

# Package ‘provenance’

June 18, 2022

**Title** Statistical Toolbox for Sedimentary Provenance Analysis

**Version** 4.0

**Date** 2022-06-15

**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** <https://www.ucl.ac.uk/~ucfbpve/provenance/>

**License** GPL-2

**LazyData** true

**RoxygenNote** 7.2.0

**Encoding** UTF-8

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2022-06-18 06:50:02 UTC

**R topics documented:**

|                               |    |
|-------------------------------|----|
| ALR . . . . .                 | 3  |
| amalgamate . . . . .          | 4  |
| as.acomp . . . . .            | 5  |
| as.compositional . . . . .    | 6  |
| as.counts . . . . .           | 6  |
| as.data.frame . . . . .       | 7  |
| as.varietal . . . . .         | 8  |
| botev . . . . .               | 9  |
| bray.diss . . . . .           | 9  |
| CA . . . . .                  | 10 |
| central . . . . .             | 11 |
| CLR . . . . .                 | 12 |
| combine . . . . .             | 13 |
| densities . . . . .           | 13 |
| diss . . . . .                | 14 |
| endmembers . . . . .          | 15 |
| get.f . . . . .               | 16 |
| get.n . . . . .               | 16 |
| get.p . . . . .               | 17 |
| GPA . . . . .                 | 18 |
| indscal . . . . .             | 18 |
| KDE . . . . .                 | 20 |
| KDEs . . . . .                | 21 |
| KS.diss . . . . .             | 22 |
| Kuiper.diss . . . . .         | 23 |
| lines.ternary . . . . .       | 23 |
| MDS . . . . .                 | 24 |
| minsorting . . . . .          | 25 |
| Namib . . . . .               | 27 |
| PCA . . . . .                 | 28 |
| plot.CA . . . . .             | 28 |
| plot.compositional . . . . .  | 29 |
| plot.distributional . . . . . | 30 |
| plot.GPA . . . . .            | 31 |
| plot.INDSCAL . . . . .        | 32 |
| plot.KDE . . . . .            | 33 |
| plot.KDEs . . . . .           | 34 |
| plot.MDS . . . . .            | 34 |
| plot.minsorting . . . . .     | 36 |
| plot.PCA . . . . .            | 37 |
| plot.ternary . . . . .        | 38 |
| points.ternary . . . . .      | 39 |
| procrustes . . . . .          | 39 |
| provenance . . . . .          | 40 |
| radialplot . . . . .          | 41 |
| read.compositional . . . . .  | 43 |

|                                   |    |
|-----------------------------------|----|
| read.counts . . . . .             | 44 |
| read.densities . . . . .          | 45 |
| read.distributional . . . . .     | 46 |
| read.varietal . . . . .           | 48 |
| restore . . . . .                 | 49 |
| SH.diss . . . . .                 | 50 |
| SNSM . . . . .                    | 51 |
| subset . . . . .                  | 51 |
| summaryplot . . . . .             | 52 |
| ternary . . . . .                 | 53 |
| ternary.ellipse . . . . .         | 54 |
| text.ternary . . . . .            | 54 |
| varietal2distributional . . . . . | 55 |

**Index 56**

ALR *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, ...)
```

## S3 method for class 'varietal'

```
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=c("KF","P"),
                 lithics=c("Lm","Lv","Ls"),heavies=HMcomponents)
plot(ternary(am))
```

---

|          |                               |
|----------|-------------------------------|
| as.acomp | <i>create an acomp object</i> |
|----------|-------------------------------|

---

## 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

|                     |                                                                                                                                          |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <code>x</code>      | an object of class <code>matrix</code> , <code>data.frame</code> or <code>acomp</code>                                                   |
| <code>method</code> | dissimilarity measure, either <code>"aitchison"</code> for Aitchison's CLR-distance or <code>"bray"</code> for the Bray-Curtis distance. |
| <code>colmap</code> | 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 'provenance' with 'compositions'
# library(compositions)
# data(Glacial)
# a.glac <- acomp(Glacial)
# 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 <code>matrix</code> or <code>data.frame</code>                     |
| 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 | <i>create a data.frame object</i> |
|---------------|-----------------------------------|

---

**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 <code>compositional</code>              |
| ... | 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))
```

---

|             |                                 |
|-------------|---------------------------------|
| as.varietal | <i>create a varietal object</i> |
|-------------|---------------------------------|

