FALSE,giveGeometry=FALSE)
```

Arguments

`x`	the dataset
`robust`	logical or a robust method description (see `robustnessInCompositions`) specifiying how the center and covariance matrix are estimated,if not given.
`...`	Further arguments to `solve`.
`center`	An estimated for the center (mean) of the dataset. If center is NULL it will be estimated based using the given robust option.
`cov`	An estimated for the spread (covariance matrix) of the dataset. If cov is NULL it will be estimated based using the given robust option.
`inverted`	TRUE if the inverse of the covariance matrix is given.
`goodOnly`	An vector of indices to the columns of x that should be used for estimation of center and spread.
`pairwise`	If FALSE the distances to the center are returned as a vector. If TRUE the distances between the cases are returned as a distance matrix.
`pow`	The power of the Mahalanobis distance to be used. 1 correponds to the square root of the squared distance in transformed space, like it is defined in most books. The choice 2 corresponds to what is implemented in many software package including the `mahalanobis` function in R.
`giveGeometry`	If true an atrributes `"center"` and `"cov"` given the center and the idt-variance used for the calculations.

Details

The Mahalanobis distance is the distance in a linearly transformed space, where the linear transformation is selected in such a way,that the variance is the unit matrix. Thus the distances are given in multiples of standard deviation.

Value

Either a vector of Mahalanobis distances to the center, or a distance matrix (like from `dist`) giving the pairwise Mahalanobis distances of the data.

Note

Unlike the `mahalanobis` function this function does not be default compute the square of the mahalanobis distance. The pow option is provided if the square is needed.
The package **robustbase** is required for using the robust estimations.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

`dist`, `OutlierClassifier1`

Examples

```
data(SimulatedAmounts)
data5 <- acomp(sa.outliers5)

  cl <- ClusterFinder1(data5,sigma=0.4,radius=1)
  plot(data5,col=as.numeric(cl$types),pch=as.numeric(cl$types))
  legend(1,1,legend=levels(cl$types),xjust=1,col=1:length(levels(cl$types)),
             pch=1:length(levels(cl$types)))
```

---

matmult *inner product for matrices and vectors*

Description

Multiplies two matrices, if they are conformable. If one argument is a vector, it will be coerced to either a row or a column matrix to make the two arguments conformable. If both are vectors it will return the inner product.

Usage

```
x %*% y
## Default S3 method:
x %*% y
```

Arguments

x,y	numeric or complex matrices or vectors

Details

This is a copy of the base::%*% function. The function is made generic to allow the definition of specific methods.

Value

The matrix product. Uses 'drop' to get rid of dimensions which have only one level.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

%*%.rmult

Examples

```
M <- matrix(c(
0.2,0.1,0.0,
0.1,0.2,0.0,
0.0,0.0,0.2),byrow=TRUE,nrow=3)
x <- c(1,1,2)
M %*% x
x %*% M
x %*% x
M %*% M
t(x) %*% M
```

---

mean.acomp *Mean amounts and mean compositions*

Description

Compute the mean in the several approaches of compositional and amount data analysis.

Usage

```
          ## S3 method for class 'acomp'
mean(x,...,robust=getOption("robust"))
          ## S3 method for class 'rcomp'
mean(x,...,robust=getOption("robust"))
          ## S3 method for class 'aplus'
```

```
mean(x,...,robust=getOption("robust"))
          ## S3 method for class 'rplus'
mean(x,...,robust=getOption("robust"))
          ## S3 method for class 'ccomp'
mean(x,...,robust=getOption("robust"))
          ## S3 method for class 'rmult'
mean(x,...,na.action=NULL,robust=getOption("robust"))
```

Arguments

x	a classed dataset of amounts or compositions
...	further arguments to mean e.g. trim
na.action	na.action
robust	A description of a robust estimator. Possible values are FALSE or "pearson" for no robustness, or TRUE or "mcd" for a covMcd based robust location scale estimation. Additional control parameters such as list(trim=0.2) or an rrcov.control object can be given as an attribute "control".

Details

The different compositional approaches acomp, rcomp, aplus, rplus correpond to different geometries. The mean is calculated in the respective canonical geometry by applying a canonical transform (see cdt), taking ordinary meanCol and backtransforming.

The Aitchison geometries imply that mean.acomp and mean.aplus are geometric means, the first one closed. The real geometry implies that mean.rcomp and mean.rplus are arithmetic means, the first one resulting in a closed composition.

In all cases the mean is again an object of the same class.

Value

The mean is given as a composition or amount vector of the same class as the original dataset.

Missing Policy

For the additive scales (rcomp,rplus) the SZ and BDL are treated as zeros and MAR and MNAR as missing information. This is not strictly correct for MNAR.
For relative scales (acomp,aplus), all four types of missings are treated as missing information. This corresponds to the idea that BDL are truncated values (and have the correspoding effect in taking means). For SZ and MAR, only the components in the observed subcomposition are fully relevant. Finally, for MNAR the problem is again that nothing could be done without knowing the MNAR mechanism, so the analysis is limited to taking them as MAR, and being *careful* with the interpretation. Missing and Below Detecion Limit Policy is explained in more detail in compositions.missing.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

clo, meanCol, geometricmean, acomp, rcomp, aplus, rplus

Examples

```
data(SimulatedAmounts)
meanCol(sa.lognormals)
mean(acomp(sa.lognormals))
mean(rcomp(sa.lognormals))
mean(aplus(sa.lognormals))
mean(rplus(sa.lognormals))
mean(rmult(sa.lognormals))
```

---

meanrow *The arithmetic mean of rows or columns*

Description

Computes the arithmetic mean.

Usage

```
meanRow(x,..., na.action=get(getOption("na.action")))
meanCol(x,..., na.action=get(getOption("na.action")))
```

Arguments

x	a numeric vector or matrix of data
...	arguments to mean
na.action	The na.action to be used: one of na.omit,na.fail,na.pass

Details

Computes the arithmetic means of the rows (meanRow) or columns (meanCol) of x.

Value

The arithmetic means of the rows (meanRow) or columns (meanCol) of x.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

mean.rplus

Examples

```
data(SimulatedAmounts)
meanCol(sa.tnormals)
meanRow(sa.tnormals)
```

`Metabolites` *Steroid metabolite patterns in adults and children*

Description

Data shows the urinary excretion (mg/24 hours) of 37 normal adults and 30 normal children of total cortisol meatbolites, total corticosterone meatbolites, total pregnanetriol and Δ-5-pregnentriol.

Usage

```
data(Metabolites)
```

Details

There are 67 cases for 37 adults and 30 children, and 5 columns: Case no., met1, met2, met3 and Type, 1 for adults, -1 for children. No sum constraint is placed on this data set: since the urinary excretion in mg for 24 hours are given.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name METABOL.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison, J. (1986) The Statistical Analysis of Compositional Data, (Data 9), pp14.

`missing.compositions` *The policy of treatment of missing values in the "compositions" package*

Description

This help section discusses some general strategies of working with missing valuess in a compositional, relative or vectorial context and shows how the various types of missings are represented and treated in the "compositions" package, according to each strategy/class of analysis of compositions or amounts.

Usage

```
is.BDL(x,mc=attr(x,"missingClassifier"))
is.SZ(x,mc=attr(x,"missingClassifier"))
is.MAR(x,mc=attr(x,"missingClassifier"))
is.MNAR(x,mc=attr(x,"missingClassifier"))
is.NMV(x,mc=attr(x,"missingClassifier"))
is.WMNAR(x,mc=attr(x,"missingClassifier"))
is.WZERO(x,mc=attr(x,"missingClassifier"))
has.missings(x,...)
## Default S3 method:
has.missings(x,mc=attr(x,"missingClassifier"),...)
## S3 method for class 'rmult'
has.missings(x,mc=attr(x,"missingClassifier"),...)
SZvalue
MARvalue
MNARvalue
BDLvalue
```

Arguments

`x`	A vector, matrix, acomp, rcomp, aplus, rplus object for which we would like to know the missing status of the entries
`mc`	A missing classifier function, giving for each value one of the values BDL (Below Detection Limit), SZ (Structural Zero), MAR (Missing at random), MNAR (Missing not at random), NMV (Not missing value) This functions are introduced to allow a different coding of the missings.
`...`	further generic arguments

Details

In the context of compositional data we have to consider at least four types of missing and zero values:

MAR (Missing at random) coded by NaN, the amount was not observed or is otherwise missing, in a way unrelated to its actual value. This is the "nice" type of missing.

MNAR (Missing not at random) coded by NA, the amount was not observed or is otherwise missing, but it was missed in a way stochastically dependent on its actual value.

BDL (Below detection limit) coded by 0.0 or a negative number giving the detection limit; the amount was observed but turned out to be below the detection limit and was thus rounded to zero. This is an informative version of MNAR.

SZ (Structural zero) coded by -Inf, the amount is absolutely zero due to structural reasons. E.g. a soil sample was dried before the analysis, or the sample was preprocessed so that the fraction is removed. Structural zeroes are mainly treated as MAR even though they are a kind of MNAR.

Based on these basic missing types, the following extended types are defined:

NMV (Not Missing Value) coded by a real number, it is just an actually-observed value.

WMNAR (Wider MNAR) includes BDL and MNAR.

WZERO (Wider Zero) includes BDL and SZ

Each function of type `is.XXX` checks the status of its argument according to the XXX type of value from those above.

Different steps of a statistical analysis and different understanding of the data will lead to different approaches with respect to missings and zeros.
In the first exploratory step, the problem is to keep the methods working and to make the missing structure visible in the analysis. The user should need as less as possible extra thinking about missings, an get nevertheless a true picture of the data. To achieve this we tried to make the basic layer of computational functions working consitently with missings and propagating the missingness character seamlessly. However some of this only works with `acomp`, where a closed form missing theories are available (e.g. proportional imputation [e.g. Mart\'in-Fern\'andez, J.A. et al.(2003)]or estimation with missings [Boogaart&Tolosana 2006]). The main graphics should hint towards missing and try to add missings to the plot by marking the remaining informaion on the axes. However one again should be clear that this is only reasonably justified in the relative geometries. Unfortunatly the missing subsystem is currently not fully compatible with the robustness subsystem.
As a second step, the analyst might want to analyse the missing structure for itself. This is preliminarly provided by these functions, since their result can be treated as a boolean data set in any other R function. Additionally a `missingSummary` provides some a convenience function to provide a fast overview over the different types of missings in the dataset.
In the later inferential steps, the problem is to get results valid with respect to a model. One needs to be able to look through the data on the true processes behind, without being distracted by artifacts stemming from missing values. For the moment, how analyses react to the presence of missings depend on the value of the na.action option. If this is set to na.omit (the default), then cases with missing values on any variable are completely ignored by the analysis. If this is set to na.pass, then some of the following applies.
The policy on how a missing value is to be introduced into the analysis depends on the purpose of the analysis, the type of analysis and the model behind. With respect to this issue this package and probabily the whole science of compositional data analysis is still very preliminary.
The four philosophies work with different approaches to these problems:

`rplus` For positive real vectors, one can either identify BDL with a true 0 or impute a value relative to the detection limit, with a function like `zeroreplace`. A structural zero can either be seen as a true zero or as a MAR value.

`rcomp` **and** `acomp` For these relative geometries, a true zero is an alien. Thus a BDL is nothing else but a small unkown value. We could either decide to replace the value by an imputation, or go through the whole analysis keeping this lack of information in mind. The main problem of imputation is that by closing to 1, the absolute value of the detection limit is lost, and the detection limit can correspond to very different portions. Raw differences between *all, observed or missed,* components (the ground of the rcomp geometry) are completely distorted by the replacement. Contrarily, log-ratios between observed components do not change but ratios between missed components dramatically depend on the replacement, e.g. typically the content of gold is some orders of magnitude smaller than the contend of silver even around a gold deposit, but far away from the deposit they both might be far under detection limit, leading to a ratio of 1, just because nothing was observed. SZ in compositions might be either seen as defining two sub-populations, one fully defined and one where only a subcomposition is defined. But SZ can also very much be like an MAR, if only a subcomposition is measured. Thus, in general we can simply understand that only a subcomposition is available, i.e. a projection of the true value onto a sub-space: for each observation, this sub-space might be different. For MAR values, this approach is stricly valid, and yields unbiased estimations (because these projections are stochastically independent of the observed phenomenon). For MNAR values, the projections depend on the actual value, which strictly speaking yields biased estimations.

`aplus` Imputation takes place by simple replacement of the value. However this can lead to a dramatic change of ratios and should thus be used only with extra care, by the same reasons explained before.

More information on how missings are actually processed can be found in the help files of each individual functions.

Value

A logical vector or matrix with the same shape as x stating wether or not the value is of the given type of missing.

Author(s)

K.Gerald v.d. Boogaart `http://www.stat.boogaart.de`, Raimon Tolosana Delgado, Matevz Bren

References

Boogaart, K.G. v.d., R. Tolosana-Delgado, M. Bren (2006) Concepts for handling of zeros and missing values in compositional data, in E. Pirard (ed.) (2006)Proccedings of the IAMG'2006 Annual Conference on "Quantitative Geology from multiple sources", September 2006, Liege, Belgium, S07-01, 4pages, `http://stat.boogaart.de/Publications/iamg06_s07_01.pdf`, ISBN: 978-2-9600644-0-7

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

Aitchison, J, C. Barcel'o-Vidal, J.J. Egozcue, V. Pawlowsky-Glahn (2002) A consise guide to the algebraic geometric structure of the simplex, the sample space for compositional data analysis, *Terra Nostra*, Schriften der Alfred Wegener-Stiftung, 03/2003

Billheimer, D., P. Guttorp, W.F. and Fagan (2001) Statistical interpretation of species composition,

Journal of the American Statistical Association, **96** (456), 1205-1214

Mart\'in-Fern\'andez, J.A., C. Barcel\'o-Vidal, and V. Pawlowsky-Glahn (2003) Dealing With Zeros and Missing Values in Compositional Data Sets Using Nonparametric Imputation. *Mathematical Geology*, **35**(3) 253-278

See Also

compositions-package, missingsInCompositions, robustnessInCompositions, outliersInCompositions, zeroreplace, rmult, ilr, mean.acomp, acomp, plot.acomp

Examples

```
require(compositions)      # load library
data(SimulatedAmounts)     # load data sa.lognormals
dat <- acomp(sa.missings)
dat
var(dat)
mean(dat)
plot(dat)
boxplot(dat)
barplot(dat)
```

---

missingProjector *Returns a projector the the observed space in case of missings.*

Description

Returns projectors on the observed subspace in the presence of missings.

Usage

```
missingProjector(x,...,by="s")
## S3 method for class 'acomp'
missingProjector(x,has=is.NMV(x),...,by="s")
## S3 method for class 'aplus'
missingProjector(x,has=is.NMV(x),...,by="s")
## S3 method for class 'rcomp'
missingProjector(x,has=!(is.MAR(x)|is.MNAR(x)),...,by="s")
## S3 method for class 'rplus'
missingProjector(x,has=!(is.MAR(x)|is.MNAR(x)),...,by="s")
```

Arguments

x	a dataset or object of the given class
has	a boolean matrix of the same size indicating nonmissing values

...	additional arguments for generic purpose only
by	the name of the dataset dimension on has for tensorial computation with tensorA package

Details

See the references for details on that function.

Value

A dataset of N square matrices of dimension DxD (with N and D respectively equal to the number of rows and columns in x). Each of these matrices gives the projection of a data row onto its observed sub-space.
The function sumMissingProjector takes all these matrices and sums them, generating a "summary" of observed sub-spaces. This matrix is useful to obtain estimates of the mean (and variance, in the future) still unbiased in the presence of lost values (only of type MAR, stricly-speaking, but anyway useful for any type of missing value, when used with care).

Author(s)

K.G.van den Boogaart

References

Boogaart, K.G. v.d. (2006) Concepts for handling of zeros and missing values in compositional data, in E. Pirard (ed.) (2006)Proccedings of the IAMG'2006 Annual Conference on "Quantitative Geology from multiple sources", September 2006, Liege, Belgium, S07-01, 4pages, http://stat.boogaart.de/Publications/iamg06_s07_01.pdf

See Also

missingsInCompositions

Examples

```
data(SimulatedAmounts)
x <- acomp(sa.lognormals)
xnew <- simulateMissings(x,dl=0.05,MAR=0.05,MNAR=0.05,SZ=0.05)
xnew
plot(missingSummary(xnew))

missingProjector(acomp(xnew))
missingProjector(rcomp(xnew))
missingProjector(aplus(xnew))
missingProjector(rplus(xnew))
```

missingsummary *Classify and summarize missing values in a dataset*

Description

Routines classifies codes of missing valuesas numbers in objects of the compositions package.

Usage

```
missingSummary(x,..., vlabs = colnames(x),
              mc=attr(x,"missingClassifier"),
              values=eval(formals(missingType)$values))
missingType(x,..., mc=attr(x,"missingClassifier"),
              values=c("NMV", "BDL", "MAR", "MNAR", "SZ", "Err"))
```

Arguments

x	a dataset which might contain missings
...	additional arguments for mc
mc	optionally in missingSummary, an alternate routine to be used instead of missingType
vlabs	labels for the variables
values	the names of the different types of missings. "Err" is a value that can not be classified e.g. Inf.

Details

The function mainly counts the various types of missing values.

Value

missingType returns a character vector/matrix with the same dimension and dimnames as x giving the type of every value.
missingSummary returns a table giving the number of missings of each type for each variable.

Author(s)

K. Gerald van den Boogaart

References

Boogaart, K.G., R. Tolosana-Delgado, M. Bren (2006) Concepts for the handling of zeros and missings in compositional data, *Proceedings of IAMG 2006, Liege*

See Also

compositions.missing

Examples

```
data(SimulatedAmounts)
x <- acomp(sa.lognormals)
xnew <- simulateMissings(x,dl=0.05,MAR=0.05,MNAR=0.05,SZ=0.05)
xnew
missingSummary(xnew)
```

`mix.Read` *Reads a data file in a mixR format*

Description

Reads a data file, which is formatted in a simple compositional file including the first row with title, the second with data labels and afterwards the matrix with the data itself. In the first column of the matrix are cases labels. This is the format used in the mixR package.

Usage

```
mix.Read(file,eps=1e-6)
```

Arguments

`file`	a file name
`eps`	the epsilon to be used for checking null values.

Details

The data files must have the adequate structure:

1 the first row with a title of the data set,

2 the second row with variables names

3 the data set in a matrix, rows as cases, variables in columns with the firs colum comprising cases labels.

A mixture object 'm' consists of `m$tit` the title, `m$mat` the matrix with the data, `m$sum` the value of the rows total, if constant and `m$sta` the status of the mixture object with values:

-2 - matrix contains negative elements,
-1 - zero row sum exists,
0 - matrix contains zero elements,
1 - matrix contains positive elements, rows with different row sum(s),
2 - matrix with constant row sum and
3 - closed mixture, the row sums are all equal to 1.

Value

A mixture object as a data frame with a title, row total, if constant, status (-2, -1, 0, 1, 2 or 3 – see above) and class attributes and the data matrix.

See Also

read.geoeas read.geoEAS read.table

Examples

```
## Not run:
  mix.Read("GLACIAL.DAT")
  mix.Read("ACTIVITY.DAT")

## End(Not run)
```

---

mvar *Metric summary statistics of real, amount or compositional data*

Description

Compute the metric variance, covariance, correlation or standard deviation.

Usage

```
mvar(x,...)
mcov(x,...)
mcor(x,...)
msd(x,...)
## Default S3 method:
mvar(x,y=NULL,...)
## Default S3 method:
mcov(x,y=x,...)
## Default S3 method:
mcor(x,y,...)
## Default S3 method:
msd(x,y=NULL,...)
```

Arguments

x	a dataset, eventually of amounts or compositions
y	a second dataset, eventually of amounts or compositions
...	further arguments to stats::var or stats::cov. Typically a robust=TRUE argument. e.g. use

Details

The metric variance (mvar) is defined by the trace of the variance in the natural geometry of the data, or also by the generalized variance in natural geometry. The natural geometry is equivalently given by the cdt or idt transforms.

The metric standard deviation (msd) is not the square root of the metric variance, but the square root of the mean of the eigenvalues of the variance matrix. In this way it can be interpreted in units of the original natural geometry, as the radius of a sperical ball around the mean with the same volume as the 1-sigma ellipsoid of the data set.

The metric covariance (mvar) is the sum over the absolute singular values of the covariance of two datasets in their respective geometries. It is always positive. The metric covariance of a dataset with itself is its metric variance. The interpretation of a metric covariance is quite difficult, but useful in regression problems.

The metric correlation (mcor) is the metric covariance of the datasets in their natural geometry normalized to unit variance matrix. It is a number between 0 and the smaller dimension of both natural spaces. A number of 1 means perfect correlation in 1 dimension, but only partial correlations in higher dimensions.

Value

a scalar number, informing of the degree of variation/covariation of one/two datasets.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de, Raimon Tolosana-Delgado

References

Daunis-i-Estadella, J., J.J. Egozcue, and V. Pawlowsky-Glahn (2002) Least squares regression in the Simplex on the simplex, *Terra Nostra*, Schriften der Alfred Wegener-Stiftung, 03/2003

Pawlowsky-Glahn, V. and J.J. Egozcue (2001) Geometric approach to statistical analysis on the simplex. *SERRA* **15**(5), 384-398

See Also

var, cov, mean.acomp, acomp, rcomp, aplus, rplus

Examples

```
data(SimulatedAmounts)
mvar(acomp(sa.lognormals))
mvar(rcomp(sa.lognormals))
mvar(aplus(sa.lognormals))
```

```
mvar(rplus(sa.lognormals))

msd(acomp(sa.lognormals))
msd(rcomp(sa.lognormals))
msd(aplus(sa.lognormals))
msd(rplus(sa.lognormals))

mcov(acomp(sa.lognormals5[,1:3]),acomp(sa.lognormals5[,4:5]))
mcor(acomp(sa.lognormals5[,1:3]),acomp(sa.lognormals5[,4:5]))
mcov(rcomp(sa.lognormals5[,1:3]),rcomp(sa.lognormals5[,4:5]))
mcor(rcomp(sa.lognormals5[,1:3]),rcomp(sa.lognormals5[,4:5]))

mcov(aplus(sa.lognormals5[,1:3]),aplus(sa.lognormals5[,4:5]))
mcor(aplus(sa.lognormals5[,1:3]),aplus(sa.lognormals5[,4:5]))
mcov(rplus(sa.lognormals5[,1:3]),rplus(sa.lognormals5[,4:5]))
mcor(rplus(sa.lognormals5[,1:3]),rplus(sa.lognormals5[,4:5]))

mcov(acomp(sa.lognormals5[,1:3]),aplus(sa.lognormals5[,4:5]))
mcor(acomp(sa.lognormals5[,1:3]),aplus(sa.lognormals5[,4:5]))
```

`names` *The names of the parts*

Description

The `names` function provide a transparent way to access the names of the parts regardless of the shape of the dataset or data item.

Usage

```
## S3 method for class 'acomp'
names(x)
## S3 method for class 'rcomp'
names(x)
## S3 method for class 'aplus'
names(x)
## S3 method for class 'rplus'
names(x)
## S3 method for class 'rmult'
names(x)
## S3 method for class 'ccomp'
names(x)
## S3 replacement method for class 'acomp'
names(x) <- value
## S3 replacement method for class 'rcomp'
names(x) <- value
## S3 replacement method for class 'aplus'
names(x) <- value
```

```
## S3 replacement method for class 'rplus'
names(x) <- value
## S3 replacement method for class 'rmult'
names(x) <- value
## S3 replacement method for class 'ccomp'
names(x) <- value
```

Arguments

x	an amount/amount dataset
value	the new names of the parts

Value

a character vector giving the names of the parts

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

aplus

Examples

```
data(SimulatedAmounts)
tmp <- acomp(sa.lognormals)
names(tmp)
names(tmp) <- c("x","y","z")
tmp
```

---

norm *Vector space norm*

Description

Each of the considered space structures has an associated norm, which is computed for each element by these functions.

Usage

```
## Default S3 method:
norm(x,...)
## S3 method for class 'acomp'
norm(x,...)
## S3 method for class 'rcomp'
norm(x,...)
## S3 method for class 'aplus'
norm(x,...)
## S3 method for class 'rplus'
norm(x,...)
## S3 method for class 'rmult'
norm(x,...)
## S3 method for class 'rmult'
norm(x,...)
```

Arguments

x	a dataset or a single vector of some type
...	currently not used, intended to select a different norm rule in the future

Value

The norms of the given vectors. ATTENTON: `norm.matrix` is a wrapper around base::norm

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

normalize

Examples

```
data(SimulatedAmounts)
tmp <- acomp(sa.lognormals)
mvar(tmp)
sum(norm( tmp - mean(tmp) )^2)/(nrow(tmp)-1)
```

normalize *Normalize vectors to norm 1*

Description

Normalize vectors to norm 1.

Usage

```
normalize(x,...)
## Default S3 method:
normalize(x,...)
```

Arguments

x	a dataset or a single vector of some type
...	currently not used, intended to select a different norm in the future

Value

The vectors given, but normalized to norm 1.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

norm.rmult

Examples

```
data(SimulatedAmounts)
normalize(c(1,2,3))
normalize(acomp(c(1,2,3)))
norm(normalize(acomp(sa.groups)))
```

NormalTests *Compositional Goodness of fit test*

Description

Tests for several groups of additive lognormally distributed compositions.

Usage

```
acompNormalLocation.test(x, g=NULL, var.equal=FALSE, paired=FALSE,
                                R=ifelse(var.equal,999,0))
```

Arguments

x	a dataset of compositions (acomp) or a list of such
g	a factor grouping the data, not used if x is a list already. Alternatively, g can be a second compositional data set.
var.equal	a boolean telling wether the variance of the groups should be considered equal
paired	true if a paired test should be performed
R	number of replicates that should be used to compute p-values. 0 means comparing the likelihood statistic with the correponding asymptotic chisq-distribution.

Details

The tests are based on likelihood ratio statistics.

Value

A classical "htest" object

data.name	The name of the dataset as specified
method	a name for the test used
alternative	an empty string
replicates	a dataset of p-value distributions under the Null-Hypothesis got from nonparametric bootstrap
p.value	The p.value computed for this test

Missing Policy

Up to now the tests cannot handle missings.

Note

Do not trust the p-values obtained forcing var.equal=TRUE and R=0. This will include soon equivalent spread tests.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

See Also

fitDirichlet,rDirichlet, runif.acomp, rnorm.acomp,

Examples

```
x <- runif.acomp(100,4)
y <- runif.acomp(100,4)
acompNormalLocation.test(list(x,y))
```

---

oneOrDataset *Treating single compositions as one-row datasets*

Description

A dataset is converted to a data matrix. A single data item (i.e. a simple vector) is converted to a one-row data matrix.

Usage

```
oneOrDataset(W,B=NULL)
```

Arguments

W	a vector, matrix or dataframe
B	an optional second vector, matrix or data frame having the intended number of rows.

Value

A data matrix containing the same data as W. If W is a vector it is interpreded as a single row. If B is given and `length(dim(B))!= 2` and W is a vector, then W is repeated `nrow(B)` times.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

Examples

```
oneOrDataset(c(1,2,3))
oneOrDataset(c(1,2,3),matrix(1:12,nrow=4))
oneOrDataset(data.frame(matrix(1:12,nrow=4)))
```

`outlierclassifier` *Detect and classify compositional outliers.*

Description

Detects outliers and classifies them according to different possible explanations.

Usage

```
OutlierClassifier1(X,...)
## S3 method for class 'acomp'
OutlierClassifier1(X,...,alpha=0.05,
           type=c("best","all","type","outlier","grade"),goodOnly=NULL,
           corrected=TRUE,RedCorrected=FALSE,robust=TRUE)
```

Arguments

`X`	the dataset as an `acomp` object
`...`	further arguments to MahalanobisDist/gsi.mahOutlier
`alpha`	The confidence level for identifying outliers.
`type`	What type of classification should be used: best: Which single component would best explain the outlier. all: Give a binary coding specifying all components, which could explain the outlier. type: Is it a a normal observation `"ok"`, a single componentn outlier `"1"` or can it not be explained by a single wrong component "?". outlier: All outliers are marked as `"outlier"`, others are marked as `"ok"`. `"grade"`: Proven Outliers are marked as "outlier"s, suspected outliers, detected without correction of the p-value are reported as "extreme", the rest is reported as "ok".
`goodOnly`	an integer vector. Only the specified index of the dataset should be used for estimation of the outlier criteria. This parameter if only a small portion of the dataset is reliable.
`corrected`	logical. Literatur often proposed to compare the Mahalanobis distances with Chisq-Approximations of there distributions. However this does not correct for multiple testing. If corrected is true a correction for multiple testing is used. In any case we do not use the chisq-approximation, but a simulation based procedure to compute confidence bounds.
`RedCorrected`	logical. If an outlier is detected we can try to find out wether a single component would be sufficient to drop the outlier under the outlier detection limit. Since in this second case we only check a few outliers no second correction step applies as long as the number of outliers is not very high.
`robust`	A robustness description as define in `var.acomp`

Details

See outliersInCompositions for a comprehensive introduction into the outlier treatment in compositions.

See ClusterFinder1 for an alternative method to classify observations in the context of outliers.

Value

A factor classifying the observations in the dataset as "ok" or some type of outlier.

Note

The package **robustbase** is required for using the robust estimations.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

outlierplot, ClusterFinder1

Examples

