Package ‘provenance’

June 6, 2019

Title  Statistical Toolbox for Sedimentary Provenance Analysis
Version  2.3
Date  2019-06-05

Description  Bundles a number of established statistical methods to facilitate the visual interpretation of large datasets in sedimentary geology. Includes functionality for adaptive kernel density estimation, principal component analysis, correspondence analysis, multidimensional scaling, generalised procrustes analysis and individual differences scaling using a variety of dissimilarity measures. Univariate provenance proxies, such as single-grain ages or (isotopic) compositions are compared with the Kolmogorov-Smirnov, Kuiper or Sircombe-Hazelton L2 distances. Categorical provenance proxies such as chemical compositions are compared with the Aitchison and Bray-Curtis distances, and point-counting data with the chi-square distance. Also included are tools to plot compositional and point-counting data on ternary diagrams and point-counting data on radial plots, to calculate the sample size required for specified levels of statistical precision, and to assess the effects of hydraulic sorting on detrital compositions. Includes an intuitive query-based user interface for users who are not proficient in R.

Author  Pieter Vermeesch [aut, cre]
Maintainer  Pieter Vermeesch <p.vermeesch@ucl.ac.uk>
Depends  R (>= 3.0.0)
Imports  MASS, methods, IsoplotR
URL  http://provenance.london-geochron.com
License  GPL-2
LazyData  true
RoxygenNote  6.1.1
Encoding  UTF-8
NeedsCompilation  no
Repository  CRAN
Date/Publication  2019-06-06 18:50:04 UTC
### R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALR</td>
<td>3</td>
</tr>
<tr>
<td>amalgamate</td>
<td>4</td>
</tr>
<tr>
<td>as.acomp</td>
<td>5</td>
</tr>
<tr>
<td>as.compositional</td>
<td>5</td>
</tr>
<tr>
<td>as.counts</td>
<td>6</td>
</tr>
<tr>
<td>as.data.frame</td>
<td>7</td>
</tr>
<tr>
<td>botev</td>
<td>7</td>
</tr>
<tr>
<td>bray.diss</td>
<td>8</td>
</tr>
<tr>
<td>CA</td>
<td>9</td>
</tr>
<tr>
<td>central</td>
<td>9</td>
</tr>
<tr>
<td>CLR</td>
<td>10</td>
</tr>
<tr>
<td>combine</td>
<td>11</td>
</tr>
<tr>
<td>densities</td>
<td>12</td>
</tr>
<tr>
<td>diss</td>
<td>13</td>
</tr>
<tr>
<td>endmembers</td>
<td>13</td>
</tr>
<tr>
<td>get.f</td>
<td>14</td>
</tr>
<tr>
<td>get.n</td>
<td>15</td>
</tr>
<tr>
<td>get.p</td>
<td>16</td>
</tr>
<tr>
<td>GPA</td>
<td>16</td>
</tr>
<tr>
<td>indscal</td>
<td>17</td>
</tr>
<tr>
<td>KDE</td>
<td>18</td>
</tr>
<tr>
<td>KDEs</td>
<td>19</td>
</tr>
<tr>
<td>KS.diss</td>
<td>20</td>
</tr>
<tr>
<td>Kuiper.diss</td>
<td>21</td>
</tr>
<tr>
<td>lines.ternary</td>
<td>21</td>
</tr>
<tr>
<td>MDS</td>
<td>22</td>
</tr>
<tr>
<td>minsorting</td>
<td>23</td>
</tr>
<tr>
<td>Namib</td>
<td>24</td>
</tr>
<tr>
<td>PCA</td>
<td>25</td>
</tr>
<tr>
<td>plot.CA</td>
<td>25</td>
</tr>
<tr>
<td>plot.compositional</td>
<td>26</td>
</tr>
<tr>
<td>plot.distributional</td>
<td>27</td>
</tr>
<tr>
<td>plot.GPA</td>
<td>28</td>
</tr>
<tr>
<td>plot.INDSCAL</td>
<td>29</td>
</tr>
<tr>
<td>plot.KDE</td>
<td>30</td>
</tr>
<tr>
<td>plot.KDEs</td>
<td>30</td>
</tr>
<tr>
<td>plot.MDS</td>
<td>31</td>
</tr>
<tr>
<td>plot.minsorting</td>
<td>32</td>
</tr>
<tr>
<td>plot.PCA</td>
<td>33</td>
</tr>
<tr>
<td>plot.ternary</td>
<td>33</td>
</tr>
<tr>
<td>points.ternary</td>
<td>34</td>
</tr>
<tr>
<td>procrustes</td>
<td>35</td>
</tr>
<tr>
<td>provenance</td>
<td>36</td>
</tr>
<tr>
<td>radialplot</td>
<td>37</td>
</tr>
<tr>
<td>read.compositional</td>
<td>38</td>
</tr>
<tr>
<td>read.counts</td>
<td>39</td>
</tr>
</tbody>
</table>
Additive logratio transformation

Description

Calculates Aitchison’s additive logratio transformation for a dataset of class compositional or a compositional data matrix.

Usage

ALR(x, ...)  

## Default S3 method:
ALR(x, inverse = FALSE, ...)

## S3 method for class 'compositional'
ALR(x, ...)

Arguments

- **x**: an object of class compositional OR a matrix of numerical values
- **...**: optional arguments
- **inverse**: perform the inverse inverse logratio transformation?

Value

- a matrix of ALR coordinates OR an object of class compositional (if inverse=TRUE).

Examples

# logratio plot of trace element concentrations:
data(Namib)
alr <- ALR(Namib$Trace)
pairs(alr[,1:5])
title('log(X/Pb)')
amalgamate

Group components of a composition

Description

Adds several components of a composition together into a single component

Usage

amalgamate(X, ...)

## Default S3 method:
amalgamate(X, ...)

## S3 method for class 'compositional'
amalgamate(X, ...)

## S3 method for class 'counts'
amalgamate(X, ...)

## S3 method for class 'SRDcorrected'
amalgamate(X, ...)

Arguments

X a compositional dataset

... a series of new labels assigned to strings or vectors of strings denoting the components that need amalgamating

Value

an object of the same class as X with fewer components

Examples

data(Namib)
HMcomponents <- c("zr","tm","rt","TiOx","sph","ap","ep",
                  "gt","st","amp","cpx","opx")
am <- amalgamate(Namib$PTHM,feldspars="KF","P"),
      lithics=c("Lm","Lv","Ls"),heavies=HMcomponents)
plot(ternary(am))
as.acomp

create an acomp object

Description
Convert an object of class compositional to an object of class acomp for use in the compositions package.