---

## Description

Convert an object of class `matrix` or `data.frame` to an object of class `varietal`

## Usage

```
as.varietal(x, snames = NULL)
```

## Arguments

|                     |                                                                                                                                                                                                                                                                                   |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>x</code>      | an object of class <code>matrix</code> or <code>data.frame</code>                                                                                                                                                                                                                 |
| <code>snames</code> | either a vector of sample names, an integer marking the length of the sample name prefix, or <code>NULL</code> . <code>read.varietal</code> assumes that the row names of the <code>.csv</code> file consist of character strings marking the sample names, followed by a number. |

## Value

an object of class `varietal`

## Examples

```
data(SNSM)
ap1 <- SNSM$ap
ap2 <- as.varietal(x=ap1$x, snames=ap1$snames)
```



---

|       |                                             |
|-------|---------------------------------------------|
| botev | <i>Compute the optimal kernel bandwidth</i> |
|-------|---------------------------------------------|

---

**Description**

Uses the diffusion algorithm of 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)**

Zdravko Botev

**References**

Botev, Z. I., J. F. Grotowski, and D. P. Kroese. "Kernel density estimation via diffusion." *The Annals of Statistics* 38.5 (2010): 2916-2957.

**Examples**

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

---

|           |                                  |
|-----------|----------------------------------|
| bray.diss | <i>Bray-Curtis dissimilarity</i> |
|-----------|----------------------------------|

---

**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$HM$x["N1",], Namib$HM$x["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 package's correspondence class.

**Examples**

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

---

|         |                                       |
|---------|---------------------------------------|
| central | <i>Calculate central compositions</i> |
|---------|---------------------------------------|

---

### 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 x 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 `theta`.

**UL** the upper limit of a ‘1 sigma’ region for `theta`.

**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$Major))  
dev.new()  
clrdat <- CLR(Namib$Major)  
biplot(princomp(clrdat))
```

---

|         |                                               |
|---------|-----------------------------------------------|
| combine | <i>Combine samples of distributional data</i> |
|---------|-----------------------------------------------|

---

**Description**

Lumps all single grain analyses of several samples together under a new name

**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 |

**Value**

a distributional data object with fewer samples than x

**Examples**

```
data(Namib)
combined <- combine(Namib$DZ,
                   east=c('N3', 'N4', 'N5', 'N6', 'N7', 'N8', 'N9', 'N10'),
                   west=c('N1', 'N2', 'N11', 'N12', 'T8', 'T13'))
summaryplot(KDEs(combined))
```

---

|           |                                             |
|-----------|---------------------------------------------|
| densities | <i>A list of rock and mineral densities</i> |
|-----------|---------------------------------------------|

---

**Description**

List of rock and mineral densities using the following abbreviations: Q (quartz), KF (K-feldspar), P (plagioclase), F (feldspar), Lv (felsic/porphyritic volcanic rock fragments), Lvm (microlithic / porphyritic / trachitic volcanic rock fragments), Lcc (calcite), Lcd (dolomite), Lp (marl), Lch (chert), Lms (argillaceous / micaceous rock fragments), Lmv (metavolcanics), Lmf (metasediments), Lmb (metabasites), Lv (volcanic rock fragments), Lc (carbonates), Ls (sedimentary rock fragments), Lm (metamorphic rock fragments), Lu (serpentinite), mica, opaques, FeOx (Fe-oxides), turbids, zr (zircon), tm (tourmaline), rt (rutile), TiOx (Ti-oxides), sph (titanite), ap (apatite), mon (monazite), oth (other minerals), ep (epidote), othLgM (prehnite + pumpellyite + lawsonite + carpholite), gt (garnet), ctd (chloritoid), st (staurolite), and (andalusite), ky (kyanite), sil (sillimanite), amp (amphibole), px (pyroxene), cpx (clinopyroxene), opx (orthopyroxene), ol (olivine), spinel and othHM (other heavy minerals).

**Author(s)**

Alberto Resentini and Pieter Vermeesch

**References**

Resentini, A, Malusa M G and Garzanti, E. "MinSORTING: An Excel worksheet for modelling mineral grain-size distribution in sediments, with application to detrital geochronology and provenance studies." *Computers & Geosciences* 59 (2013): 90-97.

Garzanti, E, Ando, S and Vezzoli, G. "Settling equivalence of detrital minerals and grain-size dependence of sediment composition." *Earth and Planetary Science Letters* 273.1 (2008): 138-151.

**See Also**

restore, minsorting

**Examples**