```
## Not run:
tmp<-set.seed(1400)
A <- matrix(c(0.1,0.2,0.3,0.1),nrow=2)
Mvar <- 0.1*ilrvar2clr(A%*%t(A))
Mcenter <- acomp(c(1,2,1))
data(SimulatedAmounts)
datas <- list(data1=sa.outliers1,data2=sa.outliers2,data3=sa.outliers3,
              data4=sa.outliers4,data5=sa.outliers5,data6=sa.outliers6)
opar<-par(mfrow=c(2,3),pch=19,mar=c(3,2,2,1))
tmp<-mapply(function(x,y) {
outlierplot(x,type="scatter",class.type="grade");
  title(y)
},datas,names(datas))


par(mfrow=c(2,3),pch=19,mar=c(3,2,2,1))
tmp<-mapply(function(x,y) {
  myCls2 <- OutlierClassifier1(x,alpha=0.05,type="all",corrected=TRUE)
  outlierplot(x,type="scatter",classifier=OutlierClassifier1,class.type="best",
  Legend=legend(1,1,levels(myCls),xjust=1,col=colcode,pch=pchcode),
  pch=as.numeric(myCls2));
  legend(0,1,legend=levels(myCls2),pch=1:length(levels(myCls2)))
  title(y)
},datas,names(datas))

par(mfrow=c(2,3),pch=19,mar=c(3,2,2,1))
for( i in 1:length(datas) )
  outlierplot(datas[[i]],type="ecdf",main=names(datas)[i])
```

```
par(mfrow=c(2,3),pch=19,mar=c(3,2,2,1))
for( i in 1:length(datas) )
  outlierplot(datas[[i]],type="portion",main=names(datas)[i])
par(mfrow=c(2,3),pch=19,mar=c(3,2,2,1))
for( i in 1:length(datas) )
  outlierplot(datas[[i]],type="nout",main=names(datas)[i])
par(opar)

## End(Not run)
```

`outlierplot` *Plot various graphics to analyse outliers.*

Description

A collection of plots emphasing different aspects of possible outliers.

Usage

```
outlierplot(X,...)
## S3 method for class 'acomp'
outlierplot(X,colcode=colorsForOutliers1,
  pchcode=pchForOutliers1,
  type=c("scatter","biplot","dendrogram","ecdf","portion","nout","distdist"),
  legend.position,pch=19,...,clusterMethod="ward",
  myCls=classifier(X,alpha=alpha,type=class.type,corrected=corrected),
  classifier=OutlierClassifier1,
  alpha=0.05,
  class.type="best",
  Legend,pow=1,
  main=paste(deparse(substitute(X))),
  corrected=TRUE,robust=TRUE,princomp.robust=FALSE,
                              mahRange=exp(c(-5,5))^pow,
                              flagColor="red",
                              meanColor="blue",
                              grayColor="gray40",
                              goodColor="green",
                              mahalanobisLabel="Mahalanobis Distance"
                              )
```

Arguments

`X`	The dataset as an `acomp` object
`colcode`	A color palette for factor given by the `myCls`, or function to create it from the factor. Use `colorForOutliers2` if `class.method="all"` is used.
`pchcode`	A function to create a plot character palette for the factor returned by the `myCls` call

`type` The type of plot to be produced. See details for more precise definitions.

`legend.position`
The location of the legend. Must!!! be given to draw a classical legend.

`pch` A default plotting char

`...` Further arguments to the used plotting function

`clusterMethod` The clustering method for `hclust` based outlier grouping.

`myCls` A factor presenting the groups of outliers

`classifier` The routine to create a factor presenting the groups of outliers heuristically. It is only used in the default argument to `myCls`.

`alpha` The confidence level to be used for outlier classification tests

`class.type` The type of classification that should be generated by `classifier`

`Legend` The content will be substituted and stored as list entry legend in the result of the function. It can than be evaluated to actually create a seperate legend on another device (e.g. for publications).

`pow` The power of Mahalanobis distances to be used.

`main` The title of the graphic

`corrected` Literature typically proposes to compare the Mahalanobis distances with the distribution of a random Mahalanobis distance. However it would be needed to correct this for (dependent) multiple testing, since we always test the whole dataset, which means comparing against the distribution of the maximum Mahalanobis distance. This argument switches to this second behavior, giving less outliers.

`robust` A robustness description as define in `robustnessInCompositions`

`princomp.robust`
Either a logical determining wether or not the principal component analysis should be done robustly or a principal component object for the dataset.

`mahRange` The range of Mahalanobis distances displayed. This is fixed to make views comparable among datasets. However if the preset default is not enough a warning is issued and a red mark is drawn in the plot

`flagColor` The color to draw critical situations.

`meanColor` The color to draw typical curves.

`goodColor` The color to draw confidence bounds.

`grayColor` The color to draw less important things.

`mahalanobisLabel`
The axis label to be used for axes displaying Mahalanobis distances.

Details

See outliersInCompositions for a comprehensive introduction into the outlier treatment in compositions.

`type="scatter"` Produces an appropriate standard plot such as a tenary diagram with the outliers marked by there codes according to the given classifier and colorcoding and pch coding. This shows the actual values of the identified outliers.

`type="biplot"` Creates a biplot based on a nonrobust principal component analysis showing the outliers classified through outliers in the given color scheme. We use the nonrobust principal component analyis since it rotates according to a good visibility of the extreme values.
This shows the position of the outliers in the usual principal components analysis. However note that a `coloredBiplot` is used rather than the usual one.

`type="dendrogram"` Shows a dendrogram based on robust Mahalanobis distance based hierachical clustering, where the observations are labeled with the identified outlier classes.
This plot can be used to see how good different categories of outliers cluster.

`type="ecdf"` This plot provides a cummulated distribution function of the Mahalanobis distances along with an expeced curve and a lower confidence limit. The empirical cdf is plotted in the default color. The expected cdf is displayed in `meanColor`. The `alpha`-quantile – i.e. a lower prediction bound – for the cdf is given in goodColor. A line in `grayColor` show the minium portion of observations above some limit to be outliers, based on the portion of observations necessary to move down to make the empirical distribution function get above its lower prediction limit under the assumption of normality.
This plot shows the basic construction for the minimal number of outlier computation done in `type="portion"`.

`type="portion"` This plot focusses on numbers of outliers. The horizontal axis give Mahalanobis distances and the vertical axis number of observations. In `meanColor` we see a curve of an estimated number of outliers above some limit, generated by estimating the portion of outliers with a Mahalanobis distance over the given limit by max(0,1-ecdf/cdf). The minimum number of outliers is computed by replacing cdf by its lower confidence limit and displayed in `goodColor`. The Mahalanobis distances of the individual data points are added as a stacked `stripchart`, such that the influence of individual observations can be seen.
The true problem of outlier detection is to detect "near" outliers. Near outliers are outliers so near to the dataset that they could well be extrem observation. These near outliers would provide no problem unless they are not many showing up in groups. Graphic allows at least to count them and to show there probable Mahalanobis distance such, however it still does not allow to conclude that an individual observation is an outlier. However still the outlier candidates can be identified comparing their mahalanobis distance (returned by the plot as`$mahalanobis`) with a cutoff inferred from this graphic.

`type="nout"` This is a simplification of the previous plot simply providing the number of outliers over a given limit.

`type="distdist"` Plots a scatterplot of the the classical and robust Mahalanobis distance with the given classification for colors and plot symbols. Furthermore it plots a horizontal line giving the 0.95-Quantil of the distribution of the maximum robust Mahalanobis distance of normally distributed dataset.

Value

a list respresenting the criteria computed to create the plots. The content of the list depends on the plotting type selected.

Note

The package **robustbase** is required for using the robust estimations.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

OutlierClassifier1, ClusterFinder1

Examples

```
## Not run:
data(SimulatedAmounts)
outlierplot(acomp(sa.outliers5))

datas <- list(data1=sa.outliers1,data2=sa.outliers2,data3=sa.outliers3,
                data4=sa.outliers4,data5=sa.outliers5,data6=sa.outliers6)

opar<-par(mfrow=c(2,3),pch=19,mar=c(3,2,2,1))
tmp<-mapply(function(x,y) {
outlierplot(x,type="scatter",class.type="grade");
  title(y)
},datas,names(datas))


par(mfrow=c(2,3),pch=19,mar=c(3,2,2,1))
tmp<-mapply(function(x,y) {
  myCls2 <- OutlierClassifier1(x,alpha=0.05,type="all",corrected=TRUE)
  outlierplot(x,type="scatter",classifier=OutlierClassifier1,class.type="best",
  Legend=legend(1,1,levels(myCls),xjust=1,col=colcode,pch=pchcode),
  pch=as.numeric(myCls2));
  legend(0,1,legend=levels(myCls2),pch=1:length(levels(myCls2)))
  title(y)
},datas,names(datas))
# To slow
par(mfrow=c(2,3),pch=19,mar=c(3,2,2,1))
for( i in 1:length(datas) )
  outlierplot(datas[[i]],type="ecdf",main=names(datas)[i])
par(mfrow=c(2,3),pch=19,mar=c(3,2,2,1))
for( i in 1:length(datas) )
  outlierplot(datas[[i]],type="portion",main=names(datas)[i])
par(mfrow=c(2,3),pch=19,mar=c(3,2,2,1))
for( i in 1:length(datas) )
  outlierplot(datas[[i]],type="nout",main=names(datas)[i])
for( i in 1:length(datas) )
  outlierplot(datas[[i]],type="distdist",main=names(datas)[i])
par(opar)


## End(Not run)
```

outliersInCompositions *Analysing outliers in compositions.*

Description

The Philosophy behind outlier treatment in library(compositions).

Details

Outliers are omnipresent in all kinds of data analysis. To avoid catastrophic misinterpreations robust statistics has developed some methods to avoid the distracting influence of the outliers. The introduction of robust methods into the compositions package is described in robustnessInCompositions.

However sometimes we are interested directly in the analysis of outliers. The central philosophy of the the outlier classification subsystem in compositions is that outlier are in most cases not simply erroneous observations, but rather products of some systematic anomality. This can e.g. be an error in an individual component, a secondary process or a minor undetected but different subpopulation. The package provides various concepts to investigate possible reasons for outliers in compositional datasets.

Proven Outliers The package relies on an additive–lognormal reference distribution in the simplex (and the correponding normal distribution in each other scale). The central tool for the detection of outliers is the Mahalanobis distance of the observation from a robustly estimated center based on a robustly estimated covariance. The robust estimation can be influenced by the given robust attributes. An outlier is considered as proven if its Mahalanobis distance is larger that the (1-alpha) quantile of the distribution of the maximum Mahalanobis distance of a dataset of the same size with a corresponding (additive)(log)normal distribution. This relies heavily on the presumption that the robust estimation is invariant under linear transformation, but make no assumptions about the actually used robust estimation method. The corresponding distributions are thus only defined with respect to a specific implementation of the robust estimation algorithm. See `OutlierClassifier1(...,type="outlier")`, `outlierplot(...,type=c("scatter","biplot"),class.type="outlier")`, `qMaxMahalanobis(...)`.

Extrem Values / Possible outliers Some cases of the dataset might have unusually high Mahalanobis distances, e.g. such that we would expect the probility of a random case to have such a value or higher might be below alpha. In Literature these cases are often rendered as outliers, because this level is approximated by the correponding chisq-based criterion proposed. However we consider these only as extrem values, but however provide tools to detect and plot them. See `OutlierClassifier1(...,type="grade")`, `outlierplot(...,type=c("scatter","biplot"),class.t`
`qEmpiricalMahalanobis(...)`

Single Component Outliers Some Outliers can be explained by a single component, e.g. because this single measurement error was wrong. These sort of outliers is detected when we reduce the dataset to a subcomposition with one component less and realise that our former outlier is now a fairly normal member of the dataset, maybe not even extrem. Thus a outlier is considered as as single component outlier, when it does not appear extrem in any of the subcompositions with one component less. For other outliers we can prove that they are still extrem for

all subcomposition with one component removed. Thus these have to be as multicomponent outliers, that can not be explained by a single measurment error. For remaining single component outliers, we can ask which component is able to explain the outlying character. See `OutlierClassifier1(...,type=c("best","type","all"))`.

Counting hidden outliers If outliers are not outlying far enough to be detected by the test for outlyingness are only at first sight harmless. One outlier is within the reasonable bounds of what a normal distribution could have delivered should not harm the analysis and might not even detectable in any way. However if there is more than one they could akt together to disrupt our analysis and more interestingly there might be some joint reason, which than might make them an interesting object of investigation in themselfs. Thus the package provides methods (e.g. `outlierplot(...,type="portions")`), to prove the existence of such outliers, to give a lower bound for there number and to provide us with suspects, with an associated outlyingness probability. See `outlierplot(...,type="portions")`, `outlierplot(...,type="nout")`, `pQuantileMahalanobis(...)`

Finding atypical subpopulations When we assume smaller subpopulation we need a tool finding these clusters. However usual cluster analysis tends to ignore the subgroups, split the main mass and then associate the subgroups prematurely to the next part of the main mass. For this task we have developed special tools to find clusters of atypical populations clearly inducing secondary modes, without ripping apart the central nonoutlying mass. See `ClusterFinder1`.

Identifying multiple distracting processes Outliers that are not due to a seperate subpopulation or due to a single component error, might still belong together for beeing influenced by the same secondary process distorting the composition to a different degrees. Out proposal is to cluster the direction of the outliers from the center, e.g. by a command like: `take<-OutlierClassifier1(data,type="grade hc<-hclust(dist(normalize(acomp(scale(data)[take,]))),method="compact")` and to plot by a command like: `plot(hc)` and `plot(acomp(data[take,]),col=cutree(hc,1.5))`

With these tools we hope to provide a systematic approach to identify various types of outliers in a exploratory analysis.

Note

The package **robustbase** is required for using the robust estimations and the outlier subsystem of compositions. To simplify installation it is not listed as required, but it will be loaded, whenever any sort of outlierdetection or robust estimation is used.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

K. Gerald van den Boogaart, Raimon Tolosana-Delgado, Matevz-Bren (2009) Robustness, classification and visualization of outliers in compositional data, in prep.

See Also

compositions-package, missingsInCompositions, robustnessInCompositions, outliersInCompositions, `outlierplot`, `OutlierClassifier1`, `ClusterFinder1`

Examples

```
## Not run:
# To slow
tmp<-set.seed(1400)
A <- matrix(c(0.1,0.2,0.3,0.1),nrow=2)
Mvar <- 0.1*ilrvar2clr(A%*%t(A))
Mcenter <- acomp(c(1,2,1))
typicalData <- rnorm.acomp(100,Mcenter,Mvar) # main population
colnames(typicalData)<-c("A","B","C")
data1 <- acomp(rnorm.acomp(100,Mcenter,Mvar))
data2 <- acomp(rbind(typicalData+rbinom(100,1,p=0.1)*rnorm(100)*acomp(c(4,1,1))))
data3 <- acomp(rbind(typicalData,acomp(c(0.5,1.5,2))))
colnames(data3)<-colnames(typicalData)
tmp<-set.seed(30)
rcauchy.acomp <- function (n, mean, var){
    D <- gsi.getD(mean)-1
   perturbe(ilrInv(matrix(rnorm(n*D)/rep(rnorm(n),D), ncol = D) %*% chol(clrvar2ilr(var))), mean)
}
data4 <- acomp(rcauchy.acomp(100,acomp(c(1,2,1)),Mvar/4))
colnames(data4)<-colnames(typicalData)
data5 <- acomp(rbind(unclass(typicalData)+outer(rbinom(100,1,p=0.1)*runif(100),c(0.1,1,2))))
data6 <- acomp(rbind(typicalData,rnorm.acomp(20,acomp(c(4,4,1)),Mvar)))
datas <- list(data1=data1,data2=data2,data3=data3,data4=data4,data5=data5,data6=data6)
tmp <-c()
opar<-par(mfrow=c(2,3),pch=19,mar=c(3,2,2,1))
tmp<-mapply(function(x,y) {
outlierplot(x,type="scatter",class.type="grade");
  title(y)
},datas,names(datas))


par(mfrow=c(2,3),pch=19,mar=c(3,2,2,1))
tmp<-mapply(function(x,y) {
  myCls2 <- OutlierClassifier1(x,alpha=0.05,type="all",corrected=TRUE)
  outlierplot(x,type="scatter",classifier=OutlierClassifier1,class.type="best",
  Legend=legend(1,1,levels(myCls),xjust=1,col=colcode,pch=pchcode),
  pch=as.numeric(myCls2));
  legend(0,1,legend=levels(myCls2),pch=1:length(levels(myCls2)))
  title(y)
},datas,names(datas))

par(mfrow=c(2,3),pch=19,mar=c(3,2,2,1))
for( i in 1:length(datas) )
  outlierplot(datas[[i]],type="ecdf",main=names(datas)[i])
par(mfrow=c(2,3),pch=19,mar=c(3,2,2,1))
for( i in 1:length(datas) )
  outlierplot(datas[[i]],type="portion",main=names(datas)[i])
par(mfrow=c(2,3),pch=19,mar=c(3,2,2,1))
for( i in 1:length(datas) )
  outlierplot(datas[[i]],type="nout",main=names(datas)[i])
par(opar)
```

```
moreData <- acomp(rbind(data3,data5,data6))
take<-OutlierClassifier1(moreData,type="grade")!="ok"
hc<-hclust(dist(normalize(acomp(scale(moreData)[take,]))),method="complete")
plot(hc)
plot(acomp(moreData[take,]),col=cutree(hc,1.5))

## End(Not run)
```

`pairs` *Pairs plot method for compositions*

Description

Pairs plot function for compositions, allowing flexible representations.

Usage

```
 ## S3 method for class 'acomp'
pairs(x, labels, panel = vp.lrdensityplot, ...,
                         horInd = 1:ncol(x), verInd = 1:ncol(x),
                         lower.panel = panel, upper.panel = panel,
                         diag.panel = NULL, text.panel = textPanel,
                         label.pos = 0.5 + has.diag/3, line.main = 3,
                         cex.labels = NULL, font.labels = 1,
                         row1attop = TRUE, gap = 1, log = "")
 ## S3 method for class 'rcomp'
pairs(x, labels, panel = vp.diffdensityplot, ...,
                         horInd = 1:ncol(x), verInd = 1:ncol(x),
                         lower.panel = panel, upper.panel = panel,
                         diag.panel = NULL, text.panel = textPanel,
                         label.pos = 0.5 + has.diag/3, line.main = 3,
                         cex.labels = NULL, font.labels = 1,
                         row1attop = TRUE, gap = 1, log = "")
  vp.lrdensityplot(x, y, col=2,..., alpha = NULL)
  vp.diffdensityplot(x, y, col=2,..., alpha = NULL)
  vp.lrboxplot(x, y, ...)
 vp.kde2dplot(x, y, grid=TRUE, legpos="bottomright", colpalette=heat.colors,...)
```

Arguments

`x`	a dataset of a compositional class; or for the panel functions, a vector of row components
`y`	for the panel functions, a vector of column components
`...`	further graphical parameters passed (see `par`)
`labels`	the names of the parts
`panel`	common panel function to use for all off-diagonal plots

horInd	indices of columns of x to plot on the horizontal axis, defaults to all columns
verInd	indices of columns of x to plot on the vertical axis, defaults to all columns
lower.panel	panel function for the lower triangle of plots, defaults to the common panel
upper.panel	panel function for the uppper triangle of plots, defaults to the common panel
diag.panel	panel function for the diagonal of plots, defaults to text.panel
text.panel	panel function to write labels on the diagonal panels
label.pos	y position of labels in the text panel
line.main	if main is specified, line.main gives the line argument to mtext() which draws the title. You may want to specify oma when changing line.main
cex.labels	graphics parameters for the text panel
font.labels	graphics parameters for the text panel
row1attop	logical. Should the layout be matrix-like with row 1 at the top, or graph-like with row 1 at the bottom?
gap	distance between subplots, in margin lines
log	a character string indicating if logarithmic axes are to be used: see plot.default. Should not be used and left to the panel function to handle
col	color for density and histogram components of the panel vp.*density
alpha	alpha level for marking normality in the panels vp.*density; default to no mark
grid	should a unit-grid be added to each panel?
legpos	where should the legend be placed? to be given as x in graphics::legend
colpalette	which color palette is desired for the 2d density levels?

Details

The data is displayed in a matrix of plots, after the indications of a panel function. This is a simple implementation of pairs compositional methods, the real functionality is controlled by the panel functions.

The three panel functions included here can be used for generating either boxplots or histograms plus kernel density plots of all pairwise logratios (in acomp) or differences (in rcomp) of the components. In the cas of histograms, these can be colored or left black-and-white depending on the adjustment to normality, controlled by a shapiro.test and the alpha-level given. These panel functions serve also as examples of how to generate user defined panels.

Author(s)

Raimon Tolosana-Delgado

See Also

vp.boxplot, vp.logboxplot

Examples

```
data(SimulatedAmounts)
pairs(acomp(sa.lognormals))
pairs(rcomp(sa.lognormals))
```

`pairwiseplot` *Creates a paneled plot like pairs for two different datasets.*

Description

Creates a plot for each element of two lists or each column of each dataset against each of the second.

Usage

```
pairwisePlot(X,Y,...)
## Default S3 method:
pairwisePlot(X,Y=X,...,
                xlab=deparse(substitute(X)),ylab=deparse(substitute(Y)),
                nm=c(length(Y),length(X)),panel=plot,
                add.line=FALSE, line.col=2,add.robust=FALSE,rob.col=4)
```

Arguments

`X`	a list, a data.frame, or a matrix representing the first set of things to be displayed.
`Y`	a list, a data.frame, or a matrix representing the second set of things to be displayed.
`...`	furter parameters to the panel function
`xlab`	The sequence of labels for the elements of X. Alternatively the labels can be given as colnames or names of X. This option takes precedence if specified.
`ylab`	The sequence of labels for the elements of Y. Alternatively the labels can be given as colnames or names of Y. This option takes precedence if specified.
`nm`	the parameter to be used in the call `par(mfrow=nm)`. If NULL no parameter is setted and a sequence of plots can be generated.
`panel`	The panel function to plot the individual panels. If the panel function admits a formula interface, it is called as `panel(y~x, xlab=xlab,ylab=ylab,...)`, otherwise as `panel(x, y,xlab=xlab,ylab=ylab,...)`. Thus the panel function must be capable of taking these arguments. It must also set up its own plot. There is no negotiation on coordinate system.
`add.line`	logical, to control the addition of a regression line in each panel
`line.col`	in case the regression line is added, which color should be used? defaults to red.
`add.robust`	logical, to control the addition of a robust regression line in each panel. Ignored if covariable is a factor. This is nowadays based on `lmrob`, but this can change in the future.
`rob.col`	in case the robust regression line is added, which color should be used? Defaults to blue.

Details

This is a light-weight convenience function to plot several aspects of one dataset against several aspects of another dataset. It is far more straight-forward than e.g. the `pairs` function and does not do any internal computation rather than organizing the names. Of course, the rows of the two data sets must be the same.

The current implementation may display a warning about the function `panel` dispatching methods for generic `plot`. It can be ignored without harm.

Optionally, classical and/or robust regression lines can be drawn, though only for non-factor covariables.

It may be convenient to use `par` capabilities to fit the device characteristics to the plot, in particular arguments `mar` and `oma`.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de, Raimon Tolosana-Delgado

References

Boogaart, K.G. v.d. , R. Tolosana (2008) Mixing Compositions and Other scales, Proceedings of CodaWork 08.

https://ima.udg.edu/Activitats/CoDaWork03/

https://ima.udg.edu/Activitats/CoDaWork05/

https://ima.udg.edu/Activitats/CoDaWork08/

See Also

`plot.aplus`, `balance`, `pwlrPlot`

Examples

```
X <- rnorm(100)
Y <- rnorm.acomp(100,acomp(c(A=1,B=1,C=1)),0.1*diag(3))+acomp(t(outer(c(0.2,0.3,0.4),X,"^")))

pairs(cbind(ilr(Y),X),panel=function(x,y,...) {points(x,y,...);abline(lm(y~x))})
pairs(cbind(balance(Y,~A/B/C),X),
         panel=function(x,y,...) {points(x,y,...);abline(lm(y~x))})
pairwisePlot(balance(Y,~A/B/C),X)
pairwisePlot(X,balance(Y,~A/B/C),
         panel=function(x,y,...) {plot(x,y,...);abline(lm(y~x))})
pairwisePlot(X,balance01(Y,~A/B/C))

# A function to extract a portion representation of subcompsitions
# with two elements:
subComps <- function(X,...,all=list(...)) {
  X <- oneOrDataset(X)
  nams <- sapply(all,function(x) paste(x[[2]],x[[3]],sep=","))
```

```
  val  <- sapply(all,function(x){
            a = X[,match(as.character(x[[2]]),colnames(X)) ]
            b = X[,match(as.character(x[[2]]),colnames(X)) ]
            c = X[,match(as.character(x[[3]]),colnames(X)) ]
            return(a/(b+c))
          })
  colnames(val)<-nams
  val
}

pairwisePlot(X,subComps(Y,A~B,A~C,B~C))

## using Hydrochemical data set as illustration of mixed possibilities
data(Hydrochem)
xc = acomp(Hydrochem[,c("Ca","Mg","Na","K")])
fk = Hydrochem$River
pH = -log10(Hydrochem$H)
covars = data.frame(pH, River=fk)
pairwisePlot(clr(xc), pH)
pairwisePlot(clr(xc), pH, col=fk)
pairwisePlot(pH, ilr(xc), add.line=TRUE)
pairwisePlot(covars, ilr(xc), add.line=TRUE, line.col="magenta")
pairwisePlot(clr(xc), covars, add.robust=TRUE)
```

`parametricMat` *Unique parametrisations for matrices.*

Description

Helper functions to parametrize positive semidefinite matrices in multivariate variogram models.

Usage

```
parametricRank1Mat(p)
parametricPosdefMat(p)
parameterRank1Mat(A)
parameterPosdefMat(A)
parametricRank1ClrMat(p)
parametricPosdefClrMat(p)
parameterRank1ClrMat(A)
parameterPosdefClrMat(A)
```

Arguments

`A`	a positiv definit matrix of the given type
`p`	a vector of parameters describing the matrix, as returned by the parameter functions.

Details

The rank 1 matrix is parametrised by the first eigenvector scaled by the square root of the eigenvalue. The positiv semidefinit matrix the entries of a upper right triangular matrix R with `t(R)%*%R==A`. The clr matrices are work with the parameters of the corresponding ilr matrix.

Value

A or p, depending on what is not given.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

vgram2lrvgram, CompLinModCoReg, vgmFit

Examples

```
parametricRank1Mat(c(0,0,2))
parametricPosdefMat(c(0,0,1,0,0,0))
parameterRank1Mat(matrix(1,nr=3,nc=3))
parameterPosdefMat(diag(5))
```

---

perturbe *Perturbation of compositions*

Description

The perturbation is the addition operation in the Aitchison geometry of the simplex.

Usage

```
perturbe(x,y)
## Methods for class "acomp"
## x + y
## x - y
##   - x
```

Arguments

x	compositions of class acomp
y	compositions of class acomp

Details

The perturbation is the basic addition operation of the Aitichson simplex as a vector space. It is defined by:

$$(x + y)_i = clo((x_i y_i)_i)_i$$

`perturbe` and `+` compute this operation. The only difference is that `+` checks the class of its argument, while `perturbe` does not check the type of the arguments and can thus directly be applied to a composition in any form (unclassed, acomp, rcomp).
The `-` operation is the inverse of the addition in the usual way and defined by:

$$(x - y)_i := clo((x_i / y_i)_i)_i$$

and as unary operation respectively as:

$$(-x)_i := clo((1/y_i)_i)_i$$

Value

An `acomp` vector or matrix.

Author(s)

K.Gerald v.d. Boogaart `http://www.stat.boogaart.de`, Raimon Tolosana-Delgado

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

See Also

`acomp`, `*.aplus`, `+.rplus`

Examples

```
tmp <- -acomp(1:3)
tmp + acomp(1:3)
```

`plot.acomp` *Ternary diagrams*

Description

Displaying compositions in ternary diagrams

Usage