Usage
as.acomp(x)

Arguments
x an object of class compositional

Value
a data.frame

Examples
data(Namib)
qfl <- ternary(Namib$PT,c('Q'),c('KF','P'),c('Lm','Lv','Ls'))
plot(qfl,type="QFL.dickinson")
qfl.acomp <- as.acomp(qfl)
## uncomment the next two lines to plot an error
## ellipse using the compositions package:
# library(compositions)
# ellipses(mean(qfl.acomp),var(qfl.acomp),r=2)

as.compositional
create a compositional object

Description
Convert an object of class matrix, data.frame or acomp to an object of class compositional.

Usage
as.compositional(x, method = NULL, colmap = "rainbow")

Arguments
x an object of class matrix, data.frame or acomp
method dissimilarity measure, either 'aitchison' for Aitchison’s CLR-distance or 'bray' for the Bray-Curtis distance.
colmap the colour map to be used in pie charts.
Value

an object of class compositional

Examples

data(Namib)
PT.acomp <- as.acomp(Namib$PT)
PT.compositional <- as.compositional(PT.acomp)
print(Namib$PT$x - PT.compositional$x)
## uncomment the following lines for an illustration of using this
## function to integrate the \code{provenance} package with \code{compositions}
# library(compositions)
# data(Glaciar)
# a.glac <- acomp(Glaciar)
# c.glac <- as.compositional(a.glac)
# summaryplot(c.glac,ncol=8)

as.counts create a counts object

Description

Convert an object of class matrix or data.frame to an object of class counts

Usage

as.counts(x, method = "chisq", colmap = "rainbow")

Arguments

x an object of class matrix or data.frame
method either "chisq" (for the chi-square distance) or "bray" (for the Bray-Curtis distance)
colmap the colour map to be used in pie charts.

Value

an object of class counts

Examples

X <- matrix(c(0,100,0,30,11,2,94,36,0),nrow=3,ncol=3)
rownames(X) <- 1:3
colnames(X) <- c('a','b','c')
comp <- as.counts(X)
d <- diss(comp)
as.data.frame

create a data.frame object

Description

Convert an object of class compositional to a data.frame for use in the robCompositions package

Usage

## S3 method for class 'compositional'
as.data.frame(x, ...)

## S3 method for class 'counts'
as.data.frame(x, ...)

Arguments

x an object of class compositional
...
optional arguments to be passed on to the generic function

Value

a data.frame

Examples

data(Namib)
Major.frame <- as.data.frame(Namib$Major)
## uncomment the next two lines to plot an error
## ellipse using the robCompositions package:
# library(robCompositions)
# plot(pcaCoDa(Major.frame))

botev

Compute the optimal kernel bandwidth

Description

Uses the diffusion algorithm of Zdravko Botev (2011) to calculate the bandwidth for kernel density estimation

Usage

botev(x)
Arguments

x  a vector of ordinal data

Value

a scalar value with the optimal bandwidth

Author(s)

Dzdravko Botev

References


Examples

fname <- system.file("DZ.csv",package="provenance")
bw <- botev(read.distributional(fname)$x$N1)
print(bw)

bray.diss  Bray-Curtis dissimilarity

Description

Calculates the Bray-Curtis dissimilarity between two samples

Usage

bray.diss(x, y)

Arguments

x  a vector containing the first compositional sample

y  a vector of length(x) containing the second compositional sample

Value

a scalar value

Examples

data(Namib)
print(bray.diss(Namib$HMS$["N1",],Namib$HMS$["N2",]))
**CA**  
*Correspondence Analysis*

**Description**
Performs Correspondence Analysis of point-counting data

**Usage**
```
CA(x, nf = 2, ...)  
```

**Arguments**
- `x`: an object of class `counts`
- `nf`: number of correspondence factors (dimensions)
- `...`: optional arguments to the `corresp` function of the MASS package

**Value**
an object of classes `CA`, which is synonymous to the MASS packages’ `correspondence` class.

**Examples**
```
data(Namib)
plot(CA(Namib$PT))
```

---

**central**  
*Calculate central compositions*

**Description**
Computes the geometric mean composition of a continuous mixture of point-counting data.

**Usage**
```
central(x, ...)  
```

**Arguments**
- `x`: an object of class `counts`
- `...`: optional arguments

**Details**
The central composition assumes that the observed point-counting distribution is the combination of two sources of scatter: counting uncertainty and true geological dispersion.
Value

an \([5 \times n]\) matrix with \(n\) being the number of categories and the rows containing:

- **theta** the ‘central’ composition.
- **err** the standard error for the central composition.
- **sigma** the overdispersion parameter, i.e. the coefficient of variation of the underlying logistic normal distribution. central computes a continuous mixture model for each component (column) separately. Covariance terms are not reported.
- **LL** the lower limit of a ‘1 sigma’ region for codetheta.
- **UL** the upper limit of a ‘1 sigma’ region for codetheta.
- **mswd** the mean square of the weighted deviates, a.k.a. reduced chi-square statistic.
- **p.value** the p-value for age homogeneity

---

CLR

Centred logratio transformation

---

Description

Calculates Aitchison’s centered logratio transformation for a dataset of class compositional or a compositional data matrix.

Usage

CLR(x, ...)

## Default S3 method:
CLR(x, inverse = FALSE, ...)

## S3 method for class 'compositional'
CLR(x, ...)

Arguments

- **x** an object of class compositional OR a matrix of numerical values
- **...** optional arguments
- **inverse** perform the inverse inverse logratio transformation?

Value

a matrix of CLR coordinates OR an object of class compositional (if inverse=TRUE)
Examples

# The following code shows that applying provenance's PCA function
# to compositional data is equivalent to applying R's built-in
# princomp function to the CLR transformed data.
data(Namib)
plot(PCA(Namib$Major))
device()
clrdat <- CLR(Namib$Major)
biplot(princomp(clrdat))

Value

a distributional data object with fewer samples than X

Usage

combine(X, ...)