```
N8 <- subset(Namib$HM,select="N8")
distribution <- minsorting(N8,densities,phi=2,sigmaphi=1,medium="air",by=0.05)
plot(distribution)
```

---

diss

*Calculate the dissimilarity matrix between two datasets of class  
distributional, compositional, counts or varietal*

---

**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)

## S3 method for class 'varietal'
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 | <i>Petrographic end-member compositions</i> |
|------------|---------------------------------------------|

---

**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**

Resentini, A, Malusa M G and Garzanti, E. "MinSORTING: An Excel worksheet for modelling mineral grain-size distribution in sediments, with application to detrital geochronology and provenance studies." *Computers & Geosciences* 59 (2013): 90-97.

Garzanti, E, Ando, S and Vezzoli, G. "Settling equivalence of detrital minerals and grain-size dependence of sediment composition." *Earth and Planetary Science Letters* 273.1 (2008): 138-151.

**See Also**

`minsorting`

**Examples**

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

---

get.f                      *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  $(1-p) \times 100\%$  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  $(1-p) \times 100\%$  certainty

**References**

Vermeesch, Pieter. "How many grains are needed for a provenance study?" *Earth and Planetary Science Letters* 224.3 (2004): 441-451.

**Examples**

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

---

get.n                      *Calculate the number of grains required to achieve a desired level of sampling resolution*

---

**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**

```
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**

Vermeesch, Pieter. "How many grains are needed for a provenance study?." Earth and Planetary Science Letters 224.3 (2004): 441-451.

**Examples**

```
# 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**

Vermeesch, Pieter. "How many grains are needed for a provenance study?." Earth and Planetary Science Letters 224.3 (2004): 441-451.

**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**

Performs 3-way Multidimensional Scaling analysis using Carroll and Chang (1970)’s INdividual Differences SCALing method as implemented using De Leeuw and Mair (2011)’s stress majorization algorithm.

**Usage**

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

**Arguments**

... a sequence of datasets of class `distributional`, `compositional`, `counts` or `varietal`, OR a single object of class `varietal`.

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**

de Leeuw, J., & Mair, P. (2009). Multidimensional scaling using majorization: The R package `smacof`. *Journal of Statistical Software*, 31(3), 1-30, <<https://www.jstatsoft.org/v31/i03/>>

**Examples**

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

KDE

*Create a kernel density estimate***Description**

Turns a vector of numbers into an object of class KDE using a combination of the Botev (2010) bandwidth selector and the Abramson (1982) adaptive kernel bandwidth modifier.

**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 <code>botev()</code> |
| 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 <code>density</code>                                        |

**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

**See Also**

KDEs

**Examples**

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

**Description**

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

**Usage**

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

**Arguments**

|                            |                                                                                                                                                                                                                          |
|----------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>x</code>             | an object of class <code>distributional</code>                                                                                                                                                                           |
| <code>from</code>          | minimum limit of the x-axis.                                                                                                                                                                                             |
| <code>to</code>            | maximum limit of the x-axis.                                                                                                                                                                                             |
| <code>bw</code>            | the bandwidth of the kernel density estimates. If <code>bw = NA</code> , the bandwidth will be set automatically using <code>botev()</code>                                                                              |
| <code>samebandwidth</code> | boolean flag indicating whether the same bandwidth should be used for all samples. If <code>samebandwidth = TRUE</code> and <code>bw = NULL</code> , then the function will use the median bandwidth of all the samples. |
| <code>adaptive</code>      | boolean flag switching on the adaptive bandwidth modifier of Abramson (1982)                                                                                                                                             |
| <code>normalise</code>     | boolean flag indicating whether or not the KDEs should all integrate to the same value.                                                                                                                                  |
| <code>log</code>           | boolean flag indicating whether the data should be plotted on a logarithmic scale.                                                                                                                                       |
| <code>n</code>             | horizontal resolution of the density estimates                                                                                                                                                                           |
| <code>...</code>           | optional parameters to be passed on to <code>density</code>                                                                                                                                                              |

**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 | <i>Kuiper dissimilarity</i> |
|-------------|-----------------------------|

---

**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[['N1']], Namib$DZ$x[['T8']]))
```