```
 ## S3 method for class 'acomp'
plot(x,...,labels=names(x),
          aspanel=FALSE,id=FALSE,idlabs=NULL,idcol=2,center=FALSE,
          scale=FALSE,pca=FALSE,col.pca=par("col"),margin="acomp",
          add=FALSE,triangle=!add,col=par("col"),axes=FALSE,
          plotMissings=TRUE,
          lenMissingTck=0.05,colMissingTck="red",
          mp=~simpleMissingSubplot(c(0,1,0.95,1),
                                   missingInfo,c("NM","TM",cn)),
          robust=getOption("robust"))
 ## S3 method for class 'rcomp'
plot(x,...,labels=names(x),
          aspanel=FALSE,id=FALSE,idlabs=NULL,idcol=2,center=FALSE,
          scale=FALSE,pca=FALSE,col.pca=par("col"),margin="rcomp",
          add=FALSE,triangle=!add,col=par("col"),axes=FALSE
          ,plotMissings=TRUE,
          lenMissingTck=0.05,colMissingTck="red",
          mp=~simpleMissingSubplot(c(0,1,0.95,1),
                                    missingInfo,c("NM","TM",cn)),
          robust=getOption("robust"))
 ## S3 method for class 'ccomp'
plot(x,...)
```

Arguments

x	a dataset of a compositional class
...	further graphical parameters passed (see par)
margin	the type of marginalisation to be computed, when displaying the individual panels. Possible values are: "acomp", "rcomp" and any of the variable names/column numbers in the composition. If one of the columns is selected each panel displays a subcomposition given by the row part, the column part and the given part. If one of the classes is given the corresponding margin acompmargin or rcompmargin is used.
add	a logical indicating whether the information should just be added to an existing plot. If FALSE a new plot is created
triangle	a logical indicating whether the triangle should be drawn
col	the color to plot the data
labels	the names of the parts
aspanel	logical indicating that only a single panel should be drawn and not the whole plot. Internal use only
id	logical, if TRUE one can identify the points like with the identify command.
idlabs	a character vector providing the labels to be used with the identification, when id=TRUE

idcol	color of the idlabs labels
center	a logical indicating whether a the data should be centered prior to the plot. Centering is done in the choosen geometry. See scale
scale	a logical indicating whether a the data should be scaled prior to the plot. Scaling is done in the choosen geometry. See scale
pca	a logical indicating whether the first principal component should be displayed in the plot. Currently, the direction of the principal component of the displayed subcomposition is displayed as a line. In a future, the projected principal componenent of the whole dataset should be displayed.
col.pca	The color to draw the principal component.
axes	Either a logical wether to plot the axes, or numerical enumerating the axes sides to be used e.g. 1 for only plotting the lower axes, or a list of parameters to ternaryAxis.
plotMissings	logical indicating that missingness should be represented graphically. Componentes with one missing subcomponent in the plot are represented by tickmarks at the three axis. Components with two or three missing components are only represented in a special panel drawn according to the mp parameter if missings are present. Missings of type BDL (below detection limit) are always plotted, even if plotMissings is false, but in this case this fact is not specially marked. In rcomp geometry an actuall 0 in the data is never treated as missing.
lenMissingTck	length of the tick-marks to be plotted for missing values. If 0 no tickmarks are plotted. Negative lengths point outside. length 1 draws right through to the opposit corner. Missing ticks in acomp geometry are inclined showing the line of possible values in acomp geometry. Missingticks in rcomp-geometry are vertical to the axis representing the fact that only the other component is unkown. That these lines can leave the plot is one of the odd consequences of rcomp geometry.
colMissingTck	colors to draw the missing tick-marks. NULL means to take the colors specified for the observations.
mp	A formula providing a call to a function plotting informations on the missings. The call is evaluted in the environment of the panel plotting function and has access (among others) to: cn the names of the components in the current plot, x the dataset of the current plot, y the transformed dataset, (c60,s60) coordinates of the upper vertex of the triangle. missingInfo is a table giving the number of observations of the types NM=Non Missing, TM=Totally missing (i.e. at least two components of the subcomposition are missing), and the three single component missing possibilities for the three components.
robust	A robustness description. See robustnessInCompositions for details. The option is used for centering, scaling and principle components.

Details

The data is displayed in ternary diagrams. Thus, it does not work for two-part compositions. Compositions of three parts are displayed in a single ternary diagram. For compositions of more than three components, the data is arranged in a scatterplot matrix through the command pairs.
In this case, the third component in each of the panels is chosen according to setting of margin=.

Possible values of margin= are: "acomp", "rcomp" and any of the variable names/column numbers in the composition. If one of the columns is selected each panel displays a subcomposition given by the row part, the column part and the given part. If one of the classes is given the corresponding margin acompmargin or rcompmargin is used.
Ternary diagrams can be read in multiple ways. Each corner of the triangle corresponds to an extreme composition containing only the part displayed in that corner. Points on the edges correspond to compositions containing only the parts in the adjacent corners. The relative amounts are displayed by the distance to the opposite corner (so-called barycentric coordinates). The individual portions of any point can be infered by drawing a line through the investigated point, and parallel to the edge opposite to the corner of the part of interest. The portion of this part is constant along the line. Thus we can read it on the sides of the ternary diagram, where the line crosses its borders. Note that these isoPortionLines remain straight under an arbitrary perturbation.
ccomp ternary diagrams are always jittered to avoid overplotting.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de, Raimon Tolosana-Delgado

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

Aitchison, J, C. Barcel'o-Vidal, J.J. Egozcue, V. Pawlowsky-Glahn (2002) A consise guide to the algebraic geometric structure of the simplex, the sample space for compositional data analysis, *Terra Nostra*, Schriften der Alfred Wegener-Stiftung, 03/2003

Billheimer, D., P. Guttorp, W.F. and Fagan (2001) Statistical interpretation of species composition, *Journal of the American Statistical Association*, **96** (456), 1205-1214

Pawlowsky-Glahn, V. and J.J. Egozcue (2001) Geometric approach to statistical analysis on the simplex. *SERRA* **15**(5), 384-398

https://ima.udg.edu/Activitats/CoDaWork03/

https://ima.udg.edu/Activitats/CoDaWork05/

See Also

plot.aplus, plot3D (for 3D plot), kingTetrahedron (for 3D-plot model export), qqnorm.acomp,boxplot.acomp

Examples

```
data(SimulatedAmounts)
plot(acomp(sa.lognormals))
plot(acomp(sa.lognormals),axes=TRUE)
plot(rcomp(sa.lognormals))
plot(rcomp(sa.lognormals5))
```

```
plot(acomp(sa.lognormals5),pca=TRUE,col.pca="red")
plot(rcomp(sa.lognormals5),pca=TRUE,col.pca="red",axes=TRUE)
```

`plot.aplus` *Displaying amounts in scatterplots*

Description

This function displays multivariate unclosed amout datasets classes "aplus" and "rplus" in a way respecting the choosen geometry eventually in log scale.

Usage

```
  ## S3 method for class 'aplus'
plot(x,...,labels=colnames(X),cn=colnames(X),
                     aspanel=FALSE,id=FALSE,idlabs=NULL,idcol=2,
                     center=FALSE,scale=FALSE,pca=FALSE,col.pca=par("col"),
                     add=FALSE,logscale=TRUE,xlim=NULL,ylim=xlim,
                     col=par("col"),plotMissings=TRUE,
                     lenMissingTck=0.05,colMissingTck="red",
                     mp=~simpleMissingSubplot(missingPlotRect,missingInfo,
                                              c("NM","TM",cn)),
                     robust=getOption("robust"))
  ## S3 method for class 'rplus'
plot(x,...,labels=colnames(X),cn=colnames(X),
                     aspanel=FALSE,id=FALSE,idlabs=NULL,idcol=2,
                     center=FALSE,scale=FALSE,pca=FALSE,col.pca=par("col"),
                     add=FALSE,logscale=FALSE,
                     xlim=NULL,
                     ylim=xlim,col=par("col"),plotMissings=TRUE,
                     lenMissingTck=0.05,colMissingTck="red",
                     mp=~simpleMissingSubplot(missingPlotRect,missingInfo,
                                              c("NM","TM",cn)),
                     robust=getOption("robust"))
  ## S3 method for class 'rmult'
plot(x,...,labels=colnames(X),cn=colnames(X),
                     aspanel=FALSE,id=FALSE,idlabs=NULL,idcol=2,
                     center=FALSE,scale=FALSE,pca=FALSE,col.pca=par("col"),
                     add=FALSE,logscale=FALSE,col=par("col"),
                     robust=getOption("robust"))
```

Arguments

x	a dataset with class aplus, rplus or rmult
...	further graphical parameters passed (see par)

add	a logical indicating whether the information should just be added to an existing plot. If FALSE, a new plot is created
col	the color to plot the data
plotMissings	logical indicating that missingness should be represented graphically. Componentes with one missing subcomponent in the plot are represented by tickmarks at the two axis. Cases with two missing components are only represented in a special panel drawn according to the mp parameter if missings are present. Missings of type BDL (below detection limit) are always plotted in nonlogaritmic plots, even if plotMissings is false, but in this case this fact is not specially marked.
lenMissingTck	length of the tick-marks (in portion of the plotting region) to be plotted for missing values. If 0 no tickmarks are plotted. Negative lengths point outside of the plot. A length of 1 runs right through the whole plot.
colMissingTck	colors to draw the missing tick-marks. NULL means to take the colors specified for the observations.
mp	A formula providing a call to a function plotting informations on the missings. The call is evaluted in the environment of the panel plotting function and has access (among others) to: cn the names of the components in the current plot, x the dataset of the current plot, missingInfo is a table giving the number of observations of the types NM=Non Missing, TM=Totally missing (i.e. two components of the subcomposition are missing), and the two single component missing possibilities.
labels	the labels for names of the parts
cn	the names of the parts to be used in a single panel. Internal use only
aspanel	logical indicating that only a single panel should be drawn and not the whole plot. Internal use only
id	a logical. If TRUE one can identify the points like with the identify command
idlabs	A character vector providing the labels to be used with the identification, when id=TRUE
idcol	color of the idlabs labels
center	a logical indicating whether the data should be centered prior to the plot. Centering is done in the chosen geometry. See scale
scale	a logical indicating whether the data should be scaled prior to the plot. Scaling is done in the chosen geometry. See scale
pca	a logical indicating whether the first principal component should be displayed in the plot. Currently, the direction of the principal component of the displayed subcomposition is displayed as a line. In a future, the projected principal componenent of the whole dataset should be displayed.
col.pca	the color to draw the principal component.
logscale	logical indicating whether a log scale should be used
xlim	2xncol(x)-matrix giving the xlims for the columns of x
ylim	2xncol(x)-matrix giving the ylims for the columns of x
robust	A robustness description. See robustnessInCompositions for details. The option is used for centering, scaling and principle components.

Details

TO DO: fix pca bug

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

plot.aplus, qqnorm.acomp,boxplot.acomp

Examples

```
data(SimulatedAmounts)
plot(aplus(sa.lognormals))
plot(rplus(sa.lognormals))
plot(aplus(sa.lognormals5))
plot(rplus(sa.lognormals5))
```

---

plot3D *plot in 3D based on rgl*

Description

3-dimensional plots, which can be rotated and zoomed in/out

Usage

```
plot3D(x,...)
## Default S3 method:
plot3D(x,...,add=FALSE,bbox=TRUE,axes=FALSE,
           cex=1,size=cex,col=1)
```

Arguments

x	an object to be plotted, e.g. a data frame or a data matrix
...	additional plotting parameters as described in rgl::material3d
add	logical, adding or new plot
bbox	logical, whether to add a bounding box
axes	logical, whether to plot an axes of coordinates
cex	size of the plotting symbol
size	size of the plotting symbol, only size or cex should be used
col	the color used for dots, defaults to black.

Details

The function provides a generic interface for 3-dimensional plotting in analogy to the 2d-plotting interface of plot, using rgl package.

Value

the 3D plotting coordinates of the objects displayed, returned invisibly

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

rgl::points3d, graphics::plot, plot3D.rmult,

plot3D.acomp,plot3D.rcomp, plot3D.aplus,plot3D.rplus

Examples

```
x <- cbind(rnorm(10),rnorm(10),rnorm(10))
data(SimulatedAmounts)
if(requireNamespace("rgl", quietly = TRUE)) {
  plot3D(x)
  plot3D(sa.lognormals,cex=4,col=1:nrow(sa.lognormals))
} ## this function requires package 'rgl'
```

---

plot3Dacomp *3D-plot of compositional data*

Description

3D-plot of compositional data. The plot is mainly an exploratory tool, not intended for exact display of data.

Usage

```
 ## S3 method for class 'acomp'
plot3D(x, parts=1:min(ncol(X),4),...,
             lwd=2, axis.col="gray", add=FALSE, cex=2,
             vlabs=colnames(x), vlabs.col=axis.col, center=FALSE,
             scale=FALSE, log=FALSE, bbox=FALSE, axes=TRUE, size=cex,col=1)
## S3 method for class 'rcomp'
plot3D(x,parts=1:min(ncol(X),4),...,
             lwd=2,axis.col="gray",add=FALSE,cex=2,
             vlabs=colnames(x),vlabs.col=axis.col,center=FALSE,
             scale=FALSE,log=FALSE,bbox=FALSE,axes=TRUE,size=cex,col=1)
```

Arguments

x	an aplus object to be plotted
parts	a numeric xor character vector of length 3 coding the columns to be plotted
...	additional plotting parameters as described in rgl::material3d
add	logical, adding or new plot
cex	size of the plotting symbols
lwd	line width
axis.col	color of the axis
vlabs	the column names to be plotted, if missing defaults to the column names of the selected columns of X
vlabs.col	color of the labels
center	logical, should the data be centered
scale	logical, should the data be scaled
log	logical, indicating wether to plot in log scale
bbox	logical, whether to add a bounding box
axes	logical, whether plot a coordinate cross
size	size of the plotting symbols
col	the color used for dots, defaults to black.

Details

The routine behaves different when 3 or four components should be plotted. In case of four components:
If log is TRUE the data is plotted in ilr coordinates. This is the isometric view of the data.
If log is FALSE the data is plotted in ipt coordinates and a tetrahedron is plotted around it if coors == TRUE. This can be used to do a tetrahedron plot.
In case of three components:
If log is TRUE the data is plotted in clr coordinates. This can be used to visualize the clr plane.
If log is FALSE the data is plotted as is, showing the embedding of the three-part simplex in the three-dimensional space.
In all cases: If coors is true, coordinate arrows are plotted of length 1 in the origin of the space, except in the tetrahedron case.

Value

Called for its side effect of a 3D plot of an acomp object in an rgl plot. It invisibly returns the 3D plotting coordinates of the objects displayed

Note

The function kingTetrahedron provides an alternate way of tetrahedron plots, based on a more advanced viewer, which must be downloaded separately.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

kingTetrahedron

rgl::points3d, graphics::plot, plot3D, plot3D.rmult,

plot3D.rcomp, plot3D.aplus,plot3D.rplus

Examples

```
data(SimulatedAmounts)
if(requireNamespace("rgl", quietly = TRUE)) {
  plot3D(acomp(sa.lognormals5),1:3,col="green")
  plot3D(acomp(sa.lognormals5),1:3,log=TRUE,col="green")
  plot3D(acomp(sa.lognormals5),1:4,col="green")
  plot3D(acomp(sa.lognormals5),1:4,log=TRUE,col="green")
} ## this function requires package 'rgl'
```

---

plot3Daplus *3D-plot of positive data*

Description

3D-plot of positive data typically in log-log-log scale. The plot is mainly an exploratory tool, and not intended for exact display of data.

Usage

```
## S3 method for class 'aplus'
plot3D(x,parts=1:3,...,
                vlabs=NULL,add=FALSE,log=TRUE,bbox=FALSE,axes=TRUE,col=1)
```

Arguments

x	an aplus object to be plotted
parts	a numeric xor character vector of length 3 coding the columns to be plotted
...	additional plotting parameters as described in rgl::material3d
add	logical, adding or new plot
vlabs	the column names to be plotted, if missing defaults to the column names of the selected columns of X
log	logical, indicating wether to plot in log scale
bbox	logical, whether to add a bounding box
axes	logical, plot a coordinate system
col	the color used for dots, defaults to black.

Details

If log is TRUE the data is plotted in ilt coordinates. If coors is true, coordinate arrows are plotted of length 1 and in the (aplus-)mean of the dataset.
If log is FALSE the data is plotted with plot.rplus

Value

Called for its side effect of a 3D plot of an aplus object in an rgl plot. It invisibly returns the 3D plotting coordinates of the objects displayed

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

kingTetrahedron

rgl::points3d, graphics::plot, plot3D, plot3D.rmult,

plot3D.acomp,plot3D.rcomp, plot3D.rplus

Examples

```
data(SimulatedAmounts)
if(requireNamespace("rgl", quietly = TRUE)) {
  plot3D(aplus(sa.lognormals),size=2)
} ## this function requires package 'rgl'
```

---

plot3Drmult *plot in 3D based on rgl*

Description

3-dimensional plots, which can be rotated and zoomed in/out

Usage

```
## S3 method for class 'rmult'
plot3D(x,parts=1:3,...,
                center=FALSE,scale=FALSE,add=FALSE,axes=!add,
                cex=2,vlabs=colnames(x),size=cex,bbox=FALSE,col=1)
```

Arguments

x	an object to be plotted, e.g. a data frame or a data matrix
parts	the variables in the rmult object to be plotted
...	additional plotting parameters as described in rgl::material3d
center	logical, center the data? This might be necessary to stay within the openGL-arithmetic used in rgl.
scale	logical, scale the data? This might be necessary to stay within the openGL-arithmetic used in rgl.
add	logical, adding or new plot
bbox	logical, whether to add a bounding box
axes	logical, whether to plot a coordinate cross
cex	size of the plotting symbol (as expanding factor)
vlabs	labels for the variables
size	size of the plotting symbol, only size or cex should be used
col	the color used for dots, defaults to black.

Details

The function provides a generic interface for 3-dimensional plotting in analogy to the 2d-plotting interface of plot, using rgl package.

Value

the 3D plotting coordinates of the objects displayed, returned invisibly

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

kingTetrahedron

rgl::points3d, graphics::plot, plot3D,

plot3D.acomp,plot3D.rcomp, plot3D.aplus,plot3D.rplus

Examples

```
x <- cbind(rnorm(10),rnorm(10),rnorm(10))
data(SimulatedAmounts)
if(requireNamespace("rgl", quietly = TRUE)) {
  plot3D(x)
  plot3D(rmult(sa.lognormals),cex=4,col=1:nrow(sa.lognormals))
} ## this function requires package 'rgl'
```

plot3Drplus *plot in 3D based on rgl*

Description

3-dimensional plots, which can be rotated and zoomed in/out

Usage

```
## S3 method for class 'rplus'
plot3D(x,parts=1:3,...,vlabs=NULL,add=FALSE,bbox=FALSE,
                          cex=1,size=cex,axes=TRUE,col=1)
```

Arguments

x	an rplus object to be plotted
parts	the variables in the rplus object to be plotted
...	additional plotting parameters as described in rgl::material3d
vlabs	the labels used for the variable axes
add	logical, adding or new plot
bbox	logical, whether to add a bounding box
cex	size of the plotting symbol (as character expansion factor)
size	size of the plotting symbol, only size or cex should be used
axes	logical, whether to plot a coordinate cross
col	the color used for dots, defaults to black.

Details

The function plots rplus objects in a 3D coordinate system, in an rgl plot.

Value

the 3D plotting coordinates of the objects displayed, returned invisibly

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

kingTetrahedron

rgl::points3d, graphics::plot, plot3D, plot3D.rmult, plot3D.acomp,plot3D.rcomp, plot3D.aplus,plot3D

Examples

```
x <- cbind(rnorm(10),rnorm(10),rnorm(10))
data(SimulatedAmounts)
if(requireNamespace("rgl", quietly = TRUE)) {
  plot3D(rplus(exp(x)))
  plot3D(rplus(sa.lognormals),cex=4,col=1:nrow(sa.lognormals))
} ## this function requires package 'rgl'
```

---

plotlogratioVariogram *Empirical variograms for compositions*

Description

Plots a logratioVariogram.

Usage

```
## S3 method for class 'logratioVariogram'
plot(x,...,type="l",lrvg=NULL,
                fcols=2:length(lrvg),oma=c(4, 4, 4, 4),gap=0,ylim=NULL)
```

Arguments

x	The logratioVariogram created by logratioVariogram
...	further parameters for plot.default
type	as in plot.default
lrvg	a model function for a logratiovariogram or a list of several, to be added to the plot.
fcols	the colors for the different lrvg variograms
oma	The outer margin of the paneled plot
gap	The distance of the plot panals used to determin mar
ylim	The limits of the Y-axis. If zero it is automatically computed.

Details

see logratioVariogram

Value

Nothing.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

vgram2lrvgram, CompLinModCoReg

Examples

```
## Not run:
data(juraset)
X <- with(juraset,cbind(X,Y))
comp <- acomp(juraset,c("Cd","Cu","Pb","Co","Cr"))
lrv <- logratioVariogram(comp,X,maxdist=1,nbins=10)
fff <- CompLinModCoReg(~nugget()+sph(0.5)+R1*exp(0.7),comp)
fit <- vgmFit(lrv,fff)
fit
fff(1:3)
plot(lrv,lrvg=vgram2lrvgram(fit$vg))

## End(Not run)
```

---

plotmissingsummary *Plot a Missing Summary*

Description

Plots a missing summary as a barplot

Usage

```
## S3 method for class 'missingSummary'
plot(x,...,main="Missings",legend.text=TRUE,
                    col=c("gray","lightgray","yellow","red","white","magenta"))
 as.missingSummary(x,...)
```

Arguments

x	a missingSummary table with columns representing different types of missing
...	further graphical parameters to barplot
main	as in barplot
legend.text	as in barplot
col	as in barplot

Details

The different types of missings are drawn in quasi-self-understandable colors: normal gray for NMV, and lightgray as for BDL (since they contain semi-numeric information), yellow (slight warning) for MAR, red (serious warning) for MNAR, white (because they are non-existing) for SZ, and magenta for the strange case of errors.

Value

called for its side effect. The return value is not defined.

Author(s)

K.Gerald van den Boogaart

References

See compositions.missings for more details.

See Also

missingSummary

Examples

```
data(SimulatedAmounts)
x <- acomp(sa.lognormals)
xnew <- simulateMissings(x,dl=0.05,MAR=0.05,MNAR=0.05,SZ=0.05)
xnew
plot(missingSummary(xnew))
```

PogoJump *Honk Kong Pogo-Jumps Championship*

Description

Yat, yee, sam measurements (in meters) for the final jumps of the 1985 Honk Kong Pogo-Jumps Championship, 4 jumps of the 7 finalists.

Usage

```
data(PogoJump)
```

Details

The data consist of 28 cases: 4 jumps of the 7 finalists, and 4 variables: Yat, Yee, Sam measurements in meters, and finalist – 1 to 7.

Pogo-Jumps is similar to the triple jump except that the competitor is mounted on a pogo-stik. After a pogo-up towards the starting board the total jump distance achieved in three consecutive bounces, known as the yat, yee and sam, is recorded.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name HKPOGO.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison, J. (1982) The Statistical Analysis of Compositional Data, (Data 19) pp21.

`powerofpsdmatrix` *power transform of a matrix*

Description

Computes the power of a positive semi-definite symmetric matrix.

Usage

```
powerofpsdmatrix( M , p,...)
```

Arguments

`M`	a matrix, preferably symmetric
`p`	a single number giving the power
`...`	further arguments to the singular value decomposition

Details

for a symmetric matrix the computed result can actually be considered as a version of the given power of the matrix fullfilling the relation:

$$M^p M^q = M^{p+q}$$

The symmetry of the matrix is not checked.

Value

`U%*% D^p %*% t(P)` where the `UDP` is the singular value decomposition of M.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

Examples

```
data(SimulatedAmounts)
d <- ilr(sa.lognormals)
var( d %*% powerofpsdmatrix(var(d),-1/2)) # Unit matrix
```

princomp.acomp *Principal component analysis for Aitchison compositions*

Description

A principal component analysis is done in the Aitchison geometry (i.e. clr-transform) of the simplex. Some gimics simplify the interpretation of the computed components as compositional perturbations.

Usage

```
 ## S3 method for class 'acomp'
princomp(x,...,scores=TRUE,center=attr(covmat,"center"),
                           covmat=var(x,robust=robust,giveCenter=TRUE),
                           robust=getOption("robust"))
 ## S3 method for class 'princomp.acomp'
print(x,...)
 ## S3 method for class 'princomp.acomp'
plot(x,y=NULL,..., npcs=min(10,length(x$sdev)),
        type=c("screeplot","variance","biplot","loadings","relative"),
        main=NULL,scale.sdev=1)
 ## S3 method for class 'princomp.acomp'
predict(object,newdata,...)
```

Arguments

x	a acomp-dataset (in princomp) or a result from princomp.acomp
y	not used
scores	a logical indicating whether scores should be computed or not
npcs	the number of components to be drawn in the scree plot
type	type of the plot: "screeplot" is a lined screeplot, "variance" is a boxplot-like screeplot, "biplot" is a biplot, "loadings" displays the loadings as a barplot.acomp
scale.sdev	the multiple of sigma to use plotting the loadings
main	title of the plot
object	a fitted princomp.acomp object
newdata	another compositional dataset of class acomp
...	further arguments to pass to internally-called functions
covmat	provides the covariance matrix to be used for the principle component analysis
center	provides the be used for the computation of scores
robust	Gives the robustness type for the calculation of the covariance matrix. See robustnessInCompositions for details.

Details

As a metric euclidean space the Aitchison simplex has its own principal component analysis, that should be performed in terms of the covariance matrix and not in terms of the meaningless correlation matrix.
To aid the interpretation we added some extra functionality to a normal `princomp(clr(x))`. First of all the result contains as additional information the compositional representation of the returned vectors in the space of the data: the center as a composition `Center`, and the loadings in terms of a composition to perturbe with, either positively (`Loadings`) or negatively (`DownLoadings`). The Up- and DownLoadings are normalized to the number of parts in the simplex and not to one to simplify the interpretation. A value of about one means no change in the specific component. To avoid confusion the meaningless last principal component is removed.
The `plot` routine provides screeplots (`type = "s"`,`type= "v"`), biplots (`type = "b"`), plots of the effect of loadings (`type = "b"`) in `scale.sdev*sdev`-spread, and loadings of pairwise (log-)ratios (`type = "r"`).
The interpretation of a screeplot does not differ from ordinary screeplots. It shows the eigenvalues of the covariance matrix, which represent the portions of variance explained by the principal components.
The interpretation of the biplot strongly differs from a classical one. The relevant variables are not the arrows drawn (one for each component), but rather the links (i.e., the differences) between two arrow heads, which represents the log-ratio between the two components represented by the arrows.
The compositional loading plot is introduced with this package. The loadings of all component can be seen as an orthogonal basis in the space of clr-transformed data. These vectors are displayed by a barplot with their corresponding composition. For a better interpretation the total of these compositons is set to the number of parts in the composition, such that a portion of one means no effect. This is similar to (but not exactly the same as) a zero loading in a real principal component analysis.
The loadings plot can work in two different modes: if `scale.sdev` is set to `NA` it displays the composition beeing represented by the unit vector of loadings in the clr-transformed space. If `scale.sdev` is numeric we use this composition scaled by the standard deviation of the respective component.
The relative plot displays the `relativeLoadings` as a barplot. The deviation from a unit bar shows the effect of each principal component on the respective ratio.

Value

`princomp` gives an object of type `c("princomp.acomp","princomp")` with the following content:

`sdev`	the standard deviation of the principal components
`loadings`	the matrix of variable loadings (i.e., a matrix which columns contain the eigenvectors). This is of class `"loadings"`. The last eigenvector is removed since it should contain the irrelevant scaling.
`center`	the clr-transformed vector of means used to center the dataset
`Center`	the `acomp` vector of means used to center the dataset
`scale`	the scaling applied to each variable
`n.obs`	number of observations
`scores`	if `scores = TRUE`, the scores of the supplied data on the principal components. Scores are coordinates in a basis given by the principal components and thus not compositions
`call`	the matched call

na.action	not clearly understood
Loadings	compositions that represent a perturbation with the vectors represented by the loadings of each of the factors
DownLoadings	compositions that represent a perturbation with the inverse of the vectors represented by the loadings of each of the factors

predict returns a matrix of scores of the observations in the newdata dataset
. The other routines are mainly called for their side effect of plotting or printing and return the object x.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

Aitchison, J, C. Barcel'o-Vidal, J.J. Egozcue, V. Pawlowsky-Glahn (2002) A consise guide to the algebraic geometric structure of the simplex, the sample space for compositional data analysis, *Terra Nostra*, Schriften der Alfred Wegener-Stiftung, 03/2003

Aitchison, J. and M. Greenacre (2002) *Biplots for Compositional Data* Journal of the Royal Statistical Society, Series C (Applied Statistics) **51** (4) 375-392

https://ima.udg.edu/Activitats/CoDaWork03/

https://ima.udg.edu/Activitats/CoDaWork05/

See Also

clr,acomp, relativeLoadings princomp.aplus, princomp.rcomp, barplot.acomp, mean.acomp, var.acomp

Examples

```
data(SimulatedAmounts)
pc <- princomp(acomp(sa.lognormals5))
pc
summary(pc)
plot(pc)           #plot(pc,type="screeplot")
plot(pc,type="v")
plot(pc,type="biplot")
plot(pc,choice=c(1,3),type="biplot")
plot(pc,type="loadings")
plot(pc,type="loadings",scale.sdev=-1) # Downward
plot(pc,type="relative",scale.sdev=NA) # The directions
plot(pc,type="relative",scale.sdev=1) # one sigma Upward
plot(pc,type="relative",scale.sdev=-1) # one sigma Downward
biplot(pc)
screeplot(pc)
```