Arguments

X a distributional dataset

... a series of new labels assigned to strings or vectors of strings denoting the samples that need amalgamating

Description

Lumps all single grain analyses of several samples together under a new name
A list of rock and mineral densities

Author(s)
Alberto Resentini and Pieter Vermeesch

References


See Also
restore, minsorting

Examples

data(Namib,densities)
N8 <- subset(Namib$HM,select="N8")
distribution <- minsorting(N8,densities,phi=2,sigmaphi=1,medium="air",by=0.05)
plot(distribution)
Calculate the dissimilarity matrix between two distributional or compositional datasets

Description

Calculate the dissimilarity matrix between two datasets of class distributional or compositional using the Kolmogorov-Smirnov, Sircombe-Hazelton, Aitchison or Bray Curtis distance

Usage

diss(x, method)

## S3 method for class 'distributional'
diss(x, method = NULL)

## S3 method for class 'compositional'
diss(x, method = NULL)

## S3 method for class 'counts'
diss(x, method = NULL)

Arguments

x    an object of class distributional, compositional or counts
method (optional) either "KS", "Kuiper", "SH", "aitchison", "bray" or "chisq"

Value

an object of class diss

Examples

data(Namib)
print(round(100*diss(Namib$DZ)))

endmembers

Petrographic end-member compositions

Description

A compositional dataset comprising the mineralogical compositions of the following end-members:
undissected_magmatic_arc, dissected_magmatic_arc, ophiolite, recycled_clastic, undissected_continental_block,
transitional_continental_block, dissected_continental_block, subcreted_axial_belt
and subducted_axial_belt
Author(s)
Alberto Resentini and Pieter Vermeesch

References

See Also
minsoring

Examples

data(endmembers,densities)
ophiolite <- subset(endmembers,select="ophiolite")
plot(minsorting(ophiolite,densities,by=0.05))

generate Nf

Calculate the largest fraction that is likely to be missed

Description
For a given sample size, returns the largest fraction which has been sampled with (1-p) x 100 % likelihood.

Usage
get.f(n, p = 0.05)

Arguments
n the number of grains in the detrital sample
p the required level of confidence

Value
the largest fraction that is sampled with at least (1-p) x 100% certainty

References
Examples

```r
print(get.f(60))
print(get.f(117))
```

<table>
<thead>
<tr>
<th>get.n</th>
<th>Calculate the number of grains required to achieve a desired level of sampling resolution</th>
</tr>
</thead>
</table>

**Description**

Returns the number of grains that need to be analysed to decrease the likelihood of missing any fraction greater than a given size below a given level.

**Usage**

```r
get.n(p = 0.05, f = 0.05)
```

**Arguments**

- `p`: the probability that all `n` grains in the sample have missed at least one fraction of size `f`
- `f`: the size of the smallest resolvable fraction (0<f<1)
- `n`: the number of grains in the sample

**Value**

the number of grains needed to reduce the chance of missing at least one fraction `f` of the total population to less than `p`

**References**


**Examples**

```r
# number of grains required to be 99% that no fraction greater than 5% was missed:
print(get.n(0.01))
# number of grains required to be 90% that no fraction greater than 10% was missed:
print(get.n(p=0.1,f=0.1))
```
### get.p

*Calculate the probability of missing a given population fraction*

**Description**

For a given sample size, returns the likelihood of missing any fraction greater than a given size.

**Usage**

```
get.p(n, f = 0.05)
```

**Arguments**

- `n`: the number of grains in the detrital sample
- `f`: the size of the smallest resolvable fraction (0 < f < 1)

**Value**

the probability that all `n` grains in the sample have missed at least one fraction of size `f`

**References**


**Examples**

```
print(get.p(60))
print(get.p(117))
```

---

### GPA

*Generalised Procrustes Analysis of configurations*

**Description**

Given a number of (2D) configurations, this function uses a combination of transformations (reflections, rotations, translations and scaling) to find a 'consensus' configuration which best matches all the component configurations in a least-squares sense.

**Usage**

```
GPA(X, scale = TRUE)
```

**Arguments**

- `X`: a list of dissimilarity matrices
- `scale`: boolean flag indicating if the transformation should include the scaling operation
Value

a two column vector with the coordinates of the group configuration

See Also

procrustes

indscal

Individual Differences Scaling of provenance data

Description


Usage

indscal(..., type = "ordinal")

Arguments

... a sequence of datasets of class distributional or compositional
type is either "ratio" or "ordinal"

Value

an object of class INDSCAL, i.e. a list containing the following items:
delta: Observed dissimilarities
obsdiss: List of observed dissimilarities, normalized
confdiss: List of configuration dissimilarities
conf: List of matrices of final configurations
gspace: Joint configurations aka group stimulus space
cweights: Configuration weights
stress: Stress-1 value
spp: Stress per point
sps: Stress per subject (matrix)
ndim: Number of dimensions
model: Type of smacof model
niter: Number of iterations
nobj: Number of objects
Author(s)
Jan de Leeuw and Patrick Mair

References

Examples
```
data(Namib)
plot(indscal(Namib$DZ,Namib$HM))
```

KDE

Create a kernel density estimate

Description

Usage
```
KDE(x, from = NA, to = NA, bw = NA, adaptive = TRUE, log = FALSE, n = 512, ...)
```

Arguments
- **x**: a vector of numbers
- **from**: minimum age of the time axis. If NULL, this is set automatically
- **to**: maximum age of the time axis. If NULL, this is set automatically
- **bw**: the bandwidth of the KDE. If NULL, bw will be calculated automatically using botev()
- **adaptive**: boolean flag controlling if the adaptive KDE modifier of Abramson (1982) is used
- **log**: transform the ages to a log scale if TRUE
- **n**: horizontal resolution of the density estimate
- **...**: optional arguments to be passed on to density

Value
an object of class KDE, i.e. a list containing the following items:
- **x**: horizontal plot coordinates
- **y**: vertical plot coordinates
- **bw**: the base bandwidth of the density estimate
- **ages**: the data values from the input to the KDE function
KDEs

See Also

KDEs

Examples

```r
data(Namib)
samp <- Namib$D$X[['N1']]
dens <- KDE(samp, 0, 3000, kernel = "epanechnikov")
plot(dens)
```

KDEs

Generate an object of class KDEs

Description

Convert a dataset of class distributional into an object of class KDEs for further processing by the `summaryplot` function.

Usage

```r
KDEs(x, from = NA, to = NA, bw = NA, samebandwidth = TRUE,
    adaptive = TRUE, normalise = FALSE, log = FALSE, n = 512, ...)
```

Arguments

- `x`: an object of class distributional
- `from`: minimum limit of the x-axis.
- `to`: maximum limit of the x-axis.
- `bw`: the bandwidth of the kernel density estimates. If `bw = NA`, the bandwidth will be set automatically using `botev()`.
- `samebandwidth`: boolean flag indicating whether the same bandwidth should be used for all samples. If `samebandwidth = TRUE` and `bw = NULL`, then the function will use the median bandwidth of all the samples.
- `normalise`: boolean flag indicating whether or not the KDEs should all integrate to the same value.
- `log`: boolean flag indicating whether the data should be plotted on a logarithmic scale.
- `n`: horizontal resolution of the density estimates
- `...`: optional parameters to be passed on to `density`
Value

an object of class KDEs, i.e. a list containing the following items:

- `kdes`: a named list with objects of class KDE
- `from`: the beginning of the common time scale
- `to`: the end of the common time scale
- `themax`: the maximum probability density of all the KDEs
- `pch`: the plot symbol to be used by `plot.KDEs`
- `xlabel`: the x-axis label to be used by `plot.KDEs`

See Also

KDE

Examples

data(Namib)
KDEs <- KDEs(Namib$DZ, 0, 3000, pch=NA)
summaryplot(KDEs, ncol=3)

KS.diss

Kolmogorov-Smirnov dissimilarity

Description

Returns the Kolmogorov-Smirnov dissimilarity between two samples

Usage

KS.diss(x, y)

Arguments

- `x`: the first sample as a vector
- `y`: the second sample as a vector

Value

a scalar value representing the maximum vertical distance between the two cumulative distributions

Examples

data(Namib)
print(KS.diss(Namib$DZ$x["N1"], Namib$DZ$x["T8"]))
Kuiper.diss

Kuiper dissimilarity

Description
Returns the Kuiper dissimilarity between two samples

Usage
Kuiper.diss(x, y)

Arguments
x the first sample as a vector
y the second sample as a vector

Value
a scalar value representing the sum of the maximum vertical distances above and below the cumulative distributions of x and y

Examples
data(Namib)
print(Kuiper.diss(Namib$DZ$x[[1]], Namib$DZ$x[[8]]))

lines.ternary

Ternary line plotting

Description
Add lines to an existing ternary diagram

Usage
## S3 method for class 'ternary'
lines(x, ...)

Arguments
x an object of class ternary, or a three-column data frame or matrix
... optional arguments to the generic lines function
Examples

```r
tern <- ternary(Namib$PT,'Q',c('KF','P'),c('Lm','Lv','Ls'))
plot(tern,pch=21,bg='red',labels=FALSE)
middle <- matrix(c(0.01,0.49,0.01,0.49,0.008,0.98),2,3)
lines(ternary(middle))
```

Description

Performs classical or nonmetric Multidimensional Scaling analysis of provenance data

Usage

```r
MDS(x, ...)
```

## S3 method for class 'compositional'
```r
MDS(x, classical = FALSE, k = 2, ...)
```

## S3 method for class 'counts'
```r
MDS(x, classical = FALSE, k = 2, ...)
```

## S3 method for class 'distributional'
```r
MDS(x, classical = FALSE, k = 2, ...)
```

## S3 method for class 'diss'
```r
MDS(x, classical = FALSE, k = 2, ...)
```

Arguments

- **x**: an object of class distributional, compositional or diss
- **...**: optional arguments to be passed onto diss (if x is of class compositional or distributional) or onto cmdscale or isomds (if x is of class dist).
- **classical**: boolean flag indicating whether classical (TRUE) or nonmetric (FALSE) MDS should be used
- **k**: the desired dimensionality of the solution

Value

an object of class MDS, i.e. a list containing the following items:
- **points**: a two column vector of the fitted configuration
- **classical**: a boolean flag indicating whether the MDS configuration was obtained by classical (TRUE) or nonmetric (FALSE) MDS.
- **diss**: the dissimilarity matrix used for the MDS analysis
- **stress**: (only if classical=TRUE) the final stress achieved (in percent)
**Examples**

```r
data(Namib)
plot(MDS(Namib$Major,classical=TRUE))
```

---

**Description**

Assess settling equivalence of detrital components

**Usage**

```r
minsoring(X, dens, sname = NULL, phi = 2, sigmaphi = 1,
          medium = "freshwater", from = -2.25, to = 5.5, by = 0.25)
```

**Arguments**

- `X`: an object of class `compositional`
- `dens`: a vector of mineral and rock densities
- `sname`: sample name if unspecified, the first sample of the dataset will be used
- `phi`: the mean grain size of the sample in Krumbein’s phi units
- `sigmaphi`: the standard deviation of the grain size distribution, in phi units
- `medium`: the transport medium, one of either "air", "freshwater" or "seawater"
- `from`: the minimum grain size to be evaluated, in phi units
- `to`: the maximum grain size to be evaluated, in phi units
- `by`: the grain size interval of the output table, in phi units

**Value**

an object of class `minsoring`, i.e. a list with two tables:

- `mfract`: the grain size distribution of each mineral (sum of the columns = 1)
- `mcomp`: the composition of each mineral (sum of the rows = 1)

**Author(s)**

Alberto Resentini and Pieter Vermeesch

**References**


### See Also

- restore

### Examples

```r
data(endmembers,densities)
distribution <- minsorting(endmembers,densities,sname='ophiolite',phi=2,
sigmaphi=1,medium="seawater",by=0.05)
plot(distribution,cumulative=FALSE)
```

### Description

A large dataset of provenance data from Namibia comprised of 14 sand samples from the Namib Sand Sea and 2 samples from the Orange River.

### Details

**Namib** is a list containing the following 6 items:

- **dz**: a distributional dataset containing the zircon U-Pb ages for ca. 100 grains from each sample, as well as their (1-sigma) analytical uncertainties.
- **pt**: a compositional dataset with the bulk petrography of the samples, i.e. the quartz ('Q'), K-feldspar ('KF'), plagioclase ('P'), and lithic fragments of metamorphic ('Lm'), volcanic ('Lv') and sedimentary ('Ls') origin.
- **hm**: a compositional dataset containing the heavy mineral composition of the samples, comprised of zircon ('rz'), tourmaline ('tm'), rutile ('rt'), Ti-oxides ('TiOx'), titanite ('sph'), apatite ('ap'), epidote ('ep'), garnet ('gt'), staurolite ('st'), andalusite ('and'), kyanite ('ky'), sillimanite ('sil'), amphibole ('amp'), clinopyroxene ('cpx') and orthopyroxene ('opx').
- **pthm**: a compositional dataset combining the variables contained in **pt** and **hm** plus 'mica', 'opaques', 'turbids' and 'other' transparent heavy minerals ('LgM'), normalised to 100.
- **major**: a compositional dataset listing the concentrations (in wt TiO2, P2O5 and MnO).
- **trace**: a compositional data listing the concentrations (in ppm) of Rb, Sr, Ba, Sc, Y, La, Ce, Pr, Nd, Sm, Gd, Dy, Er, Yb, Th, U, Zr, Hf, V, Nb, Cr, Co, Ni, Cu, Zn, Ga and Pb.

### Author(s)

Pieter Vermeesch and Eduardo Garzanti

### References

**PCA**

*Principal Component Analysis*

**Description**

Performs PCA of compositional data using a centred logratio distance

**Usage**

`PCA(x, ...)`

**Arguments**

- `x`: an object of class compositional
- `...`: optional arguments to R’s `princomp` function

**Value**

an object of classes `PCA`, which is synonymous to the stats packages’ `princomp` class.