---

|               |                              |
|---------------|------------------------------|
| lines.ternary | <i>Ternary line plotting</i> |
|---------------|------------------------------|

---

**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**

```
tern <- ternary(Namib$PT, 'Q', c('KF', 'P'), c('Lm', 'Lv', 'Ls'))
plot(tern, pch=21, bg='red', labels=NULL)
middle <- matrix(c(0.01, 0.49, 0.01, 0.49, 0.98, 0.02), 2, 3)
lines(ternary(middle))
```

MDS

*Multidimensional Scaling***Description**

Performs classical or nonmetric Multidimensional Scaling analysis of provenance data

**Usage**

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

## Default S3 method:

```
MDS(x, classical = FALSE, k = 2, ...)
```

## S3 method for class 'compositional'

```
MDS(x, classical = FALSE, k = 2, ...)
```

## S3 method for class 'counts'

```
MDS(x, classical = FALSE, k = 2, ...)
```

## S3 method for class 'distributional'

```
MDS(x, classical = FALSE, k = 2, nb = 0, ...)
```

## S3 method for class 'varietal'

```
MDS(x, classical = FALSE, k = 2, ...)
```

**Arguments**

|                        |                                                                                                                                                                                                                                                                                        |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>x</code>         | an object of class <code>distributional</code> , <code>compositional</code> , <code>counts</code> , <code>varietal</code> or <code>diss</code>                                                                                                                                         |
| <code>...</code>       | optional arguments to be passed onto <code>cmdscale</code> (if <code>classical=TRUE</code> ) or <code>isoMDS</code> (if <code>classical=FALSE</code> ).                                                                                                                                |
| <code>classical</code> | boolean flag indicating whether classical ( <code>TRUE</code> ) or nonmetric ( <code>FALSE</code> ) MDS should be used                                                                                                                                                                 |
| <code>k</code>         | the desired dimensionality of the solution                                                                                                                                                                                                                                             |
| <code>nb</code>        | number of bootstrap resamples. If <code>nb&gt;0</code> , then <code>plot.MDS(...)</code> will visualise the sampling uncertainty as polygons (inspired by Nordsvan et al. 2020). The bigger <code>nb</code> , the slower the calculations. <code>nb=10</code> seems a good compromise. |



**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)

**References**

Nordsvan, A.R., Kirscher, U., Kirkland, C.L., Barham, M. and Brennan, D.T., 2020. Resampling (detrital) zircon age distributions for accurate multidimensional scaling solutions. *Earth-Science Reviews*, p.103149.

Vermeesch, P., 2013, Multi-sample comparison of detrital age distributions. *Chemical Geology* v.341, 140-146, doi:10.1016/j.chemgeo.2013.01.010

**Examples**

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

---

minsorting

*Assess settling equivalence of detrital components*

---

**Description**

Models grain size distribution of minerals and rock fragments of different densities

**Usage**

```
minsorting(
  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 <code>compositional</code>                            |
| 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 `minsorting`, 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**

Resentini, A, Malusa, M G and Garzanti, E. "MinSORTING: An Excel worksheet for modelling mineral grain-size distribution in sediments, with application to detrital geochronology and provenance studies." *Computers & Geosciences* 59 (2013): 90-97.

Garzanti, E, Ando, S and Vezzoli, G. "Settling equivalence of detrital minerals and grain-size dependence of sediment composition." *Earth and Planetary Science Letters* 273.1 (2008): 138-151.

**See Also**

`restore`

**Examples**

```
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 ('zr'), 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 TiO<sub>2</sub>, P<sub>2</sub>O<sub>5</sub> 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

Vermeesch, P. and Garzanti, E., Making geological sense of 'Big Data' in sedimentary provenance analysis, *Chemical Geology* 409 (2015) 20-27

## Examples

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

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` package's `prcomp` class.