```
loadings(pc)
relativeLoadings(pc,mult=FALSE)
relativeLoadings(pc)
relativeLoadings(pc,scale.sdev=1)
relativeLoadings(pc,scale.sdev=2)

pc$Loadings
pc$DownLoadings
barplot(pc$Loadings)
pc$sdev^2
p = predict(pc,sa.lognormals5)
cov(p)
```

`princomp.aplus` *Principal component analysis for amounts in log geometry*

Description

A principal component analysis is done in the Aitchison geometry (i.e. ilt-transform). Some gimics simplify the interpretation of the computed components as perturbations of amounts.

Usage

```
## S3 method for class 'aplus'
princomp(x,...,scores=TRUE,center=attr(covmat,"center"),
                           covmat=var(x,robust=robust,giveCenter=TRUE),
                           robust=getOption("robust"))
## S3 method for class 'princomp.aplus'
print(x,...)
## S3 method for class 'princomp.aplus'
plot(x,y=NULL,..., npcs=min(10,length(x$sdev)),
                type=c("screeplot","variance","biplot","loadings","relative"),
                main=NULL,scale.sdev=1)
## S3 method for class 'princomp.aplus'
predict(object,newdata,...)
```

Arguments

`x`	an aplus dataset (for princomp) or a result from princomp.aplus
`y`	not used
`scores`	a logical indicating whether scores should be computed or not
`npcs`	the number of components to be drawn in the scree plot
`type`	type of the plot: `"screeplot"` is a lined screeplot, `"variance"` is a boxplot-like screeplot, `"biplot"` is a biplot, `"loadings"` displays the loadings as a `barplot.acomp`
`scale.sdev`	the multiple of sigma to use when plotting the loadings

main	title of the plot
object	a fitted princomp.aplus object
newdata	another amount dataset of class aplus
...	further arguments to pass to internally-called functions
covmat	provides the covariance matrix to be used for the principle component analysis
center	provides the be used for the computation of scores
robust	Gives the robustness type for the calculation of the covariance matrix. See var.rmult for details.

Details

As a metric euclidean space, the positive real space described in aplus has its own principal component analysis, that can be performed either in terms of the covariance matrix or the correlation matrix. However, since all parts in a composition or in an amount vector share a natural scaling, they do not need the standardization (which in fact would produce a loss of important information). For this reason, princomp.aplus works on the covariance matrix.
To aid the interpretation we added some extra functionality to a normal princomp(ilt(x)). First of all the result contains as additional information the amount representation of returned vectors in the space of the data: the center as an amount Center, and the loadings in terms of amounts to perturbe with, either positively (Loadings) or negatively (DownLoadings). The Up- and DownLoadings are normalized to the number of parts and not to one to simplify the interpretation. A value of about one means no change in the specific component.
The plot routine provides screeplots (type = "s",type= "v"), biplots (type = "b"), plots of the effect of loadings (type = "b") in scale.sdev*sdev-spread, and loadings of pairwise (log-)ratios (type = "r").
The interpretation of a screeplot does not differ from ordinary screeplots. It shows the eigenvalues of the covariance matrix, which represent the portions of variance explained by the principal components.
The interpretation of the the biplot uses, additionally to the classical one, a compositional concept: The differences between two arrowheads can be interpreted as log-ratios between the two components represented by the arrows.
The amount loading plot is introduced with this package. The loadings of all component can be seen as an orthogonal basis in the space of ilt-transformed data. These vectors are displayed by a barplot with their corresponding amounts. A portion of one means no change of this part. This is equivalent to a zero loading in a real principal component analysis.
The loadings plot can work in two different modes. If scale.sdev is set to NA it displays the amount vector being represented by the unit vector of loadings in the ilt-transformed space. If scale.sdev is numeric we use this amount vector scaled by the standard deviation of the respective component.
The relative plot displays the relativeLoadings as a barplot. The deviation from a unit bar shows the effect of each principal component on the respective ratio. The interpretation of the ratios plot may only be done in an Aitchison-compositional framework (see princomp.acomp).

Value

princomp gives an object of type c("princomp.acomp","princomp") with the following content:

sdev	the standard deviation of the principal components

loadings	the matrix of variable loadings (i.e., a matrix which columns contain the eigenvectors). This is of class "loadings".
center	the ilt-transformed vector of means used to center the dataset
Center	the aplus vector of means used to center the dataset
scale	the scaling applied to each variable
n.obs	number of observations
scores	if scores = TRUE, the scores of the supplied data on the principal components. Scores are coordinates in a basis given by the principal components and thus not compositions
call	the matched call
na.action	not clearly understood
Loadings	vectors of amounts that represent a perturbation with the vectors represented by the loadings of each of the factors
DownLoadings	vectors of amounts that represent a perturbation with the inverses of the vectors represented by the loadings of each of the factors

predict returns a matrix of scores of the observations in the newdata dataset
. The other routines are mainly called for their side effect of plotting or printing and return the object x.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

ilt,aplus, relativeLoadings princomp.acomp, princomp.rplus, barplot.aplus, mean.aplus,

Examples

```
data(SimulatedAmounts)
pc <- princomp(aplus(sa.lognormals5))
pc
summary(pc)
plot(pc)           #plot(pc,type="screeplot")
plot(pc,type="v")
plot(pc,type="biplot")
plot(pc,choice=c(1,3),type="biplot")
plot(pc,type="loadings")
plot(pc,type="loadings",scale.sdev=-1) # Downward
plot(pc,type="relative",scale.sdev=NA) # The directions
plot(pc,type="relative",scale.sdev=1) # one sigma Upward
plot(pc,type="relative",scale.sdev=-1) # one sigma Downward
biplot(pc)
screeplot(pc)
loadings(pc)
relativeLoadings(pc,mult=FALSE)
relativeLoadings(pc)
```

```
relativeLoadings(pc,scale.sdev=1)
relativeLoadings(pc,scale.sdev=2)

pc$Loadings
pc$DownLoadings
barplot(pc$Loadings)
pc$sdev^2
cov(predict(pc,sa.lognormals5))
```

`princomp.rcomp` *Principal component analysis for real compositions*

Description

A principal component analysis is done in real geometry (i.e. cpt-transform) of the simplex. Some gimics simplify the interpretation of the obtained components.

Usage

```
 ## S3 method for class 'rcomp'
princomp(x,...,scores=TRUE,center=attr(covmat,"center"),
                           covmat=var(x,robust=robust,giveCenter=TRUE),
                           robust=getOption("robust"))
 ## S3 method for class 'princomp.rcomp'
print(x,...)
 ## S3 method for class 'princomp.rcomp'
plot(x,y=NULL,...,npcs=min(10,length(x$sdev)),
        type=c("screeplot","variance","biplot","loadings","relative"),
        main=NULL,scale.sdev=1)
 ## S3 method for class 'princomp.rcomp'
predict(object,newdata,...)
```

Arguments

`x`	an rcomp dataset (for princomp) or a result from princomp.rcomp
`y`	not used
`scores`	a logical indicating whether scores should be computed or not
`npcs`	the number of components to be drawn in the scree plot
`type`	type of the plot: `"screeplot"` is a lined screeplot, `"variance"` is a boxplot-like screeplot, `"biplot"` is a biplot, `"loadings"` displays the loadings as a `barplot`
`scale.sdev`	the multiple of sigma to use when plotting the loadings
`main`	title of the plot
`object`	a fitted princomp.rcomp object
`newdata`	another compositional dataset of class rcomp
`...`	further arguments to pass to internally-called functions

covmat	provides the covariance matrix to be used for the principle component analysis
center	provides the be used for the computation of scores
robust	Gives the robustness type for the calculation of the covariance matrix. See var.rmult for details.

Details

Mainly a princomp(cpt(x)) is performed. To avoid confusion, the meaningless last principal component is removed.
The plot routine provides screeplots (type = "s",type= "v"), biplots (type = "b"), plots of the effect of loadings (type = "b") in scale.sdev*sdev-spread, and loadings of pairwise differences (type = "r").
The interpretation of a screeplot does not differ from ordinary screeplots. It shows the eigenvalues of the covariance matrix, which represent the portions of variance explained by the principal components.
The interpretation of the biplot strongly differs from a classical one. The relevant variables are not the arrows drawn (one for each component), but rather the links (i.e., the differences) between two arrow heads, which represents the difference between the two components represented by the arrows, or the transfer of mass from one to the other.
The compositional loading plot is more or less a standard one. The loadings are displayed by a barplot as positve and negative changes of amounts.
The loading plot can work in two different modes: If scale.sdev is set to NA it displays the composition being represented by the unit vector of loadings in cpt-transformed space. If scale.sdev is numeric we use this composition scaled by the standard deviation of the respective component.
The relative plot displays the relativeLoadings as a barplot. The deviation from a unit bar shows the effect of each principal component on the respective differences.

Value

princomp gives an object of type c("princomp.rcomp","princomp") with the following content:

sdev	the standard deviation of the principal components.
loadings	the matrix of variable loadings (i.e., a matrix which columns contain the eigenvectors). This is of class "loadings". The last eigenvalue is removed since it should contain the irrelevant scaling.
Loadings	the loadings as an rmult-object
center	the cpt-transformed vector of means used to center the dataset
Center	the rcomp vector of means used to center the dataset
scale	the scaling applied to each variable
n.obs	number of observations
scores	if scores = TRUE, the scores of the supplied data on the principal components. Scores are coordinates in a basis given by the principal components and thus not compositions
call	the matched call
na.action	not clearly understood

predict returns a matrix of scores of the observations in the newdata dataset. The other routines are mainly called for their side effect of plotting or printing and return the object x.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

cpt,rcomp, relativeLoadings princomp.acomp, princomp.rplus,

Examples

```
data(SimulatedAmounts)
pc <- princomp(rcomp(sa.lognormals5))
pc
summary(pc)
plot(pc)           #plot(pc,type="screeplot")
plot(pc,type="v")
plot(pc,type="biplot")
plot(pc,choice=c(1,3),type="biplot")
plot(pc,type="loadings")
plot(pc,type="loadings",scale.sdev=-1) # Downward
plot(pc,type="relative",scale.sdev=NA) # The directions
plot(pc,type="relative",scale.sdev=1) # one sigma Upward
plot(pc,type="relative",scale.sdev=-1) # one sigma Downward
biplot(pc)
screeplot(pc)
loadings(pc)
relativeLoadings(pc,mult=FALSE)
relativeLoadings(pc)
relativeLoadings(pc,scale.sdev=1)
relativeLoadings(pc,scale.sdev=2)

pc$sdev^2
cov(predict(pc,sa.lognormals5))
```

---

princomp.rmult *Principal component analysis for real data*

Description

Performs a principal component analysis for datasets of type rmult.

Usage

```
## S3 method for class 'rmult'
princomp(x,cor=FALSE,scores=TRUE,
                   covmat=var(rmult(x[subset,]),robust=robust,giveCenter=TRUE),
                   center=attr(covmat,"center"),  subset = rep(TRUE, nrow(x)),
                   ..., robust=getOption("robust"))
```

Arguments

x	a rmult-dataset
...	Further arguments to call princomp.default
cor	logical: shall the computation be based on correlations rather than covariances?
scores	logical: shall scores be computed?
covmat	provides the covariance matrix to be used for the principle component analysis
center	provides the be used for the computation of scores
subset	A rowindex to x giving the columns that should be used to estimate the variance.
robust	Gives the robustness type for the calculation of the covariance matrix. See var.rmult for details.

Details

The function just does princomp(unclass(x),...,scale=scale) and is only here for convenience.

Value

An object of type princomp with the following fields

sdev	the standard deviation of the principal components.
loadings	the matrix of variable loadings (i.e., a matrix whose columns contain the eigenvectors). This is of class "loadings".
center	the mean that was substracted from the data set
scale	the scaling applied to each variable
n.obs	number of observations
scores	if scores = TRUE, the scores of the supplied data on the principal components. Scores are coordinates in a basis given by the principal components.
call	the matched call
na.action	Not clearly understood

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

princomp.rplus

Examples

```
data(SimulatedAmounts)
pc <- princomp(rmult(sa.lognormals5))
pc
summary(pc)
plot(pc)
screeplot(pc)
screeplot(pc,type="l")
biplot(pc)
biplot(pc,choice=c(1,3))
loadings(pc)
plot(loadings(pc))
pc$sdev^2
cov(predict(pc,sa.lognormals5))
```

`princomp.rplus` *Principal component analysis for real amounts*

Description

A principal component analysis is done in real geometry (i.e. using iit-transform).

Usage

```
 ## S3 method for class 'rplus'
princomp(x,...,scores=TRUE,center=attr(covmat,"center"),
                           covmat=var(x,robust=robust,giveCenter=TRUE),
                           robust=getOption("robust"))
 ## S3 method for class 'princomp.rplus'
print(x,...)
 ## S3 method for class 'princomp.rplus'
plot(x,y=NULL,...,npcs=min(10,length(x$sdev)),
          type=c("screeplot","variance","biplot","loadings","relative"),
          main=NULL,scale.sdev=1)
 ## S3 method for class 'princomp.rplus'
predict(object,newdata,...)
```

Arguments

`x`	an rplus-dataset (for princomp) or a result from princomp.rplus
`y`	not used
`scores`	a logical indicating whether scores should be computed or not
`npcs`	the number of components to be drawn in the scree plot
`type`	type of the plot: `"screeplot"` is a lined screeplot, `"variance"` is a boxplot-like screeplot, `"biplot"` is a biplot, `"loadings"` displays the loadings as a `barplot`
`scale.sdev`	the multiple of sigma to use when plotting the loadings

main	title of the plot
object	a fitted princomp.rplus object
newdata	another amount dataset of class rcomp
...	further arguments to pass to internally-called functions
covmat	provides the covariance matrix to be used for the principle component analysis
center	provides the be used for the computation of scores
robust	Gives the robustness type for the calculation of the covariance matrix. See var.rmult for details.

Details

Mainly a princomp(iit(x)) is performed. Note all parts in a composition or in an amount vector share a natural scaling. Therefore, they do not need any preliminary standardization (which in fact would produce a loss of important information). For this reason, princomp.rplus works on the covariance matrix.
The plot routine provides screeplots (type = "s",type= "v"), biplots (type = "b"), plots of the effect of loadings (type = "b") in scale.sdev*sdev-spread, and loadings of pairwise differences (type = "r").
The interpretation of a screeplot does not differ from ordinary screeplots. It shows the eigenvalues of the covariance matrix, which represent the portions of variance explained by the principal components.
The interpretation of the biplot uses, additionally to the classical interperation, a compositional concept: the differences between two arrowheads can be interpreted as the shift of mass between the two components represented by the arrows.
The amount loading plot is more or less a standard loadings plot. The loadings are displayed by a barplot as positive and negative changes of amounts.
The loadings plot can work in two different modes: If scale.sdev is set to NA it displays the amount vector being represented by the unit vector of loadings in the iit-transformed space. If scale.sdev is numeric we use this amount vector scaled by the standard deviation of the respective component.
The relative plot displays the relativeLoadings as a barplot. The deviation from a unit bar shows the effect of each principal component on the respective differences.

Value

princomp gives an object of type c("princomp.rcomp","princomp") with the following content:

sdev	the standard deviation of the principal components
loadings	the matrix of variable loadings (i.e., a matrix which columns contain the eigenvectors). This is of class "loadings"
Loadings	the loadings as an "rmult"-object
center	the iit-transformed vector of means used to center the dataset
Center	the rplus vector of means used to center the dataset (center and Center have no difference, except that the second has a class)
scale	the scaling applied to each variable
n.obs	number of observations

scores	if scores = TRUE, the scores of the supplied data on the principal components. Scores are coordinates in a basis given by the principal components and thus not compositions
call	the matched call
na.action	not clearly understood

predict returns a matrix of scores of the observations in the newdata dataset.
The other routines are mainly called for their side effect of plotting or printing and return the object x.

See Also

iit,rplus, relativeLoadings princomp.rcomp, princomp.aplus,

Examples

```
data(SimulatedAmounts)
pc <- princomp(rplus(sa.lognormals5))
pc
summary(pc)
plot(pc)           #plot(pc,type="screeplot")
plot(pc,type="v")
plot(pc,type="biplot")
plot(pc,choice=c(1,3),type="biplot")
plot(pc,type="loadings")
plot(pc,type="loadings",scale.sdev=-1) # Downward
plot(pc,type="relative",scale.sdev=NA) # The directions
plot(pc,type="relative",scale.sdev=1) # one sigma Upward
plot(pc,type="relative",scale.sdev=-1) # one sigma Downward
biplot(pc)
screeplot(pc)
loadings(pc)
relativeLoadings(pc,mult=FALSE)
relativeLoadings(pc)
relativeLoadings(pc,scale.sdev=1)
relativeLoadings(pc,scale.sdev=2)

pc$sdev^2
cov(predict(pc,sa.lognormals5))
```

---

print.acomp *Printing compositional data.*

Description

Prints compositional objects with appropriate missing encodings.

Usage

```
## S3 method for class 'acomp'
print(x,...,replace0=TRUE)
## S3 method for class 'aplus'
print(x,...,replace0=TRUE)
## S3 method for class 'rcomp'
print(x,...,replace0=FALSE)
## S3 method for class 'rplus'
print(x,...,replace0=FALSE)
```

Arguments

x	a compositional object
...	further arguments to print.default
replace0	logical: Shall 0 be treated as "Below detection Limit" with unkown limit.

Details

Missings are displayed with an appropriate encoding:

MAR Missing at random: The value is missing independently of its true value.

MNAR Missing NOT at random: The value is missing dependently of its true value, but without a known systematic. Maybe a better name would be: Value dependen missingness.

BDL below detection limit (with unspecified detection limit): The value is missing because it was below an unkown detection limit.

<Detectionlimit below detection limit (with specified detection limit): The value is below the displayed detection limit.

SZ Structural Zero: A true value is either bound to be zero or does not exist for structural nonrandom reasons. E.g. the portion of pregnant girls at a boys school.

ERR Error: An illegal encoding value was found in the object.

Value

An invisible version of x.

Missing Policy

The policy of treatment of zeroes, missing values and values below detecion limit is explained in depth in compositions.missings.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de, Raimon Tolosana-Delgado

References

Boogaart, K.G. v.d., R. Tolosana-Delgado, M. Bren (2006) Concepts for handling of zeros and missing values in compositional data, in: E. Pirard (ed.) (2006)Proceedings of the IAMG'2006 Annual Conference on "Quantitative Geology from multiple sources", September 2006, Liege, Belgium,, S07-01, 4pages, ISBN 978-2-9600644-0-7

See Also

clr,acomp, plot.acomp, boxplot.acomp, barplot.acomp, mean.acomp, var.acomp, variation.acomp, zeroreplace

Examples

```
data(SimulatedAmounts)
mydata <- simulateMissings(sa.groups5,dl=0.01,knownlimit=TRUE,
                      MAR=0.05,MNARprob=0.05,SZprob=0.05)
mydata[1,1]<-BDLvalue
print(aplus(mydata))
print(aplus(mydata),digits=3)
print(acomp(mydata))
print(rplus(mydata))
print(rcomp(mydata))
```

---

pwlr *Pairwise log ratio transform*

Description

Compute the pairwise log ratio transform of a (dataset of) composition(s), and its inverse.

Usage

```
pwlr( x, as.rmult=FALSE, as.data.frame=!as.rmult, ...)
pwlrInv( z, orig=gsi.orig(z))
```

Arguments

x	a composition, not necessarily closed
z	the pwlr-transform of a composition, thus a [D(D-1)/2]-dimensional real vector, or a matrix with such many columns
as.rmult	logical; should the output be produced as an rmult object?
as.data.frame	logical; should be as a data.frame? if both are false, rmult will be taken
...	currently unused
orig	the original composition, to check consistency and recover component names

Details

The pwlr-transform maps a composition in the D-part Aitchison-simplex isometrically to a $D(D-1)/2$ dimensonal euclidian vector, computing each possible logratio (accounting for the fact that $log(A/B)=-log(B/A)$, and therefore only one of them is necessary). The data can then be analysed in this transformation by multivariate analysis tools not relying on the invertibility of the covariance function. The interpretation of the results is relatively simple, since each component captures the behaviour of the simple ratio between two party. However redundance between them is extremely high, and any of `alr`, `clr` or `ilr` transformations may be preferred in most applications.

The pairwise logratio transform is given by

$$pwlr(x)_{ij} := \ln \frac{x_i}{x_j}$$

.

The inverse requires some explanation, because of the redundance between pwlr scores. Note that for any three components A,B,C it holds that $log(A/C)=log(A/B)+log(B/C)$. So, any vector of $D(D-1)/2$ coefficients will not be necessarily a valid pwlr-transformed composition: if these coefficients do not satisfy that kind of relations, the vector is, strictly speaking, not a pwlr and should not be inverted. Nevertheless, the function gives a least-squares inversion, as proposed by Tolosana-Delgado and von Eynatten (2009).

Value

`pwlr` gives the pairwise log ratio transform; accepts a compositional dataset `pwlrInv` gives a closed composition with the given wplr-transform; accepts a dataset

Author(s)

R. Tolosana-Delgado http://www.stat.boogaart.de

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

Tolosana-Delgado, R. and H. von Eynatten (2009); Grain-size control on petrographic composition of sediments: compositional regression and rounded zeroes. *Mathematical Geosciences*: 41(8): 869-886. doi: 10.1007/s1100400992166.

See Also

`clr`,`alr`,`apt`, https://ima.udg.edu/Activitats/CoDaWork03/

Examples

```
(tmp <- pwlr(c(1,2,3)))
pwlrInv(tmp)
```

pwlrPlot *Plots of pairwise logratio against a covariable.*

Description

Creates a matrix of plots, with each pairwise logratio against a covariable. The covariable can be numeric or factor, and play the role of X or Y axis.

Usage

```
pwlrPlot(x,y,...,add.line=FALSE,line.col=2,add.robust=FALSE,rob.col=4)
```

Arguments

x	a vector, a column of a data.frame, or an acomp representing the first set of things to be displayed. Either x or y must be an acomp object, and the other must be a covariable. Both factors and continuous covariables allowed here.
y	a vector, a column of a data.frame, or an acomp representing the first set of things to be displayed. Either x or y must be an acomp object, and the other must be a covariable. Factors to be used here with caution.
...	furter parameters to the panel function
add.line	logical, to control the addition of a regression line in each panel. Ignored if covariable is a factor.
line.col	in case the regression line is added, which color should be used? Defaults to red.
add.robust	logical, to control the addition of a robust regression line in each panel. Ignored if covariable is a factor. This is nowadays based on lmrob, but this can change in the future.
rob.col	in case the robust regression line is added, which color should be used? Defaults to blue.

Details

This function generates a matrix of plots of all possible pairwise logratios of the acomp argument, plotted against a covariable. The covariable can be a factor or a numeric vector, or a column of a matrix or data.frame. Covariable and composition can both be represented in X or Y axis: a factor on X axis generates a boxplot; a factor on Y axis generates a spineplot; if the covariable is numeric, a default scatterplot is generated. All dot arguments are passed to these plotting functions. In any of these cases, the diagram shows the logratio of the component in the row divided by the component in the column. In the case of a numeric covariable, both classical and robust regression lines can be added.

Author(s)

Raimon Tolosana-Delgado, K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

Boogaart, K.G. v.d. , R. Tolosana (2008) Mixing Compositions and Other scales, Proceedings of CodaWork 08.

https://ima.udg.edu/Activitats/CoDaWork03/

https://ima.udg.edu/Activitats/CoDaWork05/

https://ima.udg.edu/Activitats/CoDaWork08/

See Also

plot.aplus, pairwisePlot, boxplot, spineplot, plot.default

Examples

```
data(Hydrochem)
xc = acomp(Hydrochem[,c("Ca","Mg","Na","K")])
fk = Hydrochem$River
pH = -log10(Hydrochem$H)
## x=acomp, y=factor
pwlrPlot(xc, fk, border=2:5)
## x=factor, y=acomp
pwlrPlot(fk,xc, col=2:5)
## x=acomp, y=numeric, with colors by river
pwlrPlot(xc, pH, col=as.integer(fk)+1)
## x=numeric, y=acomp, with line
pwlrPlot(pH, xc, add.robust=TRUE)
```

---

qqnorm *Normal quantile plots for compositions and amounts*

Description

The plots allow to check the normal distribution of multiple univaritate marginals by normal quantile-quantile plots. For the different interpretations of amount data a different type of normality is assumed and checked. When an alpha-level is given the marginal displayed in each panel is checked for normality.

Usage

```
## S3 method for class 'acomp'
qqnorm(y,fak=NULL,...,panel=vp.qqnorm,alpha=NULL)
## S3 method for class 'rcomp'
qqnorm(y,fak=NULL,...,panel=vp.qqnorm,alpha=NULL)
## S3 method for class 'aplus'
qqnorm(y,fak=NULL,...,panel=vp.qqnorm,alpha=NULL)
```

```
## S3 method for class 'rplus'
qqnorm(y,fak=NULL,...,panel=vp.qqnorm,alpha=NULL)
vp.qqnorm(x,y,...,alpha=NULL)
```

Arguments

y	a dataset
fak	a factor to split the dataset, not yet implemented in aplus and rplus
panel	the panel function to be used or a list of multiple panel functions
alpha	the alpha level of a test for normality to be performed for each of the displayed marginals. The levels are adjusted for multiple testing with a Bonferroni-correction (i.e. dividing each of the alpha-level by the number of test performed)
...	further graphical parameters
x	used by pairs only. Internal use

Details

qqnorm.rplus and qqnorm.rcomp display qqnorm plots of individual amounts (on the diagonal), of pairwise differences of amounts (above the diagonal) and of pairwise sums of amounts (below the diagonal).
qqnorm.aplus displays qqnorm-plots of individual log-amounts (on the diagonal), of pairwise log-ratios of amounts (above the diagonal) and of pairwise sums of log amount (below the diagonal).
qqnorm.acomp displays qqnorm-plots of pairwise log-ratios of amounts in all of diagonal panels. Nothing is displayed on the diagonal.
In all cases a joint normality of the original data in the selected framework would imply normality in all displayed marginal distributions (although the reciprocal is in general not true!).
The marginal normality can be checked in each of the plots using a shapiro.test, by specifying an alpha level. The alpha level is corrected for multiple testing. Plots displaying a marginal distribution significantly deviating from a normal distribution at that alpha level are marked by a red exclamation mark.
vp.qqnorm is internally used as a panel function to make high dimensional plots.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

plot.acomp, boxplot.acomp, rnorm.acomp, rnorm.rcomp, rnorm.aplus, rnorm.rplus

Examples

```
data(SimulatedAmounts)
qqnorm(acomp(sa.lognormals),alpha=0.05)
qqnorm(rcomp(sa.lognormals),alpha=0.05)
qqnorm(aplus(sa.lognormals),alpha=0.05)
qqnorm(rplus(sa.lognormals),alpha=0.05)
```

R2 *R square*

Description

The R2 measure of determination for linear models

Usage

```
R2(object,...)
## S3 method for class 'lm'
R2(object,...,adjust=TRUE,ref=0)
## Default S3 method:
R2(object,...,ref=0)
```

Arguments

`object`	a statistical model
`...`	further not yet used parameters
`adjust`	Logical, whether the estimate of R2 should be adjusted for the degrees of freedom of the model.
`ref`	A reference model for computation of a relative R^2.

Details

The R^2 measure of determination is defined as:

$$R^2 = 1 - \frac{var(residuals)}{var(data)}$$

and provides the portion of variance explained by the model. It is a number between 0 and 1, where 1 means the model perfectly explains the data and 0 means that the model has no better explanation of the data than a constant mean. In case of multivariate models metric variances are used.

If a reference model is given by `ref`, the variance of the residuals of that models rather than the variance of the data is used. The value of such a relative R^2 estimates how much of the residual variance is explained.

If `adjust=TRUE` the unbiased estiamators for the variances are used, to avoid the automatisme that a more parameters automatically lead to a higher R^2.

Value

The R2 measure of determination.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

lm, mvar, AIC

Examples

```
data(Orange)
R2(lm(circumference~age,data=Orange))
R2(lm(log(circumference)~log(age),data=Orange))
```

---

rAitchison *Aitchison Distribution*

Description

The Aitchison distribution is a class of distributions the simplex, containing the normal and the Dirichlet as subfamilies.

Usage