**Examples**

```r
data(Namib)
samp <- Namib$DZx[['N1']]
dens <- KDE(samp,0,3000)
plot(dens)
```

---

**plot.CA**

*Point-counting biplot*

**Description**

Plot the results of a correspondence analysis as a biplot

**Usage**

```r
## S3 method for class 'CA'
plot(x, ...)
```
plot.compositional

Arguments

x an object of class CA

... optional arguments of the biplot function

See Also

CA

Examples

data(Namib)
plot(CA(Namib$PT))

plot.compositional Plot a pie chart

Description

Plots an object of class compositional as a pie chart

Usage

## S3 method for class 'compositional'
plot(x, sname, annotate = TRUE, colmap = NULL,
...)

Arguments

x an object of class compositional

sname the sample name

annotate a boolean flag controlling if the pies of the pie-chart should be labeled

colmap an optional string with the name of one of R’s built-in colour palettes (e.g., heat.colors, terrain.colors, topo.colors, cm.colors), which are to be used for plotting the data.

... optional parameters to be passed on to the graphics object

Examples

data(Namib)
plot(Namib$Major,'N',colmap='heat.colors')
plot.distributional  Plot continuous data as histograms or cumulative age distributions

Description

Plot one or several samples from a distributional dataset as a histogram or Cumulative Age Distributions (CAD).

Usage

```
## S3 method for class 'distributional'
plot(x, snames = NULL, annotate = TRUE,
     CAD = FALSE, pch = NA, verticals = TRUE, colmap = NULL, ...)
```

Arguments

- `x` an object of class `distributional`
- `snames` a string or a vector of string with the names of the samples that need plotting if `snames` is a vector, then the function will default to a CAD.
- `annotate` boolean flag indicating whether the x- and y-axis should be labeled
- `CAD` boolean flag indicating whether the data should be plotted as a cumulative age distribution or a histogram. For multi-sample plots, the function will override this value with `TRUE`.
- `pch` an optional symbol to mark the sample points along the CAD
- `verticals` boolean flag indicating if the horizontal lines of the CAD should be connected by vertical lines
- `colmap` an optional string with the name of one of R’s built-in colour palettes (e.g., heat.colors, terrain.colors, topo.colors, cm.colors), which are to be used for plotting the data.
- `...` optional arguments to the generic `plot` function

Examples

```
data(Namib)
plot(Namib$DZ,c('N1','N2'))
```
plot.GPA

Plot a Procrustes configuration

Description

Plots the group configuration of a Generalised Procrustes Analysis

Usage

```r
## S3 method for class 'GPA'
plot(x, pch = NA, pos = NULL, col = "black",
     bg = "white", cex = 1, ...)
```

Arguments

- `x`: an object of class GPA
- `pch`: plot symbol
- `pos`: position of the sample labels relative to the plot symbols if `pch` != NA
- `col`: plot colour (may be a vector)
- `bg`: background colour (may be a vector)
- `cex`: relative size of plot symbols
- `...`: optional arguments to the generic `plot` function

See Also

- `procrustes`

Examples

```r
data(Namib)
GPA <- procrustes(Namib$DZ, Namib$HM)
coast <- c('N1', 'N2', 'N3', 'N10', 'N11', 'N12', 'T8', 'T13')
snames <- names(Namib$DZ)
bgcol <- rep('yellow', length(snames))
bgcol[which(snames %in% coast)] <- 'red'
plot(GPA, pch=21, bg=bgcol)
```
plot.INDSCAL

Plot an INDSCAL group configuration and source weights

Description

Given an object of class INDSCAL, generates two plots: the group configuration and the subject weights. Together, these describe a 3-way MDS model.

Usage

## S3 method for class 'INDSCAL'
plot(x, asp = 1, pch = NA, pos = NULL,
col = "black", bg = "white", cex = 1, xlab = "X", ylab = "Y",
xaxt = "n", yaxt = "n", ...)

Arguments

x an object of class INDSCAL

asp the aspect ratio of the plot

pch plot symbol (may be a vector)
pos position of the sample labels relative to the plot symbols if pch != NA

col plot colour (may be a vector)

bg background colour (may be a vector)
cex relative size of plot symbols

xlab a string with the label of the x axis

ylab a string with the label of the y axis

xaxt if = 'y', adds ticks to the x axis

yaxt if = 'y', adds ticks to the y axis

... optional arguments to the generic plot function

See Also

indscal

Examples

data(Namib)
coast <- c('N1','N2','N3','N10','N11','N12','T8','T13')
snames <- names(Namib$DZ)
pch <- rep(21,length(snames))
pch[which(snames %in% coast)] <- 22
plot(indscal(Namib$DZ,Namib$HM),pch=pch)
plot.KDE

**Plot a kernel density estimate**

**Description**

Plots an object of class KDE

**Usage**

```r
## S3 method for class 'KDE'
plot(x, pch = "|", xlab = "age [Ma]", ylab = "", ...)  
```

**Arguments**

- `x`: an object of class KDE
- `pch`: the symbol used to show the samples. May be a vector. Set `pch = NA` to turn them off.
- `xlab`: the label of the x-axis
- `ylab`: the label of the y-axis
- `...`: optional parameters to be passed on to the graphics object

**See Also**

KDE

**Examples**

```r
data(Namib)
samp <- Namib$DZ$x[['N1']]
dens <- KDE(samp, from=0, to=3000)
plot(dens)
```

---

plot.KDEs

**Plot one or more kernel density estimates**

**Description**

Plots an object of class KDEs

**Usage**

```r
## S3 method for class 'KDEs'
plot(x, sname = NA, annotate = TRUE, pch = "|", ...)  
```
### Arguments

- **x**: an object of class `kdes`
- **sname**: optional sample name. If `sname=NA`, all samples are shown on a summary plot
- **annotate**: add a time axis?
- **pch**: symbol to be used to mark the sample points along the x-axis. Change to `NA` to omit.
- **...**: optional parameters to be passed on to the `summaryplot` function

### See Also

- `KDEs`
- `summaryplot`

### Examples

```r
data(Namib)
kdes <- KDEs(Namib$OZ)
plot(kdes, ncol=2)
```

### Description

Plots the coordinates of a multidimensional scaling analysis as an X-Y scatter plot or 'map' and, if `x$classical = FALSE`, a Shepard plot.