**Examples**

```
data(Namib)
plot(MDS(Namib$Major,classical=TRUE))
dev.new()
plot(PCA(Namib$Major),asp=1)
print("This example demonstrates the equivalence of classical MDS and PCA")
```

plot.CA

*Point-counting biplot***Description**

Plot the results of a correspondence analysis as a biplot

**Usage**

```
## S3 method for class 'CA'
plot(x, labelcol = "black", vectorcol = "red", ...)
```

**Arguments**

x                    an object of class `CA`  
 labelcol            colour of the sample labels (may be a vector).  
 vectorcol          colour of the vector loadings for the variables  
 ...                 optional arguments of the generic `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**

|                       |                                                                                                                                                                                                                                          |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>x</code>        | an object of class <code>compositional</code>                                                                                                                                                                                            |
| <code>sname</code>    | the sample name                                                                                                                                                                                                                          |
| <code>annotate</code> | a boolean flag controlling if the pies of the pie-chart should be labelled                                                                                                                                                               |
| <code>colmap</code>   | an optional string with the name of one of R's built-in colour palettes (e.g., <code>heat.colors</code> , <code>terrain.colors</code> , <code>topo.colors</code> , <code>cm.colors</code> ), which are to be used for plotting the data. |
| <code>...</code>      | optional parameters to be passed on to the graphics object                                                                                                                                                                               |

**Examples**

```
data(Namib)
plot(Namib$Major, 'N1', 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 labelled                                                                                                                 |
| 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 | <i>Plot a Procrustes configuration</i> |
|----------|----------------------------------------|

---

## Description

Plots the group configuration of a Generalised Procrustes Analysis

## Usage

```
## 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

```
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**

```

## 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**

```

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

```
## 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

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

---

plot.MDS *Plot an MDS configuration*

---

### 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**

```
## S3 method for class 'MDS'
plot(
  x,
  nlines = 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                                                                                                                             |
| nlines | 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.                                            |
| mar    | A numerical vector of the form c(bottom, left, top, right) that gives the number of lines of margin to be specified on the four sides of the plot. |
| mgp    | The margin line (in mex units) for the axis title, axis labels and axis line. See ?par for further details.                                        |
| xpd    | A logical value or NA. See ?par for further details.                                                                                               |
| ...    | optional arguments to the generic plot function                                                                                                    |

**See Also**

MDS

**Examples**

```
data(Namib)
mds <- MDS(Namib$DZ)
coast <- c('N1', 'N2', 'N3', 'N10', 'N11', 'N12', 'T8', 'T13')
snames <- names(Namib$DZ)
bgcol <- rep('yellow', length(snames))
```

```
bgcol[which(snames %in% coast)] <- 'red'
plot(mds,pch=21,bg=bgcol)
```

---

|                 |                                               |
|-----------------|-----------------------------------------------|
| plot.minsorting | <i>Plot inferred grain size distributions</i> |
|-----------------|-----------------------------------------------|

---

### Description

Plot the grain size distributions of the different minerals under consideration

### Usage

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

### Arguments

|            |                                                                                                                             |
|------------|-----------------------------------------------------------------------------------------------------------------------------|
| x          | an object of class minsorting                                                                                               |
| 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

minsorting

### Examples

```
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'))
```

---

plot.PCA                      *Compositional biplot*

---

### Description

Plot the results of a principal components analysis as a biplot

### Usage

```
## S3 method for class 'PCA'
plot(
  x,
  labelcol = "black",
  vectorcol = "red",
  choices = 1L:2L,
  scale = 1,
  pc.biplot = FALSE,
  ...
)
```

### Arguments

|           |                                                    |
|-----------|----------------------------------------------------|
| x         | an object of class PCA                             |
| labelcol  | colour(s) of the sample labels (may be a vector).  |
| vectorcol | colour of the vector loadings for the variables    |
| choices   | see the help pages of the generic biplot function. |
| scale     | see the help pages of the generic biplot function. |
| pc.biplot | see the help pages of the generic biplot function. |
| ...       | optional arguments of the generic biplot function  |

### See Also

PCA

### Examples

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

---

plot.ternary                      *Plot a ternary diagram*

---

### Description

Plots triplets of compositional data on a ternary diagram

### Usage

```
## 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**

```
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)
```

---

points.ternary      *Ternary point plotting*

---

**Description**

Add points to an existing ternary diagram

**Usage**

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

---

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`, `counts`, `compositional` and `varietal` OR a single object of class `varietal`.

**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**

Gower, J.C. (1975). Generalized Procrustes analysis, *Psychometrika*, 40, 33-50.