```
dAitchison(x,
           theta=alpha+sigma %*% clr(mu),
           beta=-1/2*gsi.svdinverse(sigma),
           alpha=mean(theta),
           mu=clrInv(c(sigma%*%(theta-alpha))),
           sigma=-1/2*gsi.svdinverse(beta),
           grid=30,
           realdensity=FALSE,
           expKappa=AitchisonDistributionIntegrals(theta,beta,
                       grid=grid,mode=1)$expKappa)
rAitchison(n,
           theta=alpha+sigma %*% clr(mu),
           beta=-1/2*gsi.svdinverse(sigma),
           alpha=mean(theta),
           mu=clrInv(c(sigma%*%(theta-alpha))),
           sigma=-1/2*gsi.svdinverse(beta), withfit=FALSE)
AitchisonDistributionIntegrals(
           theta=alpha+sigma %*% clr(mu),
           beta=-1/2*gsi.svdinverse(sigma),
           alpha=mean(theta),
           mu=clrInv(c(sigma%*%(theta-alpha))),
           sigma=-1/2*gsi.svdinverse(beta),
           grid=30,
           mode=3)
```

Arguments

x	acomp-compositions the density should be computed for.
n	integer: number of datasets to be simulated
theta	numeric vector: Location parameter vector
beta	matrix: Spread parameter matrix (clr or ilr)
alpha	positiv scalar: departure from normality parameter (positive scalar)
mu	acomp-composition, normal reference mean parameter composition
sigma	matrix: normal reference variance matrix (clr or ilr)
grid	integer: number of discretisation points along each side of the simplex
realdensity	logical: if true the density is given with respect to the Haar measure of the real simplex, if false the density is given with respect to the Aitchison measure of the simplex.
mode	integer: desired output: -1: Compute nothing, only transform parameters, 0: Compute only oneIntegral and kappaIntegral, 1: compute also the clrMean, 2: compute also the clrSqExpectation, 3: same as 2, but compute clrVar instead of clrSqExpectation
expKappa	The closing divisor of the density
withfit	Should a pre-spliting of the Aitchison density be used for simulation?

Details

The Aitchison Distribution is a joint generalisation of the Dirichlet Distribution and the additive log-normal distribution (or normal on the simplex). It can be parametrized by Ait(theta,beta) or by Ait(alpha,mu,Sigma). Actually, beta and Sigma can be easily transformed into each other, such that only one of them is necessary. Parameter theta is a vector in R^D, alpha is its sum, mu is a composition in S^D, and beta and sigma are symmetric matrices, which can either be expressed in ilr or clr space. The parameters are transformed as

$$\beta = -1/2\Sigma^{-1}$$

$$\theta = clr(\mu)\Sigma + \alpha(1, \ldots, 1)^t$$

The distribution exists, if either, $\alpha \geq 0$ and Sigma is positive definite (or beta negative definite) in ilr-coordinates, or if each theta is strictly positive and Sigma has at least one positive eigenvalue (or beta has at least one negative eigenvalue). The simulation procedure currently only works with the first case.
AitchisonDistributionIntegral is a convenience function to compute the parameter transformation and several functions of these parameters. This is done by numerical integration over a multinomial simplex lattice of D parts with grid many elements (see xsimplex).
The density of the Aitchison distribution is given by:

$$f(x, \theta, \beta) = exp((\theta - 1)^t \log(x) + ilr(x)^t \beta ilr(x))/exp(\kappa_{Ait(\theta,\beta)})$$

with respect to the classical Haar measure on the simplex, and as

$$f(x, \theta, \beta) = exp(\theta^t \log(x) + ilr(x)^t \beta ilr(x))/exp(\kappa_{Ait(\theta,\beta)})$$

with respect to the Aitchison measure of the simplex. The closure constant expKappa is computed numerically, in `AitchisonDistributionIntegrals`.

The random composition generation is done by rejection sampling based on an optimally fitted additive logistic normal distribution. Thus, it only works if the correponding Sigma in ilr would be positive definite.

Value

`dAitchison` Returns the density of the Aitchison distribution evaluated at x as a numeric vector.

`rAitchison` Returns a sample of size n of simulated compostions as an acomp object.

`AitchisondistributionIntegrals`
Returns a list with

theta the theta parameter given or computed

beta the beta parameter given or computed

alpha the alpha parameter given or computed

mu the mu parameter given or computed

sigma the sigma parameter given or computed

expKappa the integral over the density without closing constant. I.e. the inverse of the closing constant and the exp of $\kappa_{Ait(\theta,\beta)}$

kappaIntegral The expected value of the mean of the logs of the components as numerically computed

clrMean The mean of the clr transformed random variable, computed numerically

clrSqExpectation The expectation of $clr(X)clr(X)^t$ computed numerically.

clrVar The variance covariance matrix of clr(X), computed numerically

Note

The simulation procedure currently only works with a positive definite Sigma. You need a relatively high grid constant for precise values in the numerical integration.

Author(s)

K.Gerald v.d. Boogaart, R. Tolosana-Delgado http://www.stat.boogaart.de

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

See Also

`runif.acomp`, `rnorm.acomp`, `rDirichlet.acomp`

Examples

```
(erg<-AitchisonDistributionIntegrals(c(-1,3,-2),ilrvar2clr(-diag(c(1,2))),grid=20))

(myvar<-with(erg, -1/2*ilrvar2clr(solve(clrvar2ilr(beta)))))
(mymean<-with(erg,myvar%*%theta))

with(erg,myvar-clrVar)
with(erg,mymean-clrMean)
```

`ratioLoadings` *Loadings of relations of two amounts*

Description

In a compositional dataset the relation of two objects can be interpreted safer than a single amount. These functions compute, display and plot the corresponding pair-information for the various principal component analysis results.

Usage

```
relativeLoadings(x,...)
## S3 method for class 'princomp.acomp'
relativeLoadings(x,...,log=FALSE,scale.sdev=NA,
                                                  cutoff=0.1)
## S3 method for class 'princomp.aplus'
relativeLoadings(x,...,log=FALSE,scale.sdev=NA,
                                                  cutoff=0.1)
## S3 method for class 'princomp.rcomp'
relativeLoadings(x,...,scale.sdev=NA,
                                                  cutoff=0.1)
## S3 method for class 'princomp.rplus'
relativeLoadings(x,...,scale.sdev=NA,
                                                  cutoff=0.1)
## S3 method for class 'relativeLoadings.princomp.acomp'
print(x,...,cutoff=attr(x,"cutoff"),
                                                  digits=2)
## S3 method for class 'relativeLoadings.princomp.aplus'
print(x,...,cutoff=attr(x,"cutoff"),
                                                 digits=2)
## S3 method for class 'relativeLoadings.princomp.rcomp'
print(x,...,cutoff=attr(x,"cutoff"),
                                                 digits=2)
## S3 method for class 'relativeLoadings.princomp.rplus'
print(x,...,cutoff=attr(x,"cutoff"),
                                                 digits=2)
## S3 method for class 'relativeLoadings.princomp.acomp'
```

```
plot(x,...)
## S3 method for class 'relativeLoadings.princomp.aplus'
plot(x,...)
## S3 method for class 'relativeLoadings.princomp.rcomp'
plot(x,...)
## S3 method for class 'relativeLoadings.princomp.rplus'
plot(x,...)
```

Arguments

x	a result from an amount PCA princomp.acomp/princomp.aplus/princomp.rcomp/princomp.rplus
log	a logical indicating to use log-ratios instead of ratios
scale.sdev	if not NA, a number specifying the multiple of a standard deviation, used to scale the components
cutoff	a single number. Changes under that (log)-cutoff are not displayed
digits	the number of digits to be displayed
...	further parameters to internally-called functions

Details

The relative loadings of components allow a direct interpretation of the effects of principal components. For acomp/aplus classes the relation is induced by a ratio, which can optionally be log-transformed. For the rcomp/rplus-classes the relation is induced by a difference, which is meaningless when the units are different.

Value

The value is a matrix of type "relativeLoadings.princomp.*", containing the ratios in the compositions represented by the loadings (optionally scaled by the standard deviation of the components and scale.sdev).

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

princomp.acomp, princomp.aplus, princomp.rcomp, princomp.rplus, barplot

Examples

```
data(SimulatedAmounts)
pc <- princomp(acomp(sa.lognormals5))
pc
summary(pc)
relativeLoadings(pc,log=TRUE)
relativeLoadings(pc)
relativeLoadings(pc,scale.sdev=1)
relativeLoadings(pc,scale.sdev=2)
```

```
plot(relativeLoadings(pc,log=TRUE))
plot(relativeLoadings(pc))
plot(relativeLoadings(pc,scale.sdev=1))
plot(relativeLoadings(pc,scale.sdev=2))
```

`rcomp` *Compositions as elements of the simplex embedded in the D-dimensional real space*

Description

A class providing a way to analyse compositions in the philosophical framework of the Simplex as subset of the R^D.

Usage

```
rcomp(X,parts=1:NCOL(oneOrDataset(X)),total=1,warn.na=FALSE,
      detectionlimit=NULL,BDL=NULL,MAR=NULL,MNAR=NULL,SZ=NULL)
```

Arguments

`X`	composition or dataset of compositions
`parts`	vector containing the indices xor names of the columns to be used
`total`	the total amount to be used, typically 1 or 100
`warn.na`	should the user be warned in case of NA,NaN or 0 coding different types of missing values?
`detectionlimit`	a number, vector or matrix of positive numbers giving the detection limit of all values, all columns or each value, respectively
`BDL`	the code for 'Below Detection Limit' in X
`SZ`	the code for 'Structural Zero' in X
`MAR`	the code for 'Missing At Random' in X
`MNAR`	the code for 'Missing Not At Random' in X

Details

Many multivariate datasets essentially describe amounts of D different parts in a whole. This has some important implications justifying to regard them as a scale on its own, called a "composition". The functions around the class `"rcomp"` follow the traditional (often statistically inconsistent) approach regarding compositions simply as a multivariate vector of positive numbers summing up to 1. This space of D positive numbers summing to 1 is traditionally called the D-1-dimensional simplex.

The compositional scale was in-depth analysed by Aitchison (1986) and he found serious reasons why compositional data should be analysed with a different geometry. The functions around the class "acomp" follow his approach. However the Aitchison approach based on log-ratios is sometimes criticized (e.g. Rehder and Zier, 2002). It cannot deal with absent parts (i.e. zeros). It is sensitive to large measurement errors in small amounts. The Aitchison operations cannot represent simple mixture of different compositions. The used transformations are not uniformly continuous. Straight lines and ellipses in Aitchison space look strangely in ternary diagrams. As all uncritical statistical analysis, blind application of logratio-based analysis is sometimes misleading. Therefore it is sometimes useful to analyse compositional data directly as a multivariate dataset of portions summing to 1. However a clear warning must be given that the utilisation of almost any kind of classical multivariate analysis introduce some kinds of artifacts (e.g. Chayes 1960) when applied to compositional data. So, extra care and considerable expert knowlegde is needed for the proper interpretation of results achieved in this non-Aitchison approach. The package tries to lead the user around these artifacts as much as possible and gives hints to major pitfalls in the help. However meaningless results cannot be fully avoided in this (rather inconsistent) approach.
A side effect of the procedure is to force the compositions to sum to one, which is done by the closure operation clo .
The classes rcomp, acomp, aplus, and rplus are designed in a fashion as similar as possible, in order to allow direct comparison between results achieved by the different approaches. Especially the acomp logistic transforms clr, alr, ilr are mirrored by analogous linear transforms cpt, apt, ipt in the rcomp class framework.

Value

a vector of class "rcomp" representing a closed composition or a matrix of class "rcomp" representing multiple closed compositions, by rows.

Missing Policy

Missing and Below Detecion Limit Policy is explained in deeper detail in compositions.missing.

Author(s)

Raimon Tolosana-Delgado, K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

Rehder, S. and U. Zier (2001) Letter to the Editor: Comment on "Logratio Analysis and Compositional Distance" by J. Aitchison, C. Barcel\'o-Vidal, J.A. Mart\'in-Fern\'a ndez and V. Pawlowsky-Glahn, *Mathematical Geology*, **33** (7), 845-848.

Zier, U. and S. Rehder (2002) Some comments on log-ratio transformation and compositional distance, *Terra Nostra*, Schriften der Alfred Wegener-Stiftung, 03/2003

van den Boogaart, K.G. and R. Tolosana-Delgado (2008) "compositions": a unified R package to analyze Compositional Data, *Computers & Geosciences*, 34 (4), pages 320-338, doi: 10.1016/j.cageo.2006.11.017.

See Also

cpt, apt, ipt, acomp, rplus, princomp.rcomp, plot.rcomp, boxplot.rcomp, barplot.rcomp, mean.rcomp, var.rcomp, variation.rcomp, cov.rcomp, msd, convex.rcomp, +.rcomp

Examples

```
data(SimulatedAmounts)
plot(rcomp(sa.tnormals))
```

---

rcomp-class *Class* "rcomp"

Description

The S4-version of the data container "rcomp" for compositional data. More information in rcomp

Objects from the Class

A virtual Class: No objects may be directly created from it. This is provided to ensure that rcomp objects behave as data.frame or structure under certain circumstances. Use rcomp to create these objects.

Slots

.Data: Object of class "list" containing the data itself

names: Object of class "character" with column names

row.names: Object of class "data.frameRowLabels" with row names

.S3Class: Object of class "character" with the class string

Extends

Class "data.frame", directly. Class "compositional", directly. Class "list", by class "data.frame", distance 2. Class "oldClass", by class "data.frame", distance 2. Class "vector", by class "data.frame", distance 3.

Methods

coerce signature(from = "rcomp", to = "data.frame"): to generate a data.frame

coerce signature(from = "rcomp", to = "structure"): to generate a structure (i.e. a vector, matrix or array)

coerce<- signature(from = "rcomp", to = "data.frame"): to overwrite a composition with a data.frame

Note

see rcomp

Author(s)

Raimon Tolosana-Delgado

References

see rcomp

See Also

see rcomp

Examples

```
showClass("rcomp")
```

`rcomparithm` *Arithmetic operations for compositions in a real geometry*

Description

The real compositions form a manifold of the real vector space. The induced operations +,-,*,/ give results valued in the real vector space, but possibly outside the simplex.

Usage

```
convex.rcomp(x,y,alpha=0.5)
## Methods for class "rcomp"
##   x+y
##   x-y
##   -x
##   x*r
##   r*x
##   x/r
```

Arguments

x	an rcomp composition or dataset of compositions
y	an rcomp composition or dataset of compositions
r	a numeric vector of size 1 or nrow(x)
alpha	a numeric vector of size 1 or nrow(x) with values between 0 and 1

Details

The functions behave quite like +.rmult.
The convex combination is defined as: x*alpha + (1-alpha)*y

Value

rmult-objects containing the given operations on the simplex as subset of the R^D. Only the convex combination convex.rcomp results in an rcomp-object again, since only this operation is closed.

Note

For * the arguments x and y can be exchanged.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

+.rmult, +.acomp,cpt, rcomp, rmult

Examples

```
rcomp(1:5)* -1 + rcomp(1:5)
data(SimulatedAmounts)
cdata <- rcomp(sa.lognormals)
plot( tmp <- (cdata-mean(cdata))/msd(cdata) )
class(tmp)
mean(tmp)
msd(tmp)
var(tmp)
plot(convex.rcomp(rcomp(c(1,1,1)),sa.lognormals,0.1))
```

---

rcompmargin *Marginal compositions in real geometry*

Description

Compute marginal compositions by amalgamating the rest (additively).

Usage

```
rcompmargin(X,d=c(1,2),name="+",pos=length(d)+1,what="data")
```

Arguments

X	composition or dataset of compositions
d	vector containing the indices xor names of the columns to be kept
name	The new name of the amalgamation column
pos	The position where the new amalgamation column should be stored. This defaults to the last column.
what	The role of X either "data" for data (or means) to be transformed or "var" for variances to be transformed.

Details

The amalgamation column is simply computed by adding the non-selected components after closing the composition. This is consistent with the rcomp approach and is widely used because of its easy interpretation. However, it often leads to difficult-to-read ternary diagrams and is inconsistent with the acomp approach.

With the argument what="var" the function transformes an rcomp variance to the resulting variance of the resulting composition.

Value

A closed compositions with class "rcomp" containing the selected variables given by d and the the amalgamation column.

Missing Policy

MNAR has the highest priority, MAR next and WZERO (BDL,SZ),- values are considered as 0 and reported as BDL in the End.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de, Raimon olosana-Delgado

References

References missing

See Also

acompmargin, rcomp

Examples

```
data(SimulatedAmounts)
plot.rcomp(sa.tnormals5,margin="rcomp")
plot.rcomp(rcompmargin(sa.tnormals5,c("Cd","Zn")))
plot.rcomp(rcompmargin(sa.tnormals5,c(1,2)))
```

`rDirichlet` *Dirichlet distribution*

Description

The Dirichlet distribution on the simplex.

Usage

```
dDirichlet(x, alpha, log=FALSE, measure="Lebesgue")
rDirichlet.acomp(n, alpha)
rDirichlet.rcomp(n, alpha)
```

Arguments

`n`	number of datasets to be simulated
`alpha`	parameters of the Dirichlet distribution
`x`	a data set (acomp, rcomp, data.frame, matrix; even one-row) of point(s) on the simplex
`log`	a boolean, controlling if the density or the log-density is returned
`measure`	one of: "Lebesgue" or "Aitchison" (partial match applies), or else a function returning the reference LOG-density (see details below)

Details

The Dirichlet distribution is the result of closing a vector of equally-scaled Gamma-distributed variables. It the conjugate prior distribution for a vector of probabilities of a multinomial distribution. Thus, it generalizes the beta distribution for more than two parts.

For the density, the implementation allows to obtain the conventional density (with respect to the Lebesgue measure, default behaviour or giving `measure="Lebuesgue"`), the compositional density (with respect to the Aitchison measure, giving `measure="Aitchison"`), or else w.r.to any other reference density (giving at `measure` a function returning the log-density of the reference measure for any point of the simplex)

Value

For `rDirichlet.*` a generated random dataset of class `"acomp"` or `"rcomp"`, drawn from a Dirichlet distribution with the given parameter `alpha`. The names of `alpha` are used to name the parts.

For `dDirichlet`, the (conventional) Dirichlet density

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de, Raimon Tolosana-Delgado

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

Mateu-Figueras, G.; Pawlowsky-Glahn, V. (2005). The Dirichlet distribution with respect to the Aitchison measure on the simplex, a first approach. In: Mateu-Figueras, G. and Barcel\'o-Vidal, C. (Eds.) *Proceedings of the 2nd International Workshop on Compositional Data Analysis*, Universitat de Girona, ISBN 84-8458-222-1, https://ima.udg.edu/Activitats/CoDaWork05/

See Also

`rnorm.acomp`

Examples

```
tmp <- rDirichlet.acomp(10,alpha=c(A=2,B=0.2,C=0.2))
plot(tmp)
dDirichlet(tmp, alpha=c(A=2,B=0.2,C=0.2))
dDirichlet(tmp[1,]*0, alpha=c(A=2,B=0.2,C=0.2))
```

`Read standard data files`

Reads a data file in a geoeas format

Description

Reads a data file, which must be formatted either as a geoEAS file (described below).

Usage

```
read.geoeas(file)
read.geoEAS(file)
```

Arguments

`file`	a file name, with a specific format

Details

The data files must be in the adequate format: "read.geoEAS" and "read.geoeas" read geoEAS format.

The geoEAS format has the following structure:

1 a first row with a description of the data set

2 the number of variables (=nvars)

3 "nvars" rows, each containing the name of a variable

4 the data set, in a matrix of "nvars" columns and as many rows as individuals

Value

A data set, with a "title" attribute.

Note

Labels and title should not contain tabs. This might produce an error when reading.

Author(s)

Raimon Tolosana-Delgado

References

Missing references

See Also

read.table

Examples

```
#
# Files can be found in the test-subdirectory of the package
#
## Not run:
  read.geoeas("TRUE.DAT")
  read.geoEAS("TRUE.DAT")

## End(Not run)
```

`replot` *Modify parameters of compositional plots.*

Description

Display only a subset of the plots.

Usage

```
replot(...,dev=dev.cur(),plot=TRUE,envir=NULL,add=FALSE)
replotable(expr,add=FALSE)
noreplot(expr,dev=dev.cur())
```

Arguments

expr	A (unquoted) expression that does the plotting. replotable will make the generated plot replotable and noreplot will do the inverse and avoid that the plots overwrites the current database entry.
...	Plot parameters to be modified. E.g. onlyPanel
dev	The device that currently contains the plot. It will be plotted in the current device.
plot	logical or call or list of calls. If plot is TRUE, the new version of the plot is plotted in the current environment (and typically stores itself here). If plot is FALSE the modified plot is simply stored, rather than actually plotted (in its own old plotting environment). If the parameter is something else, it is stored to the internal plot database for the given device (but not plotted or evaluated).
envir	a new enviroment to be assigned to the plot. Rarely needed.
add	either a logical to indicating that the plot adds something to the plot. Or a number / name of the added thing to be modified.

Details

Some of the plot routines of compositions internally store their call as a mean for replaying the plot when information is added or parameters are modified. The stored call can be modified by this function, which pretty much works like a simplified version of update.

replot allows to redo the plot typically in a different device or with different parameters. The function provides this functionallity at a totally different level than dev.copy and allows for the modification of high level parameters on the fly.

Plots can be stored in the internal database by calling replot with a parameter plot set to the call of that plot. Plotting functions without this functionallity can be filtered through replotable(). However in this case all parameter names should be given explicitly.

There are actually three levels of possible replay: The dev.copy level on which graphic actions are replayed. The gsi.pairs function level that organizes panels plots and uses an internal replotting facility to allow modification of the parameter, e.g. addings lines .... And than there is the high level of the actual function call generating the plot.

Value

replot returns an invisible copy of the modified call. replotable and noreplot return the result of expression.

Note

The function works by revaluating the call in its environment. Thus the plot will change!!! if the data has changed.

The function always handles the latest plot from the package. If another plot ignorant of the replot system has meanwhile be used it will be ignored.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

plot.acomp, plot.aplus, boxplot.acomp

Examples

```
data(SimulatedAmounts)
plot(acomp(sa.lognormals5))
straight(acomp(c(1,1,1,1,1)),acomp(c(1,2,3,4,5)))
replot(onlyPanel=c(2,3))
oldPlot <- replot(plot=FALSE) # get the plotting call
replotable(plot(x=1:10)) # To make a graphic replottable
replot(col=1:10)
replot(plot=oldPlot)    # Restore the old plot (without plotting)
replot(onlyPanel=NULL)    # View the whole plot again
replot(pch=20)  # Acctually plot it
replot(col=20)  # since the actual plot is gsi.pairs not a plot.acomp

## Not run:
# The following line in  a plotting function stores the plot for replotting.
replot(plot=match.call()) # Store current call as plot
replot()                  # simply plot once again
replot(dev=otherdev)      # redo a plot from an other device here.
replot(onlyPanel=c(3,4))  # modify the plot (and replot it)
replot(onlyPanel=c(3,4),dev=7,plot=FALSE) # modify a stored plot

## End(Not run)
```

---

rlnorm *The multivariate lognormal distribution*

Description

Generates random amounts with a multivariate lognormal distribution, or gives the density of that distribution at a given point.

Usage

```
rlnorm.rplus(n,meanlog,varlog)
dlnorm.rplus(x,meanlog,varlog)
```

Arguments

n	number of datasets to be simulated
meanlog	the mean-vector of the logs
varlog	the variance/covariance matrix of the logs
x	vectors in the sample space

Value

rlnorm.rplus gives a generated random dataset of class "rplus" following a lognormal distribution with logs having mean meanlog and variance varlog.
dlnorm.rplus gives the density of the distribution with respect to the Lesbesgue measure on R+ as a subset of R.

Note

The main difference between rlnorm.rplus and rnorm.aplus is that rlnorm.rplus needs a logged mean. The additional difference for the calculation of the density by dlnorm.rplus and dnorm.aplus is the reference measure (a log-Lebesgue one in the second case).

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de, Raimon Tolosana-Delgado

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

See Also

rnorm.acomp

Examples

```
MyVar <- matrix(c(
0.2,0.1,0.0,
0.1,0.2,0.0,
0.0,0.0,0.2),byrow=TRUE,nrow=3)
MyMean <- c(1,1,2)

plot(rlnorm.rplus(100,log(MyMean),MyVar))
plot(rnorm.aplus(100,MyMean,MyVar))
x <- rnorm.aplus(5,MyMean,MyVar)
dnorm.aplus(x,MyMean,MyVar)
dlnorm.rplus(x,log(MyMean),MyVar)
```

rMahalanobis *Compute distributions of empirical Mahalanobis distances based on simulations*

Description

Decissions about outliers are often made based on Mahalanobis distances with respect to robustly estimated variances. These function deliver the necessary distributions.

Usage

```
rEmpiricalMahalanobis(n,N,d,...,sorted=FALSE,pow=1,robust=TRUE)
pEmpiricalMahalanobis(q,N,d,...,pow=1,replicates=100,resample=FALSE,
                      robust=TRUE)
qEmpiricalMahalanobis(p,N,d,...,pow=1,replicates=100,resample=FALSE,
                      robust=TRUE)
rMaxMahalanobis(n,N,d,...,pow=1,robust=TRUE)
pMaxMahalanobis(q,N,d,...,pow=1,replicates=998,resample=FALSE,
                      robust=TRUE)
qMaxMahalanobis(p,N,d,...,pow=1,replicates=998,resample=FALSE,
                      robust=TRUE)
rPortionMahalanobis(n,N,d,cut,...,pow=1,robust=TRUE)
pPortionMahalanobis(q,N,d,cut,...,replicates=1000,resample=FALSE,pow=1,
                      robust=TRUE)
qPortionMahalanobis(p,N,d,cut,...,replicates=1000,resample=FALSE,pow=1,
                      robust=TRUE)
pQuantileMahalanobis(q,N,d,p,...,replicates=1000,resample=FALSE,
                      ulimit=TRUE,pow=1,robust=TRUE)
```

Arguments

n	Number of simulations to do.
q	A vector giving quantiles of the distribution
p	A vector giving probabilities. (only a single probility for pQuantileMahalanobis)
N	Number of cases in the dataset.
d	degrees of freedom (i.e. dimension) of the dataset.
cut	A cutting limit. The random variable is the portion of Mahalanobis distances lower equal to the cutting limit.
...	further arguments passed to MahalanobisDist
pow	the power of the Mahalanobis distance to be used. Higher powers can be used to stretch the outlierregion visually.
robust	logical or a robust method description (see robustnessInCompositions) specifiying how the center and covariance matrix are estimated,if not given.

sorted	Specifies a transformation to be applied to the whole sequence of Mahalanobis distances: FALSE is no transformation, TRUE sorts the entries in ascending order, a numeric vector picks the given entries from the entries sorted in ascending order; alternatively a function such as max can be given to directly transform the data.
replicates	the number of datasets in the Monte-Carlo-Computations used in these routines.
resample	a logical forcing a resampling of the Monte-Carlo-Sampling. See details.
ulimit	logical: is this an upper limit of a joint confidence bound or a lower limit.

Details

All the distribution correspond to the distribution under the Null-Hypothesis of multivariate joint Gaussian distribution of the dataset.

The set of empirically estimated Mahalanobis distances of a dataset is in the first step a random vector with exchangable but dependent entries. The distribution of this vector is given by the rEmpiricalMahalanobis if no sorted argument is given. Please be advised that this is not a fixed distribution in a mathematical sense, but an implementation dependent distribution incorporating the performance of underlying robust spread estimator. As long as no sorted argument is given pEmpiricalMahalanobis and qEmpiricalMahalanobis represent the distribution function and the quantile function of a randomly picked element of this vector.
If a sorted attribute is given, it specifies a transformation is applied to each of the vector prior to processing. Three important special cases are provided by seperate functions. The MaxMahalanobis functions correspond to picking only the larges value. The PortionMahalanobis functions correspond to reporting the portion of Mahalanobis distances over a cutoff. The QuantileMahalanobis distribution correponds to the distribution of the p-quantile of the dataset.
The Monte-Carlo-Simulations of these distributions are rather slow, since for each datum we need to simulate a whole dataset and to apply a robust covariance estimator to it, which typically itself involves Monte-Carlo-Algorithms. Therefore each type of simulations is only done the first time needed and stored for later use in the environment gsi.pStore. With the resampling argument a resampling of the cashed dataset can be forced.

Value

The r* functions deliver a vector (or a matrix of row-vectors) of simulated value of the given distributions. A total of n values (or row vectors) is returned.
The p* functions deliver a vector (of the same length as x) of probabilities for random variable of the given distribution to be under the given quantil values q.
The q* functions deliver a vector of quantiles corresponding to the length of the vector p providing the probabilities.

Note

Unlike the mahalanobis function this function does not be default compute the square of the mahalanobis distance. The pow option is provided if the square is needed.
The package **robustbase** is required for using the robust estimations.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

dist, OutlierClassifier1

Examples

```
rEmpiricalMahalanobis(10,25,2,sorted=TRUE,pow=1,robust=TRUE)
pEmpiricalMahalanobis(qchisq(0.95,df=10),11,1,pow=2,replicates=1000)
(xx<-pMaxMahalanobis(qchisq(0.95,df=10),11,1,pow=2))
qEmpiricalMahalanobis(0.95,11,2)
rMaxMahalanobis(10,25,4)
qMaxMahalanobis(xx,11,1)
```

---

rmult *Simple treatment of real vectors*

Description

A class to collect real multivariate vectors.

Usage