### Usage

```r
## S3 method for class 'MDS'
plot(x, nnlines = FALSE, pch = NA, pos = NULL,
     cex = 1, col = "black", bg = "white", oma = rep(1, 4),
     mar = rep(2, 4), mgp = c(2, 1, 0), xpd = NA, ...)
```

### Arguments

- **x**: an object of class `MDS`
- **nnlines**: if TRUE, draws nearest neighbour lines
- **pch**: plot character (see ?plot for details). May be a vector.
- **pos**: position of the sample labels relative to the plot symbols if `pch` != `NA`
- **cex**: relative size of plot symbols (see ?par for details)
- **col**: plot colour (may be a vector)
- **bg**: background colour (may be a vector)
- **oma**: A vector of the form `c(bottom, left, top, right)` giving the size of the outer margins in lines of text.
**plot.minsorting**

Plot inferred grain size distributions

**Description**

Plot the grain size distributions of the different minerals under consideration

**Usage**

```r
## S3 method for class 'minsoring'
plot(x, cumulative = FALSE, components = NULL,
     ...)  
```

**Arguments**

- `x` an object of class `minsoring`
- `cumulative` boolean flag indicating whether the grain size distribution should be plotted as a density or cumulative probability curve.
- `components` string or list of strings with the names of a subcomposition that needs plotting
- `...` optional parameters to be passed on to `graphics::matplot` (see `?par` for details)

**See Also**

`minsoring`
Examples

```r
data(endmembers, densities)
OPH <- subset(endmembers, select="ophiolite")
distribution <- minsorting(OPH, densities, phi=2, sigmaphi=1, medium="air", by=0.05)
plot(distribution, components=c('F','px','opaques'))
```

Description

Plot the results of a principal components analysis as a biplot

Usage

```r
## S3 method for class 'PCA'
plot(x, ...)
```

Arguments

- `x`: an object of class PCA
- `...`: optional arguments of the `biplot` function

See Also

PCA

Examples

```r
data(Namib)
plot(PCA(Namib$Major))
```

Description

Plot a ternary diagram

Usage

```r
## S3 method for class 'ternary'
plot(x, type = "grid", pch = NA, pos = NULL,
     labels = names(x), showpath = FALSE, bg = NA,
     col = "cornflowerblue", ticks = seq(0, 1, 0.25), ticklength = 0.02,
     lty = 2, lwd = 1, ...)
```
Arguments

- **x**: an object of class `ternary`, or a three-column data frame or matrix
- **type**: adds annotations to the ternary diagram, one of either `empty`, `grid`, `QFL.descriptive`, `QFL.folk` or `QFL.dickinson`
- **pch**: plot character, see `?par` for details (may be a vector)
- **pos**: position of the sample labels relative to the plot symbols if pch != NA
- **labels**: vector of strings to be added to the plot symbols
- **showpath**: if x has class `srdcorrected`, and showpath == TRUE, the intermediate values of the SRD correction will be plotted on the ternary diagram as well as the final composition
- **bg**: background colour for the plot symbols (may be a vector)
- **col**: colour to be used for the background lines (if applicable)
- **ticks**: vector of tick values between 0 and 1
- **ticklength**: number between 0 and 1 to mark the length of the ticks
- **lty**: line type for the annotations (see type)
- **lwd**: line thickness for the annotations
- **...**: optional arguments to the generic `points` function

See Also

`ternary`

Examples

```r
data(Namib)
tern <- ternary(Namib$PT,'Q',c('KF','P'),c('Lm','Lv','Ls'))
plot(tern,type='QFL.descriptive',pch=21,bg='red',labels=NULL)
```

Description

Add points to an existing ternary diagram

Usage

```r
## S3 method for class 'ternary'
points(x, ...)
```

Arguments

- **x**: an object of class `ternary`, or a three-column data frame or matrix
- **...**: optional arguments to the generic `points` function
Examples

tern <- ternary(Namib$PT,'Q',c('KF','P'),c('Lm','Lv','Ls'))
plot(tern,pch=21,bg='red',labels=FALSE)
# add the geometric mean composition as a yellow square:
gmean <- ternary(exp(colMeans(log(tern$x))))
points(gmean,pch=22,bg='yellow')

data(Namib)
gpa <- procrustes(Namib$DZ,Namib$HM)
plot(gpa)

procrustes

Generalised Procrustes Analysis of provenance data

Description

Given a number of input datasets, this function performs an MDS analysis on each of these and the
feeds the resulting configurations into the GPA() function.

Usage

procrustes(...)

Arguments

... a sequence of datasets of classes distributional,n counts and compositional

Value

an object of class GPA, i.e. a list containing the following items:
points: a two column vector with the coordinates of the group configuration
labels: a list with the sample names

Author(s)

Pieter Vermeesch

References


See Also

GPA

Examples

data(Namib)
gpa <- procrustes(Namib$DZ,Namib$HM)
plot(gpa)
provenance

Menu-based interface for provenance

Description

For those less familiar with the syntax of the R programming language, the provenance() function provides a user-friendly way to access the most important functionality in the form of a menu-based query interface. Further details and examples are provided on http://provenance.london-geochron.com.

provenance provides statistical tools to interpret large amounts of distributional (single grain analyses) and compositional (mineralogical and bulk chemical) data from the command line, or using a menu-based user interface.

Usage

provenance()

Details

A list of documented functions may be viewed by typing help(package='provenance'). Detailed instructions are provided at http://provenance.london-geochron.com and in the Sedimentary Geology paper by Vermeesch, Resentini and Garzanti (2016).

Author(s)

Pieter Vermeesch

Maintainer: Pieter Vermeesch <p.vermeesch@ucl.ac.uk>

References


See Also

http://provenance.london-geochron.com

Useful links:

- http://provenance.london-geochron.com
radialplot

Visualise point-counting data on a radial plot

Description

Implementation of a graphical device developed by Rex Galbraith to display several estimates of the same quantity that have different standard errors.

Usage

radialplot(x, num = 1, den = 2, from = NA, to = NA, t0 = NA, 
sigdig = 2, show.numbers = FALSE, pch = 21, levels = NA, 
clabel = "", bg = c("white", "red"), title = TRUE, ...)