**See Also**

`GPA`

**Examples**

```
data(Namib)
gpa1 <- provenance(Namib$DZ, Namib$HM)
plot(gpa1)
```

```
data(SNSM)
gpa2 <- provenance(SNSM$ap)
plot(gpa2)
```

---

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 <https://www.ucl.ac.uk/~ucfbpve/provenance/>

`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 <https://www.ucl.ac.uk/~ucfbpve/provenance/> 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

Vermeesch, P., Resentini, A. and Garzanti, E., an R package for statistical provenance analysis, *Sedimentary Geology*, doi:10.1016/j.sedgeo.2016.01.009.

Vermeesch, P., Resentini, A. and Garzanti, E., 2016, An R package for statistical provenance analysis, *Sedimentary Geology*, 336, 14-25.

## See Also

Useful links:

- <https://www.ucl.ac.uk/~ucfbpve/provenance/>

---

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  $\{t_1, \dots, t_i, \dots, t_n\}$  and uncertainties  $\{s[t_1], \dots, s[t_i], \dots, s[t_n]\}$ . 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  $y_i = (z_i - z_o)/s[z_i]$ , where  $z_o$  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**

- Galbraith, R.F., 1988. Graphical display of estimates having differing standard errors. *Technometrics*, 30(3), pp.271-281.
- Galbraith, R.F., 1990. The radial plot: graphical assessment of spread in ages. *International Journal of Radiation Applications and Instrumentation. Part D. Nuclear Tracks and Radiation Measurements*, 17(3), pp.207-214.
- Galbraith, R.F. and Laslett, G.M., 1993. Statistical models for mixed fission track ages. *Nuclear Tracks and Radiation Measurements*, 21(4), pp.459-470.

**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",
  sep = ",",
  dec = ".",
  row.names = 1,
  header = TRUE,
  check.names = FALSE,
  ...
)
```

**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.                                              |
| sep         | the field separator character. Values on each line of the file are separated by this character.                                                                                                                               |
| dec         | the character used in the file for decimal points.                                                                                                                                                                            |
| row.names   | a vector of row names. This can be a vector giving the actual row names, or a single number giving the column of the which contains the row names, or character string the name of the table column containing the row names. |
| header      | a logical value indicating whether the file contains the names of the variables as its first line.                                                                                                                            |
| check.names | logical. If TRUE then the names of the variables in the frame are checked to ensure that they are syntactically variable names.                                                                                               |
| ...         | optional arguments to the built-in read.table 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

`name`: the name of the data object, extracted from the file path

**Examples**

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

---

read.counts

*Read a .csv file with point-counting data*

---

**Description**

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

**Usage**

```
read.counts(
  fname,
  method = "chisq",
  colmap = "rainbow",
  sep = ",",
  dec = ".",
  row.names = 1,
  header = TRUE,
  check.names = FALSE,
  ...
)
```

**Arguments**

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

|             |                                                                                                                                                                                                                               |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| sep         | the field separator character. Values on each line of the file are separated by this character.                                                                                                                               |
| dec         | the character used in the file for decimal points.                                                                                                                                                                            |
| row.names   | a vector of row names. This can be a vector giving the actual row names, or a single number giving the column of the which contains the row names, or character string the name of the table column containing the row names. |
| header      | a logical value indicating whether the file contains the names of the variables as its first line.                                                                                                                            |
| check.names | logical. If TRUE then the names of the variables in the frame are checked to ensure that they are syntactically variable names.                                                                                               |
| ...         | optional arguments to the built-in read.table 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

name: the name of the data object, extracted from the file path

**Examples**

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

---

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

---

**Description**

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

**Usage**

```
read.densities(
  fname,
  sep = ",",
  dec = ".",
  header = TRUE,
  check.names = FALSE,
  ...
)
```

**Arguments**

|             |                                                                                                                                 |
|-------------|---------------------------------------------------------------------------------------------------------------------------------|
| fname       | a string with the path to the .csv file                                                                                         |
| sep         | the field separator character. Values on each line of the file are separated by this character.                                 |
| dec         | the character used in the file for decimal points.                                                                              |
| header      | a logical value indicating whether the file contains the names of the variables as its first line.                              |
| check.names | logical. If TRUE then the names of the variables in the frame are checked to ensure that they are syntactically variable names. |
| ...         | optional arguments to the built-in read.table function                                                                          |

**Value**

a vector with mineral and rock densities

**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)
```

---

read.distributional    *Read a .csv file with distributional data*

---

**Description**

Reads a data table containing distributional data, i.e. lists of continuous data such as detrital zircon U-Pb ages.