```
 rmult(X,parts=1:NCOL(oneOrDataset(X)),orig=gsi.orig(X),
                missingProjector=attr(X,"missingProjector"),
                V = gsi.getV(X))
 ## S3 method for class 'rmult'
print(x,..., verbose=FALSE)
```

Arguments

X	vector or dataset of numbers considered as elements of a R-vector
parts	vector containing the indices xor names of the columns to be used
x	an rmult object
orig	the original untransformed dataset
missingProjector	the Projector on the observed subspace
V	the *inverse* of the transformation matrix
...	further generic arguments passed to `print.default`
verbose	logical, do you want to get all information about original data and transformation function (if any) with a `print` call? defaults to FALSE (to print strict content only)

Details

The rmult class is a simple convenience class to treat data in the scale of real vectors just like data in the scale of real numbers. A major aspect to take into account is that the internal arithmetic of R is different for these vectors, e.g. mean works as colMeans in a data frame, or matrix-vector operations are done row-wise.

Value

a vector of class "rmult" representing one vector or a matrix of class "rmult", representing multiple vectors by rows.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

+.rmult, scalar, norm.rmult, %*%.rmult, rplus, acomp,

Examples

```
plot(rnorm.rmult(30,mean=0:4,var=diag(1:5)+10))
```

---

rmult-class *Class* "rmult"

Description

The S4-version of the data container "rmult" for compositional data. More information in rmult

Objects from the Class

A virtual Class: No objects may be directly created from it. This is provided to ensure that rmult objects behave as data.frame or structure under certain circumstances. Use rmult to create these objects.

Slots

.Data: Object of class "list" containing the data itself

names: Object of class "character" with column names

row.names: Object of class "data.frameRowLabels" with row names

.S3Class: Object of class "character" with the class string

Extends

Class "data.frame", directly. Class "compositional", directly. Class "list", by class "data.frame", distance 2. Class "oldClass", by class "data.frame", distance 2. Class "vector", by class "data.frame", distance 3.

Methods

coerce `signature(from = "rmult", to = "data.frame")`: to generate a data.frame

coerce `signature(from = "rmult", to = "structure")`: to generate a structure (i.e. a vector, matrix or array)

coerce<- `signature(from = "rmult", to = "data.frame")`: to overwrite a composition with a data.frame

Note

see rmult

Author(s)

Raimon Tolosana-Delgado

References

see rmult

See Also

see rmult

Examples

```
showClass("rmult")
```

---

rmultarithm *vectorial arithmetic for datasets in a classical vector scale*

Description

vector space operations computed for multiple vectors in parallel

Usage

```
## Methods for class "rmult"
## x+y
## x-y
## -x
## x*r
## r*x
## x/r
```

Arguments

x	an rmult vector or dataset of vectors
y	an rmult vector or dataset of vectors
r	a numeric vector of size 1 or nrow(x)

Details

The operators try to mimic the parallel operation of R on vectors of real numbers on vectors of vectors represented as matrices containing the vectors as rows.

Value

an object of class "rmult" containing the result of the corresponding operation on the vectors.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

rmult, %*%.rmult

Examples

```
x <- rmult(matrix( sqrt(1:12), ncol= 3 ))
x
x+x
x + rmult(1:3)
x * 1:4
1:4 * x
x / 1:4
x / 10
```

---

rmultmatmult *inner product for datasets with vector scale*

Description

An rmult object is considered as a sequence of vectors. The %*% is considered as the inner multiplication. An inner multiplication with another vector is the scalar product. an inner multiplication with a matrix is a matrix multiplication, where the rmult-vectors are either considered as row or as column vector.

Usage

```
## S3 method for class 'rmult'
x %*% y
```

Arguments

x	an rmult vector or dataset of vectors, a numeric vector of length (gsi.getD(y)), or a matrix
y	an rmult vector or dataset of vectors , a numeric vector of length (gsi.getD(x)), or a matrix

Details

The operators try to minic the behavior of %*% on c()-vectors as inner product applied in parallel to all vectors of the dataset. Thus the product of a vector with another rmult object or unclassed vector *v* results in the scalar product. For the multiplication with a matrix each vector is considered as a row or column, whatever is more appropriate.

Value

an object of class "rmult" or a numeric vector containing the result of the corresponding inner products.

Note

The product x %*% A %*% y is associative.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

rmult, %*%.rmult

Examples

```
x <- rmult(matrix( sqrt(1:12), ncol= 3 ))
x%*%x
A <- matrix( 1:9,nrow=3)
x %*% A %*% x
x %*% A
A %*% x
x %*% 1:3
x %*% 1:3
1:3 %*% x
```

rnorm *Normal distributions on special spaces*

Description

rnorm.X generates multivariate normal random variates in the space X.

Usage

```
rnorm.acomp(n,mean,var)
rnorm.rcomp(n,mean,var)
rnorm.aplus(n,mean,var)
rnorm.rplus(n,mean,var)
rnorm.rmult(n,mean,var)
rnorm.ccomp(n,mean,var,lambda)
dnorm.acomp(x,mean,var,withJacobian=FALSE)
dnorm.aplus(x,mean,var,withJacobian=FALSE)
dnorm.rmult(x,mean,var)
```

Arguments

n	number of datasets to be simulated
mean	The mean of the dataset to be simulated
var	The variance covariance matrix
lambda	The expected total count
x	vectors in the sampling space
withJacobian	should the jacobian of the log or logratio transformation be included in the density calculations? defaults to FALSE (see details)

Details

The normal distributions in the various spaces dramatically differ. The normal distribution in the rmult space is the commonly known multivariate joint normal distribution. For rplus this distribution has to be somehow truncated at 0. This is here done by setting negative values to 0, i.e. this simulation function produces a sort of multivariate tobit model.
The normal distribution of rcomp is seen as a normal distribution within the simplex as a geometrical portion of the real vector space. The variance is thus forced to be singular and restricted to the affine subspace generated by the simplex. The necessary truncation of negative values is currently done by setting them explicitly to zero and reclosing afterwards, again in the fashion of a tobit model.
The "acomp" and "aplus" are themselves metric vector spaces and thus a normal distribution is defined in them just as in the real space. The resulting distribution almost correspond to multivariate lognormal in the case of "aplus" and Aitchison normal distribution in the simplex in the case of "acomp". These models are equivalent in probability to the multivariate lognormal distribution and the addditive logistic normal distribution respectively, albeit without including the jacobian of the

log or the logratio transformation. If you are interested in the density of the additive logistic normal model, give the extra argument `withJacobian=TRUE`. If you are interested in the multivariate log-normal density cou can either do the same, or better call `dlnorm.rplus`.
Densities are only provided for the models constructed for `rmult`, `aplus` and `acomp` because they do exist w with repect to the Lebesgue measure of each of these spaces. In the other cases it is not possible to compute a measure, since the truncation at zero values produce distributions that are not absolutely continuous with respect to the real, conventional Lebesgue measure.
For count compositions `ccomp` a rnorm.acomp is realized and used as a parameter to a Poisson distribution (see `rpois.ccomp`). So, this is in reality no normal model, but a double stochastic counting process.

Value

a random dataset of the given class generated by a normal distribution with the given mean and variance in the given space. For the density functions `d*`, the value of the probability density at the values of `x` provided

Author(s)

K.Gerald v.d. Boogaart `http://www.stat.boogaart.de`

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

Pawlowsky-Glahn, V. and J.J. Egozcue (2001) Geometric approach to statistical analysis on the simplex. *SERRA* **15**(5), 384-398

Aitchison, J, C. Barcel'o-Vidal, J.J. Egozcue, V. Pawlowsky-Glahn (2002) A consise guide to the algebraic geometric structure of the simplex, the sample space for compositional data analysis, *Terra Nostra*, Schriften der Alfred Wegener-Stiftung, 03/2003

See Also

`runif.acomp`, `rlnorm.rplus`, `rDirichlet.acomp`

Examples

```
MyVar <- matrix(c(
0.2,0.1,0.0,
0.1,0.2,0.0,
0.0,0.0,0.2),byrow=TRUE,nrow=3)
MyMean <- c(1,1,2)

plot(rnorm.acomp(100,MyMean,MyVar))
plot(rnorm.rcomp(100,MyMean,MyVar))
plot(rnorm.aplus(100,MyMean,MyVar))
plot(rnorm.rplus(100,MyMean,MyVar))
```

```
plot(rnorm.rmult(100,MyMean,MyVar))
x <- rnorm.aplus(5,MyMean,MyVar)
dnorm.acomp(x,MyMean,MyVar)
dnorm.aplus(x,MyMean,MyVar)
dnorm.rmult(x,MyMean,MyVar)
```

`robustnessInCompositions` *Handling robustness issues and outliers in compositions.*

Description

The seamless transition to robust estimations in library(compositions).

Details

A statistical method is called nonrobust if an arbitrary contamination of a small portion of the dataset can produce results radically different from the results without the contamination. In this sense many classical procedures relying on distributional models or on moments like mean and variance are highly nonrobust.
We consider robustness as an essential prerequierement of all statistical analysis. However in the context of compositional data analysis robustness is still in its first years.
As of Mai 2008 we provide a new approach to robustness in the package. The central idea is that robustness should be more or less automatic and that there should be no necessity to change the code to compare results obtained from robust procedures and results from there more efficient nonrobust counterparts.
To achieve this all routines that rely on distributional models (such as e.g. mean, variance, principle component analysis, scaling) and routines relying on those routines get a new standard argument of the form:
`fkt(...,robust=getOption("robust"))`
which defaults to a new option "robust". This option can take several values:

FALSE The classical estimators such as arithmetic mean and persons product moment variance are used and the results are to be considered nonrobust.

TRUE The default for robust estimation in the package is used. At this time this is `covMcd` in the **robustbase**-package. This default might change in future.

"pearson" This is a synonym for FALSE and explicitly states that no robustness should be used.

"mcd" Minimum Covariance Determinant. This option explicitly selects the use of `covMcd` in the **robustbase**-package as the main robustness engine.

More options might follow later. To control specific parameters of the model the string can get an attribute named "control" which contains additional options for the robustness engine used. In this moment the control attribute of mcd is a control object of `covMcd`. The control argument of "pearson" is a list containing addition options to the mean, like trim.
The standard value for getOption("robust") is FALSE to avoid situation in which the user thinks he uses a classical technique. Robustness must be switched on explicitly. Either by setting the option

with options(robust=TRUE) or by giving the argument. This default might change later if the authors come to the impression that robust estimation is now considered to be the default.

For those not only interested in avoiding the influence of the outliers, but in an analysis of the outliers we added a subsystem for outlier classification. This subsystem is described in outliersInCompositions and also relies on the robust option. However evidently for these routines the factory default for the robust option is always TRUE, because it is only applicable in an outlieraware context.

We hope that in this way we can provide a seamless transition from nonrobust analysis to a robust analysis.

Note

IMPORTANT: The robust argument only works with the classes of the package. Only your compositional analysis is suddenly robust.
The package **robustbase** is required for using the robust estimations and the outlier subsystem of compositions. To simplify installation it is not listed as required, but it will be loaded, whenever any sort of outlierdetection or robust estimation is used.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

var.acomp, mean.acomp, **robustbase**, compositions-package, missings, outlierplot, OutlierClassifier1, ClusterFinder1

Examples

```
A <- matrix(c(0.1,0.2,0.3,0.1),nrow=2)
Mvar <- 0.1*ilrvar2clr(A%*%t(A))
Mcenter <- acomp(c(1,2,1))
typicalData <- rnorm.acomp(100,Mcenter,Mvar) # main population
colnames(typicalData)<-c("A","B","C")
data5 <- acomp(rbind(unclass(typicalData)+outer(rbinom(100,1,p=0.1)*runif(100),c(0.1,1,2))))

mean(data5)
mean(data5,robust=TRUE)
var(data5)
var(data5,robust=TRUE)
Mvar
biplot(princomp(data5))
biplot(princomp(data5,robust=TRUE))
```

rplus *Amounts i.e. positive numbers analysed as objects of the real vector space*

Description

A class to analyse positive amounts in a classical (non-logarithmic) framework.

Usage

```
rplus(X, parts=1:NCOL(oneOrDataset(X)), total=NA, warn.na=FALSE,
      detectionlimit=NULL, BDL=NULL, MAR=NULL, MNAR=NULL, SZ=NULL)
```

Arguments

X	vector or dataset of positive numbers considered as amounts
parts	vector containing the indices xor names of the columns to be used
total	a numeric vectors giving the total amount of each dataset
warn.na	should the user be warned in case of NA,NaN or 0 coding different types of missing values?
detectionlimit	a number, vector or matrix of positive numbers giving the detection limit of all values, all columns or each value, respectively
BDL	the code for 'Below Detection Limit' in X
SZ	the code for 'Structural Zero' in X
MAR	the code for 'Missing At Random' in X
MNAR	the code for 'Missing Not At Random' in X

Details

Many multivariate datasets essentially describe amounts of D different parts in a whole. When the whole is large in relation to the considered parts, such that they do not exclude each other, and when the total amount of each componenten is actually determined by the phenomenon under investigation and not by sampling artifacts (such as dilution or sample preparation) then the parts can be treated as amounts rather than as a composition (cf. rcomp, aplus).
In principle, amounts are just real-scaled numbers with the single restriction that they are nonnegative. Thus they can be analysed by any multivariate analysis method. This class provides a simple access interface to do so. It tries to keep in mind the positivity property of amounts and the special point zero. However there are strong arguments why an analyis based on log-scale might be much more adapted to the problem. This log-approach is provided by the class aplus.

The classes rcomp, acomp, aplus, and rplus are designed in a fashion as similar as possible in order to allow direct comparison between results obtained by the different approaches. In particular, the aplus logistic transform ilt is mirrored by the simple identity transform iit. In terms of computer science, this identity mapping is actually mapping an object of type "rplus" to a class-less datamatrix.

Value

a vector of class "rplus" representing a vector of amounts or a matrix of class "rplus" representing multiple vectors of amounts, by rows.

Missing Policy

Missing and Below Detecion Limit Policy is in mored detailed explained in compositions.missing.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

van den Boogaart, K.G. and R. Tolosana-Delgado (2008) "compositions": a unified R package to analyze Compositional Data, *Computers & Geosciences*, 34 (4), pages 320-338, doi: 10.1016/j.cageo.2006.11.017.

See Also

iit,rcomp, aplus, princomp.rplus, plot.rplus, boxplot.rplus, barplot.rplus, mean.rplus, var.rplus, variation.rplus, cov.rplus, msd

Examples

```
data(SimulatedAmounts)
plot(rplus(sa.lognormals))
```

---

rplus-class *Class* "rplus"

Description

The S4-version of the data container "rplus" for compositional data. More information in rplus

Objects from the Class

A virtual Class: No objects may be directly created from it. This is provided to ensure that rplus objects behave as data.frame or structure under certain circumstances. Use rplus to create these objects.

Slots

.Data: Object of class "list" containing the data itself

names: Object of class "character" with column names

row.names: Object of class "data.frameRowLabels" with row names

.S3Class: Object of class "character" with the class string

Extends

Class "data.frame", directly. Class "compositional", directly. Class "list", by class "data.frame", distance 2. Class "oldClass", by class "data.frame", distance 2. Class "vector", by class "data.frame", distance 3.

Methods

coerce `signature(from = "rplus", to = "data.frame")`: to generate a data.frame

coerce `signature(from = "rplus", to = "structure")`: to generate a structure (i.e. a vector, matrix or array)

coerce<- `signature(from = "rplus", to = "data.frame")`: to overwrite a composition with a data.frame

Note

see rplus

Author(s)

Raimon Tolosana-Delgado

References

see rplus

See Also

see rplus

Examples

```
showClass("rplus")
```

`rplusarithm` *vectorial arithmetic for data sets with rplus class*

Description

The positive quadrant forms a manifold of the real vector space. The induced operations +,-,*,/ give results valued in this real vector space (not necessarily inside the manifold).

Usage

```
mul.rplus(x,r)
## Methods for class rplus
##   x+y
##  x-y
##   -x
##   x*r
##   r*x
##   x/r
```

Arguments

x	an rplus composition or dataset of compositions
y	an rplus composition or dataset of compositions
r	a numeric vector of size 1 or nrow(x)

Details

The functions behave quite like +.rmult.

Value

rmult-objects containing the given operations on the rcomp manifold as subset of the R^D. Only the addition and multiplication with positive numbers are internal operation and results in an rplus-object again.

Note

For * the arguments x and y can be exchanged.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

+.rmult, +.acomp,cpt, rcomp, rmult

Examples

```
rplus(1:5)* -1 + rplus(1:5)
data(SimulatedAmounts)
cdata <- rplus(sa.lognormals)
plot( tmp <- (cdata-mean(cdata))/msd(cdata) )
class(tmp)
mean(tmp)
msd(tmp)
var(tmp)
```

rpois *Simulate count compositions without overdispersion*

Description

Generates multinomial or multi-Poission random variates based on an Aitchison composition.

Usage

```
rpois.ccomp(n,p,lambda)
rmultinom.ccomp(n,p,N)
```

Arguments

n	number of datasets to be simulated
p	The composition representing the probabilites/portions of the individual parts
lambda	scalar or vector giving the expected total count
N	scalar or vector giving the total count

Details

A count composition is a realisation of a multinomial or multivariate Poisson distribution.

Value

a random dataset ccount dataset

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

rnorm.ccomp

Examples

```
p <- acomp(c(3,3,3))
rpois.ccomp(10,p,40)
rmultinom.ccomp(10,p,40)
```

runif *The uniform distribution on the simplex*

Description

Generates random compositions with a uniform distribution on the (rcomp) simplex.

Usage

```
runif.acomp(n,D)
runif.rcomp(n,D)
```

Arguments

n	number of datasets to be simulated
D	number of parts

Value

a generated random dataset of class "acomp" or "rcomp" drawn from a uniform distribution on the simplex of D parts.

Note

The only difference between both routines is the class of the dataset returned.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

See Also

rDirichlet.acomp

Examples

```
plot(runif.acomp(10,3))
plot(runif.rcomp(10,3))
```

scalar *Parallel scalar products*

Description

Computes scalar products of datasets of vectors or vectorial quantities.

Usage

```
scalar(x,y)
## Default S3 method:
scalar(x,y)
```

Arguments

x	a vector or a matrix with rows considered as vectors
y	a vector or a matrix with rows considered as vectors

Details

The scalar product of two vectors is defined as:

$$scalar(x,y) := \sum(x_i y_i)$$

Value

a numerical vector containing the scalar products of the vectors given by x and y. If both x and y contain more than one vector the function uses parallel operation like it would happen with an ordinary product of vectors.

Note

The computation of the scalar product implicitly applies the cdt transform, which implies that the scalar products corresponding to the given geometries are returned for acomp, rcomp, aplus, rplus-objects. Even a useful scalar product for factors is induced in this way.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

Examples

```
scalar(acomp(c(1,2,3)),acomp(c(1,2,3)))
scalar(rmult(c(1,2,3)),rmult(c(1,2,3)))
```

scale *Normalizing datasets by centering and scaling*

Description

The dataset is standardized by optional scaling and centering.

Usage

```
scale(x, center = TRUE, scale = TRUE,...)
## Default S3 method:
scale(x,center=TRUE, scale=TRUE,...)
## S3 method for class 'acomp'
scale(x,center=TRUE, scale=TRUE,...,robust=getOption("robust"))
## S3 method for class 'rcomp'
scale(x,center=TRUE, scale=TRUE,...,robust=getOption("robust"))
## S3 method for class 'aplus'
scale(x,center=TRUE, scale=TRUE,...,robust=getOption("robust"))
## S3 method for class 'rplus'
scale(x,center=TRUE, scale=TRUE,...,robust=getOption("robust"))
## S3 method for class 'rmult'
scale(x,center=TRUE, scale=TRUE,...,robust=getOption("robust"))
```

Arguments

x	a dataset or a single vector of some type
center	logical value or the center to be substracted.
scale	logical value or a scaling factor to for multiplication.
robust	A robustness description. See robustnessInCompositions for details.
...	added for generic generality

Details

scaling is defined in different ways for the different data types. It is always performed as an operation in the enclosing vector space. In all cases an independent scaling of the different coordinates is not always appropriate. This is only done for rplus and rmult geometries. The other three geometries are treated with a global scaling, keeping the relative variations of every part/amount.

The scaling factors can be a matrix (for cdt or idt space), a scalar, or for the r* geometries vector for scaling the entries individually. However scaling the entries individually does not make sense in the a* geometries. The operation achieve in the r*-geometries is indeed the centering of the a*-geometries.

Value

a vector or data matrix, as x and with the same class, but acordingly transformed.

Note

Note that the "rcomp" and "rplus" objects does not preserve their geometry during scaling and are therefore reported as "rmult" objects.

See the documentation in package base for details on

base::scale and base::scale.default These functions are only modified to allow the additional robustness parameter.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

split{base}

Examples

```
  data(SimulatedAmounts)
  plot(scale(acomp(sa.groups)))
  ## Not run:
plot(scale(rcomp(sa.groups)))

## End(Not run)
  plot(scale(aplus(sa.groups)))
  ## Not run:
plot(scale(rplus(sa.groups)))

## End(Not run)
  plot(scale(rmult(sa.groups)))
```

Sediments *Proportions of sand, silt and clay in sediments specimens*

Description

Data provide sand, silt and clay compositions of 21 sediments specimens, 10 of which are identified as *offshore,* 7 as *near shore* and 4 new samples.

Usage

```
data(Sediments)
```

Format

sand numeric: the portion of sand

silt numeric: the protion of silt

clay numeric: the portion of clay

type numeric: 1 for offshore, 2 for near shore and 3 for new samples

Details

The data comprise 21 cases: 10 offshore, 7 near shore and 4 new samples, and 4 variables: sand, silt and clay proportions and in addition the type of sediments specimens – 1 for offshore, 2 for near shore and 3 for new samples.

All 3-part compositions sum to one.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name YATQUAD.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison, J. (1986) The Statistical Analysis of Compositional Data, (Data 20) pp17.

`segments` *Draws straight lines from point to point.*

Description

The function draws lines from a points x0 to a point y1 in the given geometry.

Usage

```
          segments(x0,...)
          ## Default S3 method:
segments(x0,...)
          ## S3 method for class 'acomp'
segments(x0,y1,...,steps=30,aspanel=FALSE)
          ## S3 method for class 'rcomp'
segments(x0,y1,...,steps=30,aspanel=FALSE)
          ## S3 method for class 'aplus'
segments(x0,y1,...,steps=30,aspanel=FALSE)
          ## S3 method for class 'rplus'
segments(x0,y1,...,steps=30,aspanel=FALSE)
          ## S3 method for class 'rmult'
segments(x0,y1,...,steps=30,aspanel=FALSE)
```

Arguments

x0	dataset of points (of the given type) to draw the line from
y1	dataset of points (of the given type) to draw the line to
...	further graphical parameters
steps	the number of discretisation points to draw the segments, since the representation might not visually be a straight line
aspanel	Logical, indicates use as slave to do acutal drawing only.

Details

The default ‘segments.default(x0,...)‘ redirects to ‘segments‘ in package "graphics".

The other methods add lines to the graphics generated with the corresponding plot functions of "compositions"-classes.

Adding to multipaneled plots redraws the plot completely, and is only possible when the plot has been created with the plotting routines from this library.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

plot.acomp,lines.acomp

Examples

```
data(SimulatedAmounts)

plot(acomp(sa.lognormals))
segments.acomp(acomp(c(1,2,3)),acomp(c(2,3,1)),col="red")
segments.rcomp(acomp(c(1,2,3)),acomp(c(2,3,1)),col="blue")

plot(aplus(sa.lognormals[,1:2]))
segments.aplus(aplus(c(10,20)),aplus(c(20,10)),col="red")
segments.rplus(rplus(c(10,20)),rplus(c(20,10)),col="blue")

plot(rplus(sa.lognormals[,1:2]))
segments.aplus(aplus(c(10,20)),aplus(c(20,10)),col="red")
segments.rplus(rplus(c(10,20)),rplus(c(20,10)),col="blue")
```

SerumProtein *Serum Protein compositions of blood samples*

Description

Data recording the proportions of the 4 serum proteins from blood samples of 30 patients, 14 with known disease A, 16 with known disease B, and 6 new cases.

Usage

```
data(SerumProtein)
```

Format

a numeric a protein type

b numeric a protein type

c numeric a protein type

d numeric a protein type

Type 1 deasease A, 2 disease B, 3 new cases

Details

The data consist of 36 cases: 14 with known disease A, 16 with known disease B, and 6 new cases and 5 v variables: a, b, c, and d for 4 serum proteins and Type for the diseases: 1 for disease A, 2 for disease B, and 3 for new cases. All row serum proteins proportions sums to 1 except some rounding errors.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name SERPROT.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison, J. (1986) The Statistical Analysis of Compositional Data (Data 16) pp20.

ShiftOperators *Shifts of machine operators*

Description

Compsitions of eight-hours shifts of 27 machine operators.

Usage

```
data(ShiftOperators)
```

Details

A study of the activities of 27 machine operators during their eight-hours shifts has been conducted, and proportions of time spend in the following categories:

A: high-quality production,
B: low-quality production,
C: machine setting,
D: machine repair,

are recorded. Of particular interest are any insights which such data might give of relationships between productive and nonproductive parts of such shifts. All compositions sum up one except for rounding error.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name SHIFT.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison, J. (1986) The Statistical Analysis of Compositional Data, (Data 22), pp22.

simplemissingplot *Ternary diagrams*

Description

Displaying compositions in ternary diagrams

Usage

```
simpleMissingSubplot(loc, portions, labels=NULL,
          col=c("white","yellow","red","green","blue"),
          ..., border="gray60", vertical=NULL, xpd=NA)
```

Arguments

loc	a vector of the form c(x1,x2,y1,y2) giving the drawing rectangle for the subplot in coordinates as in par("usr"). I.e. if the plot is logrithmic the base 10 logarithm is to be used: **x1** left boundary of drawing rectangle **y1** lower boundary of drawing rectangle **x2** right boundary of drawing rectangle **y2** upper boundary of drawing rectangle
portions	The portions of different missing categories
labels	The labels for the categories.
col	The colors to plot the different categories.
...	further graphical parameters passed to text
border	The color to draw the borders of the rectangles.
vertical	Should a horizontal or a vertical plot be produced. If NULL the choice is done automatically according to the size of the recangle provided.
xpd	extended plot region. See par("xpd").

Details

This function is typically not called directly,however it could in principle be used to add to plots. The user will modify the function call only to modify the appearance of the missing plot.

The labels are only plotted for nonzero portions. In this way it is always possible to realize the presence of a given missing type, even if it is a too small portion to be actually displayed. In case of overplotting of different labels a further investigation using missingSummary should be used.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

plot.aplus

Examples

```
data(SimulatedAmounts)
plot(acomp(sa.missings))
plot(acomp(sa.missings),mp=~simpleMissingSubplot(c(0,0.1,0.2,1),
   missingInfo[c(1,3:5,2)],
   c("Not Missing",paste("Missing Only:",cn),"Totally Missing"),
```

```
   col=c("gray","red","green","blue","darkgray"))
)
ms <- missingSummary(sa.missings)
for( i in 1:3 )
  simpleMissingSubplot(c(0.9+0.03*(i-1),0.9+0.03*i,0.2,1), ms[i,])
```

SimulatedAmounts *Simulated amount datasets*

Description

Several simulated datasets intended as reference examples for various conceptual and statistical models of compositions and amounts.

Usage

```
data(SimulatedAmounts)
```

Format

Data matrices with 60 cases and 3 or 5 variables.

Details

The statistical analysis of amounts and compositions is set to discussion. Four essentially different approaches are provided in this package around the classes "rplus", "aplus", "rcomp", "acomp". There is no absolutely "right" approach, since there is a conection between these approaches and the processes originating the data. We provide here simulated standard datasets and the corresponding simulation procedures following these several models to provide "good" analysis examples and to show how these models actually look like in data.

The data sets are simulated according to correlated lognormal distributions (sa.lognormals, sa.lognormal5), winsorised correlated normal distributions (sa.tnormals, sa.tnormal5), Dirichlet distribution on the simplex (sa.dirichlet, sa.dirichlet5), uniform distribution on the simplex (sa.uniform, sa.uniform5), and a grouped dataset (sa.groups, sa.groups5) with three groups (given in sa.groups.area and sa.groups5.area) all distributed accordingly with a lognormal distribution with group-dependent means.

We can imagine that amounts evolve in nature e.g. in part of the soil they are diluted and transported in a transport medium, usually water, which comes from independent source (the rain, for instance) and this new composition is normalized by taking a sample of standard size. For each of the datasets *sa.X* there is a corresponding *sa.X*`.dil` dataset which is build by simulating exactly that process on the corresponding *sa.X* dataset . The amounts in the *sa.X*`.dil` are given in ppm. This idea of a transport medium is a major argument for a compositional approach, because the total amount given by the sum of the parts is induced by the dilution given by the medium and thus non-informative for the original process investigated.

If we imagine now these amounts flowing into a river and sedimenting, the different contributions are accumulated along the river and renormalized to a unit portion on taking samples again. For each of the dataset *sa.X*`.dil` there is a corresponding *sa.X*`.mix` dataset which is built from the corresponding *sa.X* dataset by simulating exactly that accumulation process. Mixing of different compositions is a major argument against the log based approaches (`aplus`, `acomp`) since mixing is a highly nonlinear operation in terms of log-ratios.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

Source

The datasets are simulated for this package and are under the GNU Public Library Licence Version 2 or newer.

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

Rehder, S. and U. Zier (2001) Letter to the Editor: Comment on "Logratio Analysis and Compositional Distance" by J. Aitchison, C. Barcel\'o -Vidal, J.A. Mart\'in-Fern\'andez and V. Pawlowsky-Glahn, *Mathematical Geology*, **33** (7), 845-848.

Zier, U. and S. Rehder (2002) Some comments on log-ratio transformation and compositional distance, *Terra Nostra*, Schriften der Alfred Wegener-Stiftung, 03/2003

Examples