Arguments

x an object of class counts
num index or name of the numerator variable
den index or name of the denominator variable
from minimum limit of the radial scale
to maximum limit of the radial scale
t0 central value
sigdig the number of significant digits of the numerical values reported in the title of the graphical output.
show.numbers boolean flag (TRUE to show sample numbers)
pch plot character (default is a filled circle)
levels a vector with additional values to be displayed as different background colours of the plot symbols.
clabel label of the colour legend
bg a vector of two background colours for the plot symbols. If levels=NA, then only the first colour is used. If levels is a vector of numbers, then bg is used to construct a colour ramp.
title add a title to the plot?
... additional arguments to the generic points function

Details

The radial plot (Galbraith, 1988, 1990) is a graphical device that was specifically designed to display heteroscedastic data, and is constructed as follows. Consider a set of dates \{t_1, ..., t_i, ..., t_n\} and uncertainties \{s[t_1], ..., s[t_i], ..., s[t_n]\}. Define \(z_i = z[t_i]\) to be a transformation of \(t_i\) (e.g., \(z_i = \log(t_i)\)), and let \(s[z_i]\) be its propagated analytical uncertainty (i.e., \(s[z_i] = s[t_i]/t_i\) in the case of a logarithmic transformation). Create a scatterplot of \((x_i, y_i)\) values, where \(x_i = 1/s[z_i]\) and \(y_i = (z_i - z_0)/s[z_i]\), where \(z_0\) is some reference value such as the mean. The slope of a line
connecting the origin of this scatterplot with any of the \((x_i, y_i)s\) is proportional to \(z_i\) and, hence, the date \(t_i\). These dates can be more easily visualised by drawing a radial scale at some convenient distance from the origin and annotating it with labelled ticks at the appropriate angles. While the angular position of each data point represents the date, its horizontal distance from the origin is proportional to the precision. Imprecise measurements plot on the left hand side of the radial plot, whereas precise age determinations are found further towards the right. Thus, radial plots allow the observer to assess both the magnitude and the precision of quantitative data in one glance.

References


Examples

data(Namib)
radialplot(Namib$PT, components=c('Q','P'))

read.compositional Read a .csv file with compositional data

Description
Reads a data table containing compositional data (e.g. chemical concentrations)

Usage
read.compositional(fname, method = NULL, colmap = "rainbow", ...)

Arguments
fname a string with the path to the .csv file
method either "bray" (for the Bray-Curtis distance) or "aitchison" (for Aitchison’s central logratio distance). If omitted, the function defaults to ‘aitchison’, unless there are zeros present in the data.
colmap an optional string with the name of one of R’s built-in colour palettes (e.g., heat.colors, terrain.colors, topo.colors, cm.colors), which are to be used for plotting the data.
... optional arguments to the built-in read.csv function
**Value**

an object of class `compositional`, i.e. a list with the following items:

- **x**: a data frame with the samples as rows and the categories as columns
- **method**: either "aitchison" (for Aitchison's centred logratio distance) or "bray" (for the Bray-Curtis distance)
- **colmap**: the colour map provided by the input argument

**Examples**

```r
fname <- system.file("Major.csv", package="provenance")
Major <- read.compositional(fname)
plot(PCA(Major))
```

```r
read.counts(fname, method = "chisq", colmap = "rainbow", ...)
```

**Description**

Reads a data table containing point-counting data (e.g. petrographic, heavy mineral, palaeontological or palynological data)

**Usage**

```r
read.counts(fname, method = "chisq", colmap = "rainbow", ...)
```

**Arguments**

- **fname**: a string with the path to the .csv file
- **method**: either "chisq" (for the chi-square distance) or "bray" (for the Bray-Curtis distance)
- **colmap**: an optional string with the name of one of R's built-in colour palettes (e.g., heat.colors, terrain.colors, topo.colors, cm.colors), which are to be used for plotting the data.
- **...**: optional arguments to the built-in `read.csv` function

**Value**

an object of class `counts`, i.e. a list with the following items:

- **x**: a data frame with the samples as rows and the categories as columns
- **colmap**: the colour map provided by the input argument

**Examples**

```r
fname <- system.file("HM.csv", package="provenance")
Major <- read.counts(fname)
# plot(PCA(HM))
```
**read.distributions**

*Read a .csv file with mineral and rock densities*

**Description**

Reads a data table containing densities to be used for hydraulic sorting corrections (minsoring and srd functions)

**Usage**

```r
read.densities(fname, ...)  
```

**Arguments**

`fname`  
a string with the path to the .csv file

`...`  
optional arguments to the built-in `read.csv` function

**Value**

a vector with mineral and rock densities

**Examples**

```r
data(Namib,densities)
N8 <- subset(Namib[,1:5],select="N8")
distribution <- minsorting(N8,densities,phi=2,sigmaphi=1,medium="air",by=0.05)
plot(distribution)
```

---

**read.distributional**

*Read a .csv file with continuous (detrital zircon) data*

**Description**

Reads a data table containing continuous data (e.g. detrital zircon ages)

**Usage**

```r
read.distributional(fname, errorfile = NA, method = "KS",
xlab = "age [Ma]", colmap = "rainbow", ...)
```

---


Arguments

fname        the path of a .csv file with the input data, arranged in columns.
errorfile    the (optional) path of a .csv file with the standard errors of the input data, arranged by column in the same order as fname. Must be specified if the data are to be compared with the Sircombe-Hazelton dissimilarity.
method       an optional string specifying the dissimilarity measure which should be used for comparing this with other datasets. Should be one of either "KS" (for Kolmogorov-Smirnov) or "SH" (for Sircombe and Hazelton). If method = "SH", then errorfile should be specified. If method = "SH" and errorfile is unspecified, then the program will default back to the Kolmogorov-Smirnov dissimilarity.
xlab         an optional string specifying the nature and units of the data. This string is used to label kernel density estimates.
colmap       an optional string with the name of one of R's built-in colour palettes (e.g., heat.colors, terrain.colors, topo.colors, cm.colors), which are to be used for plotting the data.

...         optional arguments to the built-in read.csv function

Value

an object of class distributional, i.e. a list with the following items:
x: a named list of vectors containing the numerical data for each sample
err: an (optional) named list of vectors containing the standard errors of x
method: either "KS" (for Kolmogorov-Smirnov), "Kuiper" (for the Kuiper statistic) or "SH" (for Sircombe Hazelton)
breaks: a vector with the locations of the histogram bin edges
xlab: a string containing the label to be given to the x-axis on all plots
colmap: the colour map provided by the input argument

Examples

agefile <- system.file("DZ.csv",package="provenance")
errfile <- system.file("DZerr.csv",package="provenance")
DZ <- read.distributional(agefile,errfile)
plot(KDE(DZ$x$N1))

Description

Undo the effect of hydraulic sorting

Restore the detrital composition back to a specified source rock density (SRD)
Usage

```r
restore(X, dens, target = 2.71)
```

Arguments

- `X`: an object of class `compositional`
- `dens`: a vector of rock and mineral densities
- `target`: the target density (in g/cm³)

Value

an object of class `SRDcorrected`, i.e. an object of class `compositional` which is a daughter of class `compositional` containing the restored composition, plus one additional member called `restoration`, containing the intermediate steps of the SRD correction algorithm.