**Usage**

```
read.distributional(
  fname,
  errorfile = NA,
  method = "KS",
  xlab = "age [Ma]",
  colmap = "rainbow",
  sep = ",",
  dec = ".",
  header = TRUE,
  check.names = FALSE,
  ...
)
```

**Arguments**

|                          |                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|--------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>fname</code>       | the path of a .csv file with the input data, arranged in columns.                                                                                                                                                                                                                                                                                                                                                                                                        |
| <code>errfile</code>     | the (optional) path of a .csv file with the standard errors of the input data, arranged by column in the same order as <code>fname</code> . Must be specified if the data are to be compared with the Sircombe-Hazelton dissimilarity.                                                                                                                                                                                                                                   |
| <code>method</code>      | 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), "Kuiper" (for Kuiper) or "SH" (for Sircombe and Hazelton). If <code>method = "SH"</code> , then <code>errfile</code> should be specified. If <code>method = "SH"</code> and <code>errfile</code> is unspecified, then the program will default back to the Kolmogorov-Smirnov dissimilarity. |
| <code>xlab</code>        | an optional string specifying the nature and units of the data. This string is used to label kernel density estimates.                                                                                                                                                                                                                                                                                                                                                   |
| <code>colmap</code>      | an optional string with the name of one of R's built-in colour palettes (e.g., <code>heat.colors</code> , <code>terrain.colors</code> , <code>topo.colors</code> , <code>cm.colors</code> ), which are to be used for plotting the data.                                                                                                                                                                                                                                 |
| <code>sep</code>         | the field separator character. Values on each line of the file are separated by this character.                                                                                                                                                                                                                                                                                                                                                                          |
| <code>dec</code>         | the character used in the file for decimal points.                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <code>header</code>      | a logical value indicating whether the file contains the names of the variables as its first line.                                                                                                                                                                                                                                                                                                                                                                       |
| <code>check.names</code> | logical. If TRUE then the names of the variables in the frame are checked to ensure that they are syntactically variable names.                                                                                                                                                                                                                                                                                                                                          |
| <code>...</code>         | optional arguments to the built-in <code>read.csv</code> 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

`name`: the name of the data object, extracted from the file path

**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))
```

---

|               |                                            |
|---------------|--------------------------------------------|
| read.varietal | <i>Read a .csv file with varietal data</i> |
|---------------|--------------------------------------------|

---

### Description

Reads a data table containing compositional data (e.g. chemical concentrations) for multiple grains and multiple samples

### Usage

```
read.varietal(
  fname,
  snames = NULL,
  sep = ",",
  dec = ".",
  method = "KS",
  check.names = FALSE,
  row.names = 1,
  ...
)
```

### Arguments

|             |                                                                                                                                                                                                                                                                     |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| fname       | file name (character string)                                                                                                                                                                                                                                        |
| snames      | either a vector of sample names, an integer marking the length of the sample name prefix, or NULL. <code>read.varietal</code> assumes that the row names of the <code>.csv</code> file consist of character strings marking the sample names, followed by a number. |
| sep         | the field separator character. Values on each line of the file are separated by this character.                                                                                                                                                                     |
| dec         | the character used in the file for decimal points.                                                                                                                                                                                                                  |
| 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 "Kuiper" (for Kuiper)                                                                 |
| check.names | logical. If TRUE then the names of the variables in the frame are checked to ensure that they are syntactically variable names.                                                                                                                                     |
| row.names   | logical. See the documentation for the <code>read.table</code> function.                                                                                                                                                                                            |
| ...         | optional arguments to the built-in <code>read.csv</code> function                                                                                                                                                                                                   |

### Value

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

`x`: a compositional data table

`snames`: a vector of strings corresponding to the sample names

`name`: the name of the dataset, extracted from the file path



**Examples**