```
data(SimulatedAmounts)
plot.acomp(sa.lognormals)
plot.acomp(sa.lognormals.dil)
plot.acomp(sa.lognormals.mix)
plot.acomp(sa.lognormals5)
plot.acomp(sa.lognormals5.dil)
plot.acomp(sa.lognormals5.mix)

plot(acomp(sa.missings))
plot(acomp(sa.missings5))

#library(MASS)
plot.rcomp(sa.tnormals)
plot.rcomp(sa.tnormals.dil)
plot.rcomp(sa.tnormals.mix)
plot.rcomp(sa.tnormals5)
plot.rcomp(sa.tnormals5.dil)
plot.rcomp(sa.tnormals5.mix)
```

```
plot.acomp(sa.groups,col=as.numeric(sa.groups.area),pch=20)
plot.acomp(sa.groups.dil,col=as.numeric(sa.groups.area),pch=20)
plot.acomp(sa.groups.mix,col=as.numeric(sa.groups.area),pch=20)
plot.acomp(sa.groups5,col=as.numeric(sa.groups.area),pch=20)
plot.acomp(sa.groups5.dil,col=as.numeric(sa.groups.area),pch=20)
plot.acomp(sa.groups5.mix,col=as.numeric(sa.groups.area),pch=20)

plot.acomp(sa.uniform)
plot.acomp(sa.uniform.dil)
plot.acomp(sa.uniform.mix)
plot.acomp(sa.uniform5)
plot.acomp(sa.uniform5.dil)
plot.acomp(sa.uniform5.mix)

plot.acomp(sa.dirichlet)
plot.acomp(sa.dirichlet.dil)
plot.acomp(sa.dirichlet.mix)
plot.acomp(sa.dirichlet5)
plot.acomp(sa.dirichlet5.dil)
plot.acomp(sa.dirichlet5.mix)

# The data was simulated with the following commands:

#library(MASS)
dilution <- function(x) {clo(cbind(x,exp(rnorm(nrow(x),5,1))))[,1:ncol(x)]*1E6}
seqmix   <- function(x) {clo(apply(x,2,cumsum))*1E6}


vars  <- c("Cu","Zn","Pb")
vars5 <- c("Cu","Zn","Pb","Cd","Co")

sa.lognormals <- structure(exp(matrix(rnorm(3*60),ncol=3) %*%
                               chol(matrix(c(1,0.8,-0.2,0.8,1,
                                             -0.2,-0.2,-0.2,1),ncol=3))+
                               matrix(rep(c(1:3),each=60),ncol=3)),
                           dimnames=list(NULL,vars))

plot.acomp(sa.lognormals)
pairs(sa.lognormals)

sa.lognormals.dil <- dilution(sa.lognormals)
plot.acomp(sa.lognormals.dil)
pairs(sa.lognormals.dil)

sa.lognormals.mix <- seqmix(sa.lognormals.dil)
plot.acomp(sa.lognormals.mix)
pairs(sa.lognormals.mix)


sa.lognormals5 <- structure(exp(matrix(rnorm(5*60),ncol=5) %*%
                               chol(matrix(c(1,0.8,-0.2,0,0,
                                             0.8,1,-0.2,0,0,
                                             -0.2,-0.2,1,0,0,
```

```
                                        0,0,0,5,4.9,
                                        0,0,0,4.9,5),ncol=5))+
                                matrix(rep(c(1:3,-2,-2),each=60),ncol=5)),
                            dimnames=list(NULL,vars5))

plot.acomp(sa.lognormals5)
pairs(sa.lognormals5)

sa.lognormals5.dil <- dilution(sa.lognormals5)
plot.acomp(sa.lognormals5.dil)
pairs(sa.lognormals5.dil)

sa.lognormals5.mix <- seqmix(sa.lognormals5.dil)
plot.acomp(sa.lognormals5.mix)
pairs(sa.lognormals5.mix)



sa.groups.area <- factor(rep(c("Upper","Middle","Lower"),each=20))
sa.groups <- structure(exp(matrix(rnorm(3*20*3),ncol=3) %*%
                                chol(0.5*matrix(c(1,0.8,-0.2,0.8,1,
                                        -0.2,-0.2,-0.2,1),ncol=3))+
                                matrix(rep(c(1,2,2.5,2,2.9,5,4,2,5),
                                        each=20),ncol=3)),
                            dimnames=list(NULL,c("clay","sand","gravel")))

plot.acomp(sa.groups,col=as.numeric(sa.groups.area),pch=20)
pairs(sa.lognormals,col=as.numeric(sa.groups.area),pch=20)

sa.groups.dil <- dilution(sa.groups)
plot.acomp(sa.groups.dil,col=as.numeric(sa.groups.area),pch=20)
pairs(sa.groups.dil,col=as.numeric(sa.groups.area),pch=20)

sa.groups.mix <- seqmix(sa.groups.dil)
plot.acomp(sa.groups.mix,col=as.numeric(sa.groups.area),pch=20)
pairs(sa.groups.mix,col=as.numeric(sa.groups.area),pch=20)



sa.groups5.area <- factor(rep(c("Upper","Middle","Lower"),each=20))
sa.groups5 <- structure(exp(matrix(rnorm(5*20*3),ncol=5) %*%
                                chol(matrix(c(1,0.8,-0.2,0,0,
                                        0.8,1,-0.2,0,0,
                                        -0.2,-0.2,1,0,0,
                                        0,0,0,5,4.9,
                                        0,0,0,4.9,5),ncol=5))+
                                matrix(rep(c(1,2,2.5,
                                        2,2.9,5,
                                        4,2.5,0,
                                        -2,-1,-1,
                                        -1,-2,-3),
                                        each=20),ncol=5)),
                            dimnames=list(NULL,
```

```
                              vars5))

plot.acomp(sa.groups5,col=as.numeric(sa.groups5.area),pch=20)
pairs(sa.groups5,col=as.numeric(sa.groups5.area),pch=20)

sa.groups5.dil <- dilution(sa.groups5)
plot.acomp(sa.groups5.dil,col=as.numeric(sa.groups5.area),pch=20)
pairs(sa.groups5.dil,col=as.numeric(sa.groups5.area),pch=20)

sa.groups5.mix <- seqmix(sa.groups5.dil)
plot.acomp(sa.groups5.mix,col=as.numeric(sa.groups5.area),pch=20)
pairs(sa.groups5.mix,col=as.numeric(sa.groups5.area),pch=20)



sa.tnormals <- structure(pmax(matrix(rnorm(3*60),ncol=3) %*%
                                chol(matrix(c(1,0.8,-0.2,0.8,1,
                                              -0.2,-0.2,-0.2,1),ncol=3))+
                                matrix(rep(c(0:2),each=60),ncol=3),0),
                           dimnames=list(NULL,c("clay","sand","gravel")))

plot.rcomp(sa.tnormals)
pairs(sa.tnormals)

sa.tnormals.dil <- dilution(sa.tnormals)
plot.acomp(sa.tnormals.dil)
pairs(sa.tnormals.dil)

sa.tnormals.mix <- seqmix(sa.tnormals.dil)
plot.acomp(sa.tnormals.mix)
pairs(sa.tnormals.mix)



sa.tnormals5 <- structure(pmax(matrix(rnorm(5*60),ncol=5) %*%
                                chol(matrix(c(1,0.8,-0.2,0,0,
                                              0.8,1,-0.2,0,0,
                                              -0.2,-0.2,1,0,0,
                                              0,0,0,0.05,0.049,
                                              0,0,0,0.049,0.05),ncol=5))+
                                matrix(rep(c(0:2,0.1,0.1),each=60),ncol=5),0),
                           dimnames=list(NULL,
                             vars5))

plot.rcomp(sa.tnormals5)
pairs(sa.tnormals5)

sa.tnormals5.dil <- dilution(sa.tnormals5)
plot.acomp(sa.tnormals5.dil)
pairs(sa.tnormals5.dil)

sa.tnormals5.mix <- seqmix(sa.tnormals5.dil)
plot.acomp(sa.tnormals5.mix)
```

```
pairs(sa.tnormals5.mix)



sa.dirichlet <- sapply(c(clay=0.2,sand=2,gravel=3),rgamma,n=60)
colnames(sa.dirichlet) <- vars

plot.acomp(sa.dirichlet)
pairs(sa.dirichlet)

sa.dirichlet.dil <- dilution(sa.dirichlet)
plot.acomp(sa.dirichlet.dil)
pairs(sa.dirichlet.dil)

sa.dirichlet.mix <- seqmix(sa.dirichlet.dil)
plot.acomp(sa.dirichlet.mix)
pairs(sa.dirichlet.mix)



sa.dirichlet5 <- sapply(c(clay=0.2,sand=2,gravel=3,humus=0.1,plant=0.1),rgamma,n=60)
colnames(sa.dirichlet5) <- vars5

plot.acomp(sa.dirichlet5)
pairs(sa.dirichlet5)

sa.dirichlet5.dil <- dilution(sa.dirichlet5)
plot.acomp(sa.dirichlet5.dil)
pairs(sa.dirichlet5.dil)

sa.dirichlet5.mix <- seqmix(sa.dirichlet5.dil)
plot.acomp(sa.dirichlet5.mix)
pairs(sa.dirichlet5.mix)


sa.uniform   <- sapply(c(clay=1,sand=1,gravel=1),rgamma,n=60)
colnames(sa.uniform) <- vars

plot.acomp(sa.uniform)
pairs(sa.uniform)

sa.uniform.dil <- dilution(sa.uniform)
plot.acomp(sa.uniform.dil)
pairs(sa.uniform.dil)

sa.uniform.mix <- seqmix(sa.uniform.dil)
plot.acomp(sa.uniform.mix)
pairs(sa.uniform.mix)



sa.uniform5   <- sapply(c(clay=1,sand=1,gravel=1,humus=1,plant=1),rgamma,n=60)
colnames(sa.uniform5) <- vars5
```

```
plot.acomp(sa.uniform5)
pairs(sa.uniform5)

sa.uniform5.dil <- dilution(sa.uniform5)
plot.acomp(sa.uniform5.dil)
pairs(sa.uniform5.dil)

sa.uniform5.mix <- seqmix(sa.uniform5.dil)
plot.acomp(sa.uniform5.mix)
pairs(sa.uniform5.mix)

tmp<-set.seed(1400)
A <- matrix(c(0.1,0.2,0.3,0.1),nrow=2)
Mvar <- 0.1*ilrvar2clr(A %*% t(A))
Mcenter <- acomp(c(1,2,1))
typicalData <- rnorm.acomp(100,Mcenter,Mvar) # main population
colnames(typicalData)<-c("A","B","C")
# A dataset without outliers
sa.outliers1 <- acomp(rnorm.acomp(100,Mcenter,Mvar))
# A dataset with 10% data with a large error in the first component
sa.outliers2 <- acomp(rbind(typicalData+rbinom(100,1,p=0.1)*rnorm(100)*acomp(c(4,1,1))))
# A dataset with a single outlier
sa.outliers3 <- acomp(rbind(typicalData,acomp(c(0.5,1.5,2))))
colnames(sa.outliers3)<-colnames(typicalData)
tmp<-set.seed(30)
rcauchy.acomp <- function (n, mean, var){
  D <- gsi.getD(mean)-1
 perturbe(ilrInv(matrix(rnorm(n*D)/rep(rnorm(n),D), ncol = D) %*% chol(clrvar2ilr(var))), mean)
}
# A dataset with a Cauchy type distribution
sa.outliers4 <- acomp(rcauchy.acomp(100,acomp(c(1,2,1)),Mvar/4))
colnames(sa.outliers4)<-colnames(typicalData)
# A dataset with like sa.outlier2 but a differently strong distortions
sa.outliers5 <- acomp(rbind(unclass(typicalData)+outer(rbinom(100,1,p=0.1)*runif(100),c(0.1,1,2))))
# A dataset with a second population
sa.outliers6 <- acomp(rbind(typicalData,rnorm.acomp(20,acomp(c(4,4,1)),Mvar)))

# Missings
sa.missings <- simulateMissings(sa.lognormals,dl=0.05,MAR=0.05,MNAR=0.05,SZ=0.05)
sa.missings[5,2]<-BDLvalue

sa.missings5 <- simulateMissings(sa.lognormals5,dl=0.05,MAR=0.05,MNAR=0.05,SZ=0.05)
sa.missings5[5,2]<-BDLvalue


objects(pattern="sa.*")
```

---

simulatemissings *Artifical simulation of various kinds of missings/polluted data*

Description

These are simulation mechanisms to check that missing techniques perform in sensible ways. They just generate additional missings of the various types in a given dataset, according to a specific process.

Usage

```
simulateMissings(x, dl=NULL, knownlimit=FALSE,
     MARprob=0.0, MNARprob=0.0, mnarity=0.5, SZprob=0.0)
observeWithAdditiveError(x, sigma=dl/dlf, dl=sigma*dlf, dlf=3,
     keepObs=FALSE, digits=NA, obsScale=1,
     class="acomp")
```

Arguments

x	a dataset that should get the missings
dl	the detection limit described in clo, to impose an artificial detection limit
knownlimit	a boolean indicating wether the actual detection limit is still known in the dataset.
MARprob	the probability of occurence of 'Missings At Random' values
MNARprob	the probability of occurrence of 'Missings Not At Random'. The tendency is that small values have a higher probability to be missed.
mnarity	a number between 0 and 1 giving the strength of the influence of the actual value in becoming a MNAR. 0 means a MAR like behavior and 1 means that it is just the smallest values that is lost
SZprob	the probability to obtain a structural zero. This is done at random like a MAR.
sigma	the standard deviation of the normal distributed extra additive error
dlf	the distance from 0 at which a datum will be considered BDL
keepObs	should the (closed) data without additive error be returned as an attribute?
digits	rounding to be applied to the data with additive error (see Details)
obsScale	rounding to be applied to the data with additive error (see Details). Should be a power of 10.
class	class of the output object

Details

Without any additional parameters no missings are generated. The procedure to generate MNAR affects all variables.

Function "simulateMissings" is a multipurpose simulator, where each class of missing value is treated separately, and where detection limits are specified as thresholds.

Function "observeWithAdditiveError" simulates data within a very specific framework, where an additive error of sd=sigma is added to the input data x, and BDLs are generated if a datum is less than dfl times sigma. Afterwards, the resulting data are rounded as round(data/obsScale,digits)*obsScale, i.e. a certain observation scale obsScale is chosen, and at that scale, only some digits are kept. This framework is typical of chemical analyses, and it generates both BDLs and pollution/rounding of (apparently) "right" data.

Value

A dataset like x but with some additional missings.

Author(s)

K.Gerald van den Boogaart

References

van den Boogaart, K., R. Tolosana-Delgado, and M. Bren (2011). The Compositional Meaning of a Detection Limit. In Proceedings of the 4th International Workshop on Compositional Data Analysis (2011).

van den Boogaart, K.G., R. Tolosana-Delgado and M. Templ (2014) Regression with compositional response having unobserved components or below detection limit values. Statistical Modelling (in press).

See compositions.missings for more details.

See Also

compositions.missings

Examples

```
data(SimulatedAmounts)
x <- acomp(sa.lognormals)
xnew <- simulateMissings(x,dl=0.05,MAR=0.05,MNAR=0.05,SZ=0.05)
acomp(xnew)
plot(missingSummary(xnew))
```

Skulls *Measurement of skulls*

Description

Measurement in degrees of the angles N, A, B in skulls of English seventeenth-century people and Naquada people.

Usage

```
data(Skulls)
```

Details

As a part of a study of seventeenth-century English skulls three angles of a triangle in the cranium

N: nasial angle,
A: alveolar angle,
B: basilar angle,

were measured for 22 female and 29 male skulls. These, together with similar measurement of 22 female and 29 male skulls of the Naqada race are presented. The general objective is to investigate possible sex and race differences in skull shape. The angles sums in the row are all equal to 180 degrees.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name SKULLS.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison, J. (1986) The Statistical Analysis of Compositional Data, (Data 24), pp22.

`SkyeAFM` *AFM compositions of 23 aphyric Skye lavas*

Description

AFM compositions of 23 aphyric Skye lavas. AFM diagrams formed from the relative proportions of A: alkali or Na2O + K2O, F: Fe2O3, and M: MgO, are common in geochemistry.

Usage

```
data(SkyeAFM)
```

Details

AFM compositions of 23 aphyric Skye lavas. AFM diagrams formed from the relative proportions of tive proportions of A: alkali or Na2O + K2O, F: Fe2O3, and M: MgO. Adapted from Thompson, Esson and Duncan: Major element chemical variations in the Eocene lavas of the Isle of Skye, Scotland. All row percentage sums to 100.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name SKYEAFM.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison, J. (1986) The Statistical Analysis of Compositional Data, (Data 6) pp12.

Thompson, Esson and Duncan: Major element chemical variations in the Eocene lavas of the Isle of Skye, Scotland. 1972, J.Petrology, 13, 219-235.

See Also

Skye

`split` *Splitting datasets in groups given by factors*

Description

Splits data sets of compositions in groups given by factors, and gives the same class as the data to the result.

Usage

```
## S3 method for class 'acomp'
split(x,f,drop=FALSE,...)
## S3 method for class 'rcomp'
split(x,f,drop=FALSE,...)
## S3 method for class 'aplus'
split(x,f,drop=FALSE,...)
## S3 method for class 'rplus'
split(x,f,drop=FALSE,...)
## S3 method for class 'rmult'
split(x,f,drop=FALSE,...)
## S3 method for class 'ccomp'
split(x,f,drop=FALSE,...)
```

Arguments

`x`	a dataset or a single vector of some type
`f`	a factor that defines the grouping or a list of factors
`drop`	drop=FALSE also gives (empty) datsets for empty categories
`...`	Further arguments passed to split.default. Currently (and probably) without any use.

Value

a list of objects of the same type as `x`.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

split

Examples

```
data(SimulatedAmounts)
split(acomp(sa.groups),sa.groups.area)
lapply( split(acomp(sa.groups),sa.groups.area), mean)
```

---

straight *Draws straight lines.*

Description

The function draws lines in a given direction d through points x.

Usage

```
          straight(x,...)
          ## S3 method for class 'acomp'
straight(x,d,...,steps=30,aspanel=FALSE)
          ## S3 method for class 'rcomp'
straight(x,d,...,steps=30,aspanel=FALSE)
          ## S3 method for class 'aplus'
straight(x,d,...,steps=30,aspanel=FALSE)
          ## S3 method for class 'rplus'
straight(x,d,...,steps=30,aspanel=FALSE)
          ## S3 method for class 'rmult'
straight(x,d,...,steps=30,aspanel=FALSE)
```

Arguments

x	dataset of points of the given type to draw the line through
d	dataset of directions of the line
...	further graphical parameters
steps	the number of discretisation points to draw the segments, since the representation might not visually be a straight line
aspanel	Logical, indicates use as slave to do acutal drawing only.

Details

The functions add lines to the graphics generated with the corresponding plot functions.
Adding to multipaneled plots redraws the plot completely, and is only possible when the plot has been created with the plotting routines from this library.
Lines end when they leave the space (e.g. the simplex), which sometimes leads to the impression of premature end (specially in rcomp geometry).

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

plot.acomp,lines.acomp

Examples

```
data(SimulatedAmounts)

plot(acomp(sa.lognormals))
straight(mean(acomp(sa.lognormals)),
         princomp(acomp(sa.lognormals))$Loadings[1,],
         col="red")
straight(mean(rcomp(sa.lognormals)),
         princomp(rcomp(sa.lognormals))$loadings[,1],
         col="blue")

plot(aplus(sa.lognormals[,1:2]))
straight(mean(aplus(sa.lognormals[,1:2])),
         princomp(aplus(sa.lognormals[,1:2]))$Loadings[1,],
         col="red")
straight(mean(rplus(sa.lognormals[,1:2])),
         princomp(rplus(sa.lognormals[,1:2]))$loadings[,1],
         col="blue")

plot(rplus(sa.lognormals[,1:2]))
straight(mean(aplus(sa.lognormals[,1:2])),
         princomp(aplus(sa.lognormals[,1:2]))$Loadings[1,],
         col="red")
straight(mean(rplus(sa.lognormals[,1:2])),
         princomp(rplus(sa.lognormals[,1:2]))$loadings[,1],
         col="blue")
```

---

subsetting *Subsetting of compositions*

Description

Extract subsets (rows) or subsompositions (columns) of a compositional data set

Usage

```
  getStickyClassOption()
  setStickyClassOption(value)
## S3 method for class 'acomp'
x[i, j, drop=gsi.LengthOne(j)]
## S3 method for class 'rcomp'
x[i, j, drop=gsi.LengthOne(j)]
## S3 method for class 'aplus'
x[i, j, drop=gsi.LengthOne(j)]
## S3 method for class 'rplus'
x[i, j, drop=gsi.LengthOne(j)]
## S3 method for class 'ccomp'
x[i, j, drop=gsi.LengthOne(j)]
## S3 method for class 'rmult'
x[i, j, drop=gsi.LengthOne(j)]
## S3 method for class 'acomp'
x$name
## S3 method for class 'rcomp'
x$name
## S3 method for class 'aplus'
x$name
## S3 method for class 'rplus'
x$name
## S3 method for class 'ccomp'
x$name
## S3 method for class 'rmult'
x$name
```

Arguments

x	vector or dataset of a compositions class
i	row indices/names to select/exclude, resp. boolean of fitting length (recyling appplied if length(i)<nrow(x)); if x is a compositional vector, this gives the elements (equivalent to variables) to be extracted/selected
j	column indices/names to select/exclude, resp. boolean of fitting length (recyling appplied if length(i)<ncol(x))
drop	boolean, should matrices be simplified to vectors? defaults to FALSE (a difference with standard R). If set to TRUE, it has the extra effect of removing the compositional class
name	column name of the variable to be extracted OR name of a scaling function to be applied. It accepts unclass (or raw), clo, pwlr, alr, clr, ilr, apt, cpt, ipt, iit, ilt, cdt, idt and their inverses *Inv
value	logical, controlling the global options for sticky classes

Value

For [a vector or matrix with the relevant elements selected. When selecting rows, this object is of the same class than x, i.e. the class is sticky. When selecting columns, the class depends on

the number of columns selected and the value of drop. With drop=T, output is always a matrix or a vector. The same happens if gsi.LengthOne(j)==TRUE, which happens if and only if j is a non-null vector of length one (i.e. if you only want one single column).

If you want to get rid of sticky classes and return to the behaviour of "compositions" v1.xx, call setStickyClassOption(FALSE). This may be a good idea if you run old scripts written for that versions of "compositions". You can recover the default behaviour from "compositions" v2 with setStickyClassOption(TRUE), and check which sticky class status is currently defined in the global options with getStickyClassOption().

For $ the output is either a transformed data set of the appropriate class, or the selected column as a class-less vector. The transformation ability is particularly useful if you have put a whole compositional class into one column of a data set, in which case you can confortably use the transformations in formula interfaces (see example below). This is NEVER sticky.

Author(s)

R. Tolosana-Delgado

See Also

rmult, acomp, rcomp, aplus, rplus, ccomp,

Examples

```
data(Hydrochem)
xc = acomp(Hydrochem[,6:10])
xc[1:3,]
xc[-(10:nrow(xc)),]
xc[1:3,1:3]
xc[1:3,1:3, drop=TRUE]

xc[1:3,1]
class(xc[1:4,1])
class(xc[1:4,1, drop=TRUE])

data("Hydrochem")
xc = acomp(Hydrochem[, 9:14])
Hydrochem$compo = xc
lm(compo$clr~River, data=Hydrochem)
```

---

summary.acomp *Summarizing a compositional dataset in terms of ratios*

Description

Summaries in terms of compositions are quite different from classical ones. Instead of analysing each variable individually, we must analyse each pair-wise ratio in a log geometry.

Usage

```
          ## S3 method for class 'acomp'
summary( object, ... ,robust=getOption("robust"))
```

Arguments

object	a data matrix of compositions, not necessarily closed
...	not used, only here for generics
robust	A robustness description. See robustnessInCompositions for details. The parameter can be null for avoiding any estimation.

Details

It is quite difficult to summarize a composition in a consistent and interpretable way. We tried to provide such a summary here, based on the idea of the variation matrix.

Value

The result is an object of type "summary.acomp"

mean	the mean.acomp composition
mean.ratio	a matrix containing the geometric mean of the pairwise ratios
variation	the variation matrix of the dataset ({variation.acomp})
expsd	a matrix containing the one-sigma factor for each ratio, computed as exp(sqrt(variation.acomp(W))). To obtain a two-sigma-factor, one has to take its squared value (power 1.96, actually).
invexpsd	the inverse of the preceding one, giving the reverse bound. Additionally, it can be "almost" intepreted as a correlation coefficient, with values near one indicating high proportionality between the components.
min	a matrix containing the minimum of each of the pairwise ratios
q1	a matrix containing the 1-Quartile of each of the pairwise ratios
median	a matrix containing the median of each of the pairwise ratios
q1	a matrix containing the 3-Quartile of each of the pairwise ratios
max	a matrix containing the maximum of each of the pairwise ratios

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de, R. Tolosana-Delgado

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

See Also

acomp

Examples

```
data(SimulatedAmounts)
summary(acomp(sa.lognormals))
```

---

summary.aplus *Summaries of amounts*

Description

Summary of a vector of amounts, according to its underlying geometry.

Usage

```
          ## S3 method for class 'aplus'
summary( object, ...,
           digits=max(3, getOption("digits")-3), robust=NULL)
          ## S3 method for class 'rplus'
summary( object, ..., robust=NULL)
          ## S3 method for class 'rmult'
summary( object, ..., robust=NULL)
```

Arguments

object	an aplus/rplus set of amounts
digits	the number of significant digits to be used. The argument can also be used with rplus/rmult.
...	not used, only here for generics
robust	A robustness description. See robustnessInCompositions for details. The option is currently not supported. If support is added the default will change to getOption(robust).

Details

The obtained value is the same as for the classical summary summary, although in the case of aplus objects, the statistics have been computed in a logarithmic geometry, and exponentiated afterwards (which just changes the mean, equivalent to the geometric mean of the data set).

Value

A matrix containing summary statistics (minimum, the three quantiles, the mean and the maximum) of each component.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

aplus,rplus,summary.acomp, summary.rcomp

Examples

```
data(SimulatedAmounts)
summary(aplus(sa.lognormals))
summary(aplus(sa.tnormals))
summary(rplus(sa.lognormals))
summary(rplus(sa.tnormals))
summary(rmult(sa.lognormals))
```

---

summary.rcomp *Summary of compositions in real geometry*

Description

Compute a summary of a composition based on real geometry.

Usage

```
          ## S3 method for class 'rcomp'
summary( object, ... ,robust=NULL)
```

Arguments

object	an rcomp dataset of compositions
...	further arguments to summary
robust	A robustness description. See robustnessInCompositions for details. The option is currently not supported. If support is added the default will change to getOption(robust).

Details

The data is applied a clo operation before the computation. Note that the statistics obtained will not keep any consistency if computed with all the parts available or only with a subcomposition.

Value

A matrix containing summary statistics. The value is the same as for the classical summary summary applied to a closed dataset.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

rcomp, summary.aplus, summary.acomp

Examples

```
data(SimulatedAmounts)
summary(rcomp(sa.lognormals))
summary(rcomp(sa.tnormals))
```

---

sumprojector *Compute the global projector to the observed subspace.*

Description

Routines to compute the global projector to the observed subspace, down-weighting the subspaces with more missing values.

Usage

```
sumMissingProjector(x,...)
## S3 method for class 'acomp'
sumMissingProjector(x,has=is.NMV(x),...)
## S3 method for class 'aplus'
sumMissingProjector(x,has=is.NMV(x),...)
## S3 method for class 'rcomp'
sumMissingProjector(x,has=!(is.MAR(x)|is.MNAR(x)),...)
## S3 method for class 'rplus'
sumMissingProjector(x,has=!(is.MAR(x)|is.MNAR(x)),...)
## S3 method for class 'rmult'
sumMissingProjector(x,has=is.finite(x),...)
```

Arguments

x	a dataset of some type containing missings
has	the values to be regarded as non missing
...	further generic arguments that might be useful for other functions.

Details

The function missingProjector generates a list of N square matrices of dimension DxD (with N and D respectively equal to the number of rows and columns in x). Each of these matrices gives the projection of a data row onto its observed sub-space. Then, the function sumMissingProjector takes all these matrices and sums them in a efficient way, generating a "summary" of observed sub-spaces.

Value

The matrix of rotation/re-weighting of the original data set, down-weighting the subspaces with more missing values. This matrix is useful to obtain estimates of the mean (and variance, in the future) still unbiased in the presence of lost values (only of type MAR, stricly-speaking, but anyway useful for any type of missing value, when used with care). This matrix is the Fisher Information in the presence of missing values.

Missing Policy

No missing policy is given by the routine itself. Its treatment of missing values depends on the "has" argument.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de, Raimon Tolosana-Delgado

References

Boogaart, K.G. v.d., R. Tolosana-Delgado, M. Bren (2006) Concepts for handling of zeros and missing values in compositional data, in E. Pirard (ed.) (2006)Proccedings of the IAMG'2006 Annual Conference on "Quantitative Geology from multiple sources", September 2006, Liege, Belgium, S07-01, 4pages, http://stat.boogaart.de/Publications/iamg06_s07_01.pdf

See Also

missingProjector, clr,rcomp, aplus, princomp.acomp, plot.acomp, boxplot.acomp, barplot.acomp, mean.acomp, var.acomp, variation.acomp, cov.acomp, msd

Examples