Author(s)

Alberto Resentini and Pieter Vermeesch

References


See Also

`minsorting`

Examples

```r
data(Namib,densities)
rescomp <- restore(Namib$PTHM,densities,2.71)
HMcomp <- c("zr","tm","rt","sph","ap","ep","gt",
"st","amp","cpx","opx")
amcomp <- amalgamate(rescomp,Flag="P",HM=HMcomp,Opq="opales")
plot(ternary(amcomp),showpath=TRUE)
```

---

**SH.diss**  
*Sircombe and Hazelton distance*

Description

Calculates Sircombe and Hazelton’s L2 distance between the Kernel Functional Estimates (KFEs, not to be confused with Kernel Density Estimates!) of two samples with specified analytical uncertainties.
subset

Usage

SH.diss(x, i, j, c.con = 0)

Arguments

x an object of class distributional
i index of the first sample
j index of the second sample
c.con smoothing bandwidth of the kernel functional estimate

Value

da scalar value expressing the L2 distance between the KFEs of samples i and j

Author(s)

Keith Sircombe and Martin Hazelton

References


See Also

KS.diss

Examples

datfile <- system.file("DZ.csv", package="provenance")
errfile <- system.file("DZerr.csv", package="provenance")
DZ <- read.distributional(datfile, errfile)
d <- SH.diss(DZ, 1, 2)
print(d)

Description

Return a subset of provenance data according to some specified indices
Usage

```r
## S3 method for class 'distributional'
subset(x, subset = NULL, select = NULL, ...)

## S3 method for class 'compositional'
subset(x, subset = NULL, components = NULL, select = NULL, ...)

## S3 method for class 'counts'
subset(x, subset = NULL, components = NULL, select = NULL, ...)
```

Arguments

- `x`: an object of class `distributional`
- `subset`: logical expression indicating elements or rows to keep; missing values are taken as false.
- `select`: a vector of sample names
- `components`: categories to keep

Value

an object of class `distributional`

See Also

- `read.distributional`

Examples

```r
data(Namib)
coast <- c("N1","N2","T8","T13","N12","N13")
ZTRcoast <- subset(Namib$HM,select=coast,components=c('gt','cpx','ep'))
DZcoast <- subset(Namib$DZ,select=coast)
summaryplot(ZTRcoast,KDEs(DZcoast),ncol=2)
```

---

**summaryplot**  
*Joint plot of several provenance datasets*

Description

Arranges kernel density estimates and pie charts in a grid format

Usage

```r
summaryplot(..., ncol = 1, pch = NA)
```
Arguments

... a sequence of datasets of class compositional, KDEs, or distributional
ncol the number of columns
pch (optional) symbol to be used to mark the sample points along the x-axis of the KDEs (if appropriate).

Value

a summary plot of all the data comprised of KDEs for the datasets of class KDEs, pie charts for those of class compositional and histograms for those of class distributional.

See Also

KDEs

Examples

data(Namib)
KDEs <- KDEs(Namib$OZ, 0, 3000)
summaryplot(KDEs, Namib$HM, Namib$PT, ncol=2)

ternary

Definition a ternary composition

Description

Create an object of class ternary

Usage

ternary(X, x = 1, y = 2, z = 3)

Arguments

X an object of class compositional OR a matrix or data frame with numerical data
x string/number or a vector of strings/numbers indicating the variables/indices making up the first subcomposition of the ternary system.
y second (set of) variables
z third (set of) variables

Value

an object of class ternary, i.e. a list containing:
x: a three column matrix (or vector) of ternary compositions.
and (if X is of class SRDcorrected)
restoration: a list of intermediate ternary compositions inherited from the SRD correction
ternary.ellipse

Description

plot a 100(1 - \(\alpha\))% confidence region around the data or around its mean.

Usage

ternary.ellipse(x, ...)

## Default S3 method:
ternary.ellipse(x, alpha = 0.05, population = TRUE, ...)

## S3 method for class 'compositional'
ternary.ellipse(x, alpha = 0.05, population = TRUE, ...)

## S3 method for class 'counts'
ternary.ellipse(x, alpha = 0.05, population = TRUE, ...)

Arguments

- **x**: an object of class ternary
- **...**: optional formatting arguments
- **alpha**: cutoff level for the confidence ellipse
- **population**: show the standard deviation of the entire population or the standard error of the mean?

Examples

data(Namib)
tern <- ternary(Namib$PT, c('Q'), c('KF', 'P'), c('Lm', 'Lv', 'Ls'))
plot(tern, type="QFL")

---

ternary.ellipse

Ternary confidence ellipse

See Also

restore

Examples

data(Namib)
tern <- ternary(Namib$Major, 'CaO', 'Na2O', 'K2O')
plot(tern)
ternary.ellipse(tern)
Description

Add text to an existing ternary diagram

Usage

```
## S3 method for class 'ternary'
text(x, labels = 1:nrow(x$x), ...)
```

Arguments

- `x` an object of class `ternary`, or a three-column data frame or matrix
- `labels` a character vector or expression specifying the text to be written
- `...` optional arguments to the generic `text` function

Examples

```r
data(Namib)
tern <- ternary(Namib$Major, 'CaO', 'Na2O', 'K2O')
plot(tern, pch=21, bg='red', labels=NULL)
# add the geometric mean composition as a text label:
gmean <- ternary(exp(colMeans(log(tern$x))))
text(gmean, labels='geometric mean')
```
Index

_PACKAGE (provenance), 36
ALR, 3
amalgamate, 4
as.acomp, 5
as.compositional, 5
as.counts, 6
as.data.frame, 7
botev, 7
bray.diss, 8
CA, 9
central, 9
CLR, 10
combine, 11
densities, 12
diss, 13
endmembers, 13
get.f, 14
get.n, 15
get.p, 16
GPA, 16
indscal, 17
KDE, 18
KDEs, 19
KS.diss, 20
Kuiper.diss, 21
lines(points.ternary), 34
lines.ternary, 21
MDS, 22
minsorting, 23
Namib, 24
PCA, 25
plot.CA, 25
plot.compositional, 26
plot.distributional, 27
plot.GPA, 28
plot.INDSCAL, 29
plot.KDE, 30
plot.KDEs, 30
plot.MDS, 31
plot.minsorting, 32
plot.PCA, 33
plot.ternary, 33
points.ternary, 34
procrustes, 35
provenance, 36
provenance-package (provenance), 36
radialplot, 37
read.compositional, 38
read.counts, 39
read.densities, 40
read.distributional, 40
restore, 41
SH.diss, 42
subset, 43
summaryplot, 44
ternary, 45
ternary.ellipse, 46
text (points.ternary), 34
text.ternary, 47