```
data(SimulatedAmounts)
sumMissingProjector(acomp(sa.lognormals))
sumMissingProjector(acomp(sa.tnormals))
```

`Supervisor` *Proportions of supervisor's statements assigned to different categories*

Description

The results of a study of a single supervisor in his relationship to three supervisee are recorded. Instructions in a technical subject took place in sessions of one hour and with only one supervisee at the time. Each supervisee attended six sessions (once every two weeks in a twelve-week period). All of 18 sessions were recorded and for each session the 'statements' of the supervisor were classified into four categories. Thus for each session the proportion of statements in the four categories are set out in a two-way table according to the fortnight (6) and the supervisee (3).

Usage

```
data(Supervisor)
```

Format

A 18x13 matrix

Details

For each session the 'statements' of the supervisor were classified into four categories

C: commanding, posing a specific instruction to the supervisee,

D: demanding, posing a specific question to the supervisee,

E: exposing, providing the supervisee with an explanation,

F: faulting, pointing out faulty technique to the supervisee.

Thus for each session the proportion of statements in the four categories are set out in a two-way table according to the fortnight (6) and the supervisee (3). The C, D, E, F values in the rows sum mostly to 1, except for some rounding errors.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name SUPERVIS.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison, J. (1986): The Statistical Analysis of Compositional Data, (Data 7) pp12.

`ternaryAxis` *Axis for ternary diagrams*

Description

Displaying compositions in ternary diagrams

Usage

```
ternaryAxis(side=1:3,at=seq(0.2,0.8,by=0.2),
                      labels=if(is.list(at)) lapply(at,format) else format(at),
                        ...,
                        tick=TRUE,pos=0,
                        font.axis=par("font.axis"),
                        font.lab=par("font.lab"),
                        lty="solid",lwd=1,
                        len.tck=0.025,dist.lab=0.03,
                        dist.axis=0.03,
                        lty.tck="solid",
                        col.axis=par("col.axis"),
                        col.lab=par("col.lab"),
                        cex.axis=par("cex.axis"),
                        cex.lab=par("cex.lab"),
                        Xlab=NULL,Ylab=NULL,Zlab=NULL,small=TRUE,
                        xpd=NA,aspanel=FALSE)
```

Arguments

`side`	a vector giving the sides to draw the axis on. 1=under the plot, 2=the upper right axis, 3=the upper left axis. -1 is the portion axis of the first component, -2 is the portion axis of the second component, -3 is the portion axis of the third component. An empty vector or 0 suppresses axis plotting, but still plots the Xlab, Ylab and Zlab parameters.
`at`	a vector or a list of vectors giving the positions of the tickmarks.
`labels`	a vector giving the labels or a list of things that can serve as graphics annotations. Each element of the list is than sean as the labels for one of axes. IMPORTANT: if plotting formulae enclose the list of labels into a list.
`tick`	a logical whether to draw the tickmark lines
`pos`	the portion of the opposite component to draw the axis on. Proportion axss shrinks, when pos>0 !
`font.axis`	the font for the axis annotations
`font.lab`	the font for the variable labels
`lty`	the line type of the axis line. (see `par`). NA supresses plotting.
`lty.tck`	the line type of the tickmarks. NA suppresses plotting.

len.tck	the line length of the tickmarks.
dist.axis	the distance of the variable labels from the axes. Positve values point outward from the plot.
dist.lab	the distance of the axes labels from the axes. Positve values point outward from the plot.
lwd	the line widths of axis line and tickmarks. (see par)
col.axis	the color to plot the axis line, the tickmarks and the axes labels.
col.lab	the color to plot the variable labels.
cex.axis	The character size to plot the axes labels. (see par)
cex.lab	The character size for the variable labels
Xlab	the label for the lower left component.
Ylab	the label for the lower right component.
Zlab	the label for the upper component.
small	wether to plot the lower labels under the corners
xpd	Extended plotting region. See (see par).
aspanel	Is this called as a slave to acutally plot the axis (TRUE), or as a user level function to instatiate the axis (FALSE).
...	further graphical that might be of use for other functions, but are silently ignored here

Details

This function has two uses. If called with aspanel=TRUE it acutally draws the axes to a panel. In other cases it tries to modify the axes argument of the current plot to add the axis. I.e. it will force a replotting of the plot with the new axes settings. Thus an old axes is removed.

To ensure that various axes can be drawn with various parameters most of the arguments can take a vector or list of the same length as side providing the different parameters for each of the axes to be drawn.

There are two types of axes: Proportion axes (1:3) and portions axes (-1:-3). The best place to draw a Proportion axes is pos=0, which is the standard for axis in ternary diagrams. Portion axes are best drawn at pos=0.5 in the middle of the plot.

Author(s)

K.Gerald v.d. Boyogaart http://www.stat.boogaart.de, Raimon Tolosana-Delgado

See Also

plot.aplus, plot3D (for 3D plot), kingTetrahedron (for 3D-plot model export), qqnorm.acomp,boxplot.acomp

Examples

```
data(SimulatedAmounts)
plot(acomp(sa.lognormals),axes=TRUE)
ternaryAxis(side=1:3,pos=0,col.axis="red",col.lab="green")
ternaryAxis(side=1:3,at=1:9/10,
            labels=expression(9:1,4:1,7:3,3:2,1:1,2:3,3:7,1:4,1:9),
            pos=0,col.axis="red",col.lab="green")
ternaryAxis(side=rep(-1:-3,3),labels=paste(seq(20,80,by=20),"%"),
            pos=rep(c(0,0.5,1),each=3),col.axis=1:3,col.lab="green")
ternaryAxis(side=rep(1:3,3),at=1:9/10,
            labels=expression(9:1,4:1,7:3,3:2,1:1,2:3,3:7,1:4,1:9),
            pos=rep(c(0,0.5,1),each=3))

plot(acomp(sa.lognormals5),axes=TRUE)
ternaryAxis(side=1:3,pos=0,col.axis="red",col.lab="green")
ternaryAxis(side=1:3,at=1:9/10,
            labels=expression(9:1,4:1,7:3,3:2,1:1,2:3,3:7,1:4,1:9),
            pos=0,col.axis="red",col.lab="green")
```

---

totals *Total sum of amounts*

Description

Calculates the total amount by summing the individual parts.

Usage

```
totals(x,...)
## S3 method for class 'acomp'
totals(x,...,missing.ok=TRUE)
## S3 method for class 'rcomp'
totals(x,...,missing.ok=TRUE)
## S3 method for class 'aplus'
totals(x,...,missing.ok=TRUE)
## S3 method for class 'rplus'
totals(x,...,missing.ok=TRUE)
## S3 method for class 'ccomp'
totals(x,...,missing.ok=TRUE)
```

Arguments

x	an amount/amount dataset
...	not used, only here for generic purposes
missing.ok	if TRUE ignores missings; if FALSE issues an error if the total cannot be calculated due to missings.

Value

a numeric vector of length equal to `ncol(x)` containing the total amounts

Missing Policy

if `missing.ok=TRUE` missings are just regarded as 0, if `missing.ok=FALSE` WZERO values is still regarded as 0 and other sorts lead to `NA` in the respective totals.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

`aplus`

Examples

```
data(SimulatedAmounts)
totals(acomp(sa.lognormals))
totals(rcomp(sa.lognormals,total=100))
totals(aplus(sa.lognormals))
totals(rplus(sa.lognormals))
aplus(acomp(sa.lognormals),total=totals(aplus(sa.lognormals)))
```

`transformations from 'mixtures' to 'compositions' classes`

Transformations from 'mixtures' to 'compositions' classes

Description

Transformations from 'mixtures' of the "mixR" library to 'compositions' classes 'aplus', 'acomp', 'rcomp', 'rplus' and 'rmult'.

Usage

```
mix.2aplus(X)
mix.2acomp(X)
mix.2rcomp(X)
mix.2rplus(X)
mix.2rmult(X)
```

Arguments

`X`	mixture object to be converted

Details

A 'compositions' object is obtained from the mixtute object m, having the same data matrix as mixture object m i.e. `m$mat`.

Value

A 'compositions' object of the class 'aplus', 'acomp', 'rcomp', 'rplus' or 'rmult'.

See Also

`aplus` `acomp` `rcomp` `rplus` `rmult`

Examples

```
## Not run:
m <- mix.Read("Glac.dat")      # reads the Glacial data set from Aitchison (1986)
m <- mix.Extract(m,c(1,2,3,4))   # mix object with closed four parts subcomposition
ap <- mix.2aplus(m)   # ap is a 'compositions' object of the aplus class
ac <- mix.2acomp(m)   # ac is a 'compositions' object of the acomp class

## End(Not run)
```

---

tryDebugger *Empirical variograms for compositions*

Description

An R-debugger that also works with errors in parameters.

Usage

```
tryDebugger(dump = last.dump)
```

Arguments

dump	An R dump object created by 'dump.frames'.

Details

Works like debugger, with the small exception that it also works in situations of nasty errors, like recursive parameter evaluation, missing parameters, and additional errors in arguments.

Value

Nothing.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

debugger

Examples

```
## Not run:
f <- function(x,y=y) {y}
f(1)
tryDebugger() # works
debugger() # Does not allow to browse anything

## End(Not run)
```

---

ult *Uncentered log transform*

Description

Compute the uncentered log ratio transform of a (dataset of) composition(s) and its inverse.

Usage

```
ult( x ,...)
ultInv( z ,..., orig=gsi.orig(z))
Kappa( x ,...)
```

Arguments

x	a composition or a data matrix of compositions, not necessarily closed
z	the ult-transform of a composition or clr-transforms of compositions (or a data matrix), not necessarily centered
...	for generic use only
orig	a compositional object which should be mimicked by the inverse transformation. It is the generic argument. Typically the orig argument is stored as an attribute in z and will be extracted automatically by this function; if this fails, orig can be set equal to the dataset that was transformed in the first place.

Details

The ult-transform is simply the elementwise log of the closed composition. The ult has some important properties in the scope of Information Theory of probability vectors (but might be mostly misleading for exploratory analysis of compositions). DO NOT USE if you do not know what you are doing.

Value

`ult` gives the uncentered log transform,
`ultInv` gives closed compositions with the given ult/clr-transforms
`Kappa` gives the difference between the clr and the ult transforms. It is quite linked to information measures.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

`ilr`,`alr`,`apt`

Examples

```
(tmp <- ult(c(1,2,3)))
ultInv(tmp)
ultInv(tmp) - clo(c(1,2,3)) # 0
data(Hydrochem)
cdata <- Hydrochem[,6:19]
pairs(ult(cdata),pch=".")
Kappa(c(1,2,3))
```

---

var.acomp *Variances and covariances of amounts and compositions*

Description

Compute the (co)variance matrix in the several approaches of compositional and amount data analysis.

Usage

```
  var(x,...)
    ## Default S3 method:
var(x, y=NULL, na.rm=FALSE, use, ...)
    ## S3 method for class 'acomp'
var(x,y=NULL,...,robust=getOption("robust"),
       use="all.obs",giveCenter=FALSE)
    ## S3 method for class 'rcomp'
var(x,y=NULL,...,robust=getOption("robust"),
       use="all.obs",giveCenter=FALSE)
    ## S3 method for class 'aplus'
var(x,y=NULL,...,robust=getOption("robust"),
       use="all.obs",giveCenter=FALSE)
    ## S3 method for class 'rplus'
var(x,y=NULL,...,robust=getOption("robust"),
```

```
       use="all.obs",giveCenter=FALSE)
    ## S3 method for class 'rmult'
var(x,y=NULL,...,robust=getOption("robust"),
       use="all.obs",giveCenter=FALSE)
  cov(x,y=x,...)
    ## Default S3 method:
cov(x, y=NULL, use="everything",
       method=c("pearson", "kendall", "spearman"), ...)
    ## S3 method for class 'acomp'
cov(x,y=NULL,...,robust=getOption("robust"),
       use="all.obs",giveCenter=FALSE)
    ## S3 method for class 'rcomp'
cov(x,y=NULL,...,robust=getOption("robust"),
       use="all.obs",giveCenter=FALSE)
    ## S3 method for class 'aplus'
cov(x,y=NULL,...,robust=getOption("robust"),
       use="all.obs",giveCenter=FALSE)
    ## S3 method for class 'rplus'
cov(x,y=NULL,...,robust=getOption("robust"),
       use="all.obs",giveCenter=FALSE)
    ## S3 method for class 'rmult'
cov(x,y=NULL,...,robust=getOption("robust"),
    use="all.obs",giveCenter=FALSE)
```

Arguments

x	a dataset, eventually of amounts or compositions
y	a second dataset, eventually of amounts or compositions
na.rm	see stats::var
use	see stats::var
method	see stats::cov
...	further arguments to stats::var e.g. use
robust	A description of a robust estimator. FALSE for the classical estimators. See robustnessInCompositions for further details.
giveCenter	If TRUE the center used in the variance calculation is reported as a "center" attribute. This is especially necessary for robust estimations, where a reasonable center can not be computed independently for the me variance calculation.

Details

The basic functions of stats::var and stats::cov are turned to S3-generics. The original versions are copied to the default method. This allows us to introduce generic methods to handle variances and covariances of other data types, such as amounts or compositions.
If classed amounts or compositions are involved, they are transformed with their corresponding transforms, using the centered default transform (cdt). That implies that the variances have to be interpreded in a log scale level for acomp and aplus.

We should be aware that variance matrices of compositions (acomp and rcomp) are singular. They can be transformed to the correponding nonsingular variances of ilr or ipt-space by clrvar2ilr.

In R versions older than v2.0.0,

stats::var and stats::cov were defined in package "base" instead of in "stats". This might produce some misfunction.

Value

The variance matrix of x or the covariance matrix of x and y.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

cdt, clrvar2ilr, clo, mean.acomp, acomp, rcomp, aplus, rplus, variation

Examples

```
data(SimulatedAmounts)
meanCol(sa.lognormals)
var(acomp(sa.lognormals))
var(rcomp(sa.lognormals))
var(aplus(sa.lognormals))
var(rplus(sa.lognormals))
cov(acomp(sa.lognormals5[,1:3]),acomp(sa.lognormals5[,4:5]))
cov(rcomp(sa.lognormals5[,1:3]),rcomp(sa.lognormals5[,4:5]))
cov(aplus(sa.lognormals5[,1:3]),aplus(sa.lognormals5[,4:5]))
cov(rplus(sa.lognormals5[,1:3]),rplus(sa.lognormals5[,4:5]))
cov(acomp(sa.lognormals5[,1:3]),aplus(sa.lognormals5[,4:5]))

svd(var(acomp(sa.lognormals)))
```

---

variation *Variation matrices of amounts and compositions*

Description

Compute the variation matrix in the various approaches of compositional and amount data analysis. Pay attention that this is not computing the variance or covariance matrix!

Usage

```
     variation(x,...)
          ## S3 method for class 'acomp'
variation(x, ...,robust=getOption("robust"))
          ## S3 method for class 'rcomp'
variation(x, ...,robust=getOption("robust"))
          ## S3 method for class 'aplus'
variation(x, ...,robust=getOption("robust"))
          ## S3 method for class 'rplus'
variation(x, ...,robust=getOption("robust"))
          ## S3 method for class 'rmult'
variation(x, ...,robust=getOption("robust"))
          is.variation(M, tol=1e-10)
```

Arguments

x	a dataset, eventually of amounts or compositions
...	currently unused
robust	A description of a robust estimator. FALSE for the classical estimators. See robustnessInCompositions for further details.
M	a matrix, to check if it is a valid variation
tol	tolerance for the check

Details

The variation matrix was defined in the acomp context of analysis of compositions as the matrix of variances of all possible log-ratios among components (Aitchison, 1986). The generalization to rcomp objects is simply to reproduce the variance of all possible differences between components. The amount (aplus, rplus) and rmult objects should not be treated with variation matrices, because this was intended to skip the existence of a closure (which does not exist in the case of amounts).

Value

The variation matrix of x.

For is.variation, a boolean saying if the matrix satisfies the conditions to be a variation matrix.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

cdt, clrvar2ilr, clo, mean.acomp, acomp, rcomp, aplus, rplus

Examples

```
data(SimulatedAmounts)
meanCol(sa.lognormals)
variation(acomp(sa.lognormals))
variation(rcomp(sa.lognormals))
variation(aplus(sa.lognormals))
variation(rplus(sa.lognormals))
variation(rmult(sa.lognormals))
```

---

variograms *Variogram functions*

Description

Valid scalar variogram model functions.

Usage

```
vgram.sph( h , nugget = 0, sill = 1, range= 1,... )
vgram.exp( h , nugget = 0, sill = 1, range= 1,... )
vgram.gauss( h , nugget = 0, sill = 1, range= 1,... )
vgram.cardsin( h , nugget = 0, sill = 1, range= 1,... )
vgram.lin( h , nugget = 0, sill = 1, range= 1,... )
vgram.pow( h , nugget = 0, sill = 1, range= 1,... )
vgram.nugget( h , nugget = 1,...,tol=1E-8 )
```

Arguments

h	a vector providing distances, a matrix of distance vectors in its rows or a data.frame of distance vectors.
nugget	The size of the nugget effect (i.e. the limit to 0). At zero itself the value is always 0.
sill	The sill (i.e. the limit to infinity)
range	The range parameter. I.e. the distance in which sill is reached or if this does not exist, where the value is in some sense nearly the sill.
...	not used
tol	The distance that is considered as nonzero.

Details

The univariate variograms are used in the CompLinCoReg as building blocks of multivariate variogram models.

sph Spherical variogram

exp Exponential variogram

gauss The Gaussian variogram.

gauss The cardinal sine variogram.

lin Linear Variogram. Increases over the sill, which is reached at `range`.

pow The power variogram. Increases over the sill, which is reached at `range`.

nugget The pure nugget effect variogram.

Value

A vector of size NROW(h), giving the variogram values.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

References

Cressie, N.C. (1993) Spatial statistics

Tolosana, van den Boogaart, Pawlowsky-Glahn (2009) Estimating and modeling variograms of compositional data with occasional missing variables in R, StatGis09

See Also

vgram2lrvgram, CompLinModCoReg, vgmFit

Examples

```
## Not run:
data(juraset)
X <- with(juraset,cbind(X,Y))
comp <- acomp(juraset,c("Cd","Cu","Pb","Co","Cr"))
lrv <- logratioVariogram(comp,X,maxdist=1,nbins=10)
plot(lrv)

## End(Not run)
```

---

varmlm *Residual variance of a model*

Description

Computes the unbiased estimate for the variance of the residuals of a model.

Usage

```
## S3 method for class 'mlm'
var(x,...)
## S3 method for class 'lm'
var(x,...)
```

Arguments

x	a linear model object
...	Unused, for generic purposes only.

Details

The difference of this command to `var(resid(X))` is that this command correctly adjusts for the degrees of freedom of the model.

Value

var.lm	returns a scalar giving the estimated variance of the residuals
var.mlm	returns a the estimated variance covariance matrix of the residuals

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de, Raimon Tolosana-Delgado

See Also

vcov

Examples

```
data(Orange)
var(lm(circumference~age,data=Orange))
var(lm(cbind(circumference,age)~age,data=Orange))
```

---

vcovAcomp *Variance covariance matrix of parameters in compositional regression*

Description

The variance covariance tensor structured according of linear models with ilr(acomp(...)) responses.

Usage

```
vcovAcomp(object,...)
```

Arguments

object	a statistical model
...	further optional parameters for vcov

Details

The prediction error in compositional linear regression models is a complicated object. The function should help to organize it.

Value

An array with 4 dimensions. The first 2 are the index dimensions of the ilr transform. The later 2 are the index of the parameter.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

vcov

Examples

```
data(SimulatedAmounts)
model <- lm(ilr(sa.groups)~sa.groups.area)
vcovAcomp(model)[,,1,1]
```

---

vgmFit *Compositional variogram model fitting*

Description

Fits a parametric variogram model to an empirical logratio-Variogram

Usage

```
vgmFit2lrv(emp,vg,...,mode="log",psgn=rep(-1,length(param)),print.level=1)
## S3 method for class 'logratioVariogram'
fit.lmc(v,model,...,mode="log",psgn=rep(-1,length(param)),print.level=1)
vgmFit(emp,vg,...,mode="log",psgn=rep(-1,length(param)),print.level=1)
vgmGof(p = vgmGetParameters(vg), emp, vg, mode = "log")
vgmGetParameters(vg,envir=environment(vg))
vgmSetParameters(vg,p)
fit.lmc(v,...)
```

Arguments

emp	An empirical logratio-Variogram as e.g. returned by logratioVariogram
v	An empirical logratio-Variogram as e.g. returned by logratioVariogram
vg	A compositional clr-variogram (or ilt-vagriogram) model function.
model	A compositional clr-variogram (or ilt-vagriogram) model function, output of a call to .
...	further parameters to nlm
mode	either "ls" or "log" for selection of either using either least squares or least squares on logarithmic values.
psgn	Contains a parameter code for each of the parameters. -1 means the parameter should be used as is. 0 means the parameter is nonnegativ and 1 means the parameter is striktly positiv. This allows to provide parameter limits if the fitting procedure fails.
print.level	The print.level of nlm. 0 for no printing. 1 for a rough information about the sucess and 2 for step by step printing.
p	Is the parameter of the variogram model in linearized form as e.g. returned by vgmGetParameters.
envir	The environment the default parameters of the model should be evaluated in.

Details

The function is mainly a wrapper to nlm specifying the an objective function for modell fitting, taking the starting values of fitting procedure from the default arguments and writing the results back. Variogram model fitting is more an art than a straight forward procedure. Fitting procedures typically only find a right optimum if reasonable starting parameters are provided. The fit should be visually checked afterwards.
The meaning of psgn is subject to change. We will probably provide a more automatic procedure later.
vgmFit is a copy of vgmFit2lrv, but deprecated. The name will later be used for other functionality.

Value

vgmFit2lrv returns a list of two elements.

nlm	The result of nlm containing covergence codes.
vg	A version of vg but with default parameters modified according to the fitting.

vgmGof returns a scalar quantifiying the goodness of fit, of a model and an empirical variogram.
vgmGetParameters extracts the default values of a variogram model function to a parameter vector. It returns a numeric vector.
vgmSetParameters does the inverse operation and modifies the default according to the new values in p. It returns vg with modifiend default parameter values.

Author(s)

K.Gerald v.d. Boogaart http://www.stat.boogaart.de

See Also

vgram2lrvgram, CompLinModCoReg, logratioVariogram

Examples

```
## Not run:
data(juraset)
X <- with(juraset,cbind(X,Y))
comp <- acomp(juraset,c("Cd","Cu","Pb","Co","Cr"))
lrv <- logratioVariogram(comp,X,maxdist=1,nbins=10)
fff <- CompLinModCoReg(~nugget()+sph(0.5)+R1*exp(0.7),comp)
fit <- vgmFit(lrv,fff)
fit
fff(1:3)
plot(lrv,lrvg=vgram2lrvgram(fit$vg))

## End(Not run)
```

WhiteCells *White-cell composition of 30 blood samples by two different methods*

Description

In 30 blood samples portions of three kinds of white cells

G: granulocytes,

L: lymphocytes,

M: monocytes,

were determined with two methods, time-consuming microscopic and automatic image analysis. The resulting 30 pairs of 3-part compositions are recorded.

Usage

```
data(WhiteCells)
```

Format

A 30x6 matrix

Details

In an experiment each of 30 blood samples was halved, one half being assigned randomly to one method, the other half to the other method. We have 60 cases of 3-part compositions but these are essentially 30 pairs of related compositions. All 3-part portions sums to one, except for some rounding errors.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name WCELLS.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison, J. (1986) The Statistical Analysis of Compositional Data, 1986 (Data 9) pp16.

---

wrapped_functions *Standard R functions wrapped for compatibility*

Description

These functions of a standard R distribution (from package "base" or "stats") are wrapped by naive functions in "compositions" with the goal to ensure their normal behavior with compositional data objects.

Usage

```
anova(...)
```

Arguments

...	arguments passed to the original function (all!)

Details

The functions documented in this page are just wrapers around base functions from R that, due to a variety of reasons, need pre- or post-processing when "compositions" is loaded. Pre-processing are e.g., converting "rmult" class objects to plain "matrix" objects, or removing sticky class behaviour (see `getStickyClassOption`)

Value

The same as the original function from package base (i.e. search for it with ‘?base::anova‘).

Author(s)

Raimon Tolosana-Delgado http://www.stat.boogaart.de

See Also

anova in package "base" .

Examples

```
# anova:
data("Hydrochem") # load data
Z = acomp(Hydrochem[,7:19]) # select composition
Hydrochem$compo = Z # attach to dataset
md = lm(alr(compo)~log(H), data=Hydrochem) # fit model
anova(md)   # anova test
```

Yatquat *Yatquat fruit evaluation*

Description

The quality of yatquat tree fruit is assessed in terms of relative proportions by volume of flesh, skin and stone. In an experiment an arboriculturist uses 40 trees, randomly allocated 20 to the hormone treatment and leaves untreated the remaining 20 trees. Data provides fruit compositions of the present season and the preceding season, as well as the treatment: 1 for the treated trees, -1 for untreated trees.

Usage

```
data(Yatquat)
```

Format

A 40x7 data matrix

Details

The yatquat tree produces each season a single large fruit. Data provides fruit compositions of the present season, the compositions of the fruit of the same 40 trees for the preceding season when none of the trees were treated, and in addition the Type: 1 for the treated trees, -1 for untreated trees. For each of the 40 cases we have two 3-part composition on flesh, skin and stone. The column names are:

prFL	portion of fruit flesh in the present season,
prSK	portion of fruit skin in the present season,
prST	portion of fruit stone in the present season,
Type	1 for treated, -1 for untreated trees,
paFL	portion of fruit flesh in the preceding season,
paSK	portion of fruit skin in the preceding season,
paST	portion of fruit stone in the preceding season,

All 3-part compositions sum to one.

Note

Courtesy of J. Aitchison

Source

Aitchison: CODA microcomputer statistical package, 1986, the file name YATQUAT.DAT, here included under the GNU Public Library Licence Version 2 or newer.

References

Aitchison, J. (1986) The Statistical Analysis of Compositional Data, (Data 12) pp17.

Examples

```
#data(Yatquat)
#plot(acomp(Yatquat[,1:3]),col=as.numeric(Yatquat[,4])+2)
#plot(acomp(Yatquat[,5:7]),col=as.numeric(Yatquat[,4])+2)
```

`zeroreplace` *Zero-replacement routine*

Description

A function to automatically replace rounded zeroes/BDLs in a composition.

Usage

```
zeroreplace(x,d=NULL,a=2/3)
```

Arguments

`x`	composition or dataset of compositions
`d`	vector containing the detection limits of each part
`a`	fraction of the detection limit to be used in replacement

Details

If `d` is given, zeroes from each column of `x` are replaced by the corresponding detection limit contained there, scaled down by the value of `a` (usually a scalar, although if it is a vector it will be recycled with a warning). The variable `d` should be a vector of length equal to `ncol(x)` or a matrix of the same shape as `x`.
If `d=NULL`, then the detection limit is extracted from the data set, if it is available there (i.e., if there are negative numbers). If no negative number is present in the data set, and no value is given for `d`, the result will be equal to `x`. See `compositions.missings` for more details on the missing policy.

Value

an object of the same class as x, where all WZERO values have been replaced. Output contains a further attribute (named Losts), with a logical array of the same dimensions as x, showing which elements were replaced (TRUE) and which were kept unchanged (FALSE).

References

Aitchison, J. (1986) *The Statistical Analysis of Compositional Data* Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK). 416p.

Mart\'in-Fern\'andez, J.A.; Barcel\'o-Vidal, C. and Pawlowsky-Glahn, V. (2003) Dealing With Zeros and Missing Values in Compositional Data Sets Using Nonparametric Imputation. *Mathematical Geology*, 35 , 253-278

https://ima.udg.edu/Activitats/CoDaWork03/

https://ima.udg.edu/Activitats/CoDaWork05/

See Also

compositions.missings,getDetectionlimit

Examples

```
data(SimulatedAmounts)
x <- acomp(sa.lognormals)
xnew <- simulateMissings(x,dl=0.05,knownlimit=FALSE)
xnew
xrep <- zeroreplace(xnew,0.05)
xrep
```

Index