```

Afile <- system.file("apatite.csv",package="provenance")
Ap <- read.varietal(fn=Afile,snames=3)
plot(MDS(Ap))

```

---

restore

*Undo the effect of hydraulic sorting*


---

**Description**

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

**Usage**

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

**Arguments**

|        |                                               |
|--------|-----------------------------------------------|
| X      | an object of class <code>compositional</code> |
| dens   | a vector of rock and mineral densities        |
| target | the target density (in g/cm <sup>3</sup> )    |

**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**

Garzanti E, Ando, S and Vezzoli, G. "Settling equivalence of detrital minerals and grain-size dependence of sediment composition." *Earth and Planetary Science Letters* 273.1 (2008): 138-151.

**See Also**

`minsorting`

**Examples**

```

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,Plag="P",HM=HMcomp,Opq="opaques")
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

**Usage**

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

**Arguments**

|                    |                                                       |
|--------------------|-------------------------------------------------------|
| <code>x</code>     | an object of class <code>distributional</code>        |
| <code>i</code>     | index of the first sample                             |
| <code>j</code>     | index of the second sample                            |
| <code>c.con</code> | smoothing bandwidth of the kernel functional estimate |

**Value**

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

**Author(s)**

Keith Sircombe and Martin Hazelton

**References**

Sircombe, K. N., and M. L. Hazelton. "Comparison of detrital zircon age distributions by kernel functional estimation." *Sedimentary Geology* 171.1 (2004): 91-111.

**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)
```

---

|      |                              |
|------|------------------------------|
| SNSM | <i>varietal data example</i> |
|------|------------------------------|

---

**Description**

A list of varietal datasets including detrital zircon (zr), apatite (ap) and titanite (tit) compositions from the Sierra Nevada de Santa Marta, provided by L. Caracciolo (FAU Erlangen).

**Author(s)**

Luca Caracciolo, Diana Hatzenbuehler and David Chew.

**Examples**

```
data(SNSM)
plot(indscal(zircon=SNSM$zr,apatite=SNSM$ap,titanite=SNSM$tit))
```

---

|        |                                        |
|--------|----------------------------------------|
| subset | <i>Get a subset of provenance data</i> |
|--------|----------------------------------------|

---

**Description**

Return a subset of provenance data according to some specified indices

**Usage**

```
## 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, ...)

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

**Arguments**

|            |                                                                                                                              |
|------------|------------------------------------------------------------------------------------------------------------------------------|
| x          | an object of class <code>distributional</code> , <code>compositional</code> , <code>counts</code> or <code>varietal</code> . |
| subset     | logical expression indicating elements or rows to keep: missing values are taken as false.                                   |
| select     | a vector of sample names                                                                                                     |
| ...        | optional arguments for the generic subset function                                                                           |
| components | vector of categories (column names) to keep                                                                                  |

**Value**

an object of the same class as x

**See Also**

[amalgamate](#), [combine](#)

**Examples**

```
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**

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

**Arguments**

|                   |                                                                                                                                       |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------|
| ...               | a sequence of datasets of class <code>compositional</code> , <code>distributional</code> , <code>counts</code> or <code>KDEs</code> . |
| <code>ncol</code> | the number of columns                                                                                                                 |
| <code>pch</code>  | (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` or `counts` and histograms for those of class `distributional`.

**See Also**

[KDEs](#)

**Examples**

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

---

|         |                                     |
|---------|-------------------------------------|
| ternary | <i>Define a ternary composition</i> |
|---------|-------------------------------------|

---

**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

**See Also**

restore

**Examples**

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

---

|                 |                                   |
|-----------------|-----------------------------------|
| ternary.ellipse | <i>Ternary confidence ellipse</i> |
|-----------------|-----------------------------------|

---

**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$Major, 'Ca0', 'Na20', 'K20')
plot(tern)
ternary.ellipse(tern)
```

---

|              |                              |
|--------------|------------------------------|
| text.ternary | <i>Ternary text plotting</i> |
|--------------|------------------------------|

---

**Description**

Add text 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**

```
data(Namib)
tern <- ternary(Namib$Major, 'Ca0', 'Na20', 'K20')
plot(tern, pch=21, bg='red', labels=NULL)
# add the geometric mean composition as a text label:
gmean <- ternary(exp(colMeans(log(tern$x))))
text(gmean, labels='geometric mean')
```

---

varietal2distributional

*Convert varietal to distributional data*

---

**Description**

Convert an object of class `varietal` either to a list of distributional objects by breaking it up into separate elements, or to a single distributional object corresponding to the first principal component.

**Usage**

```
varietal2distributional(x, bycol = FALSE, plot = FALSE)
```

**Arguments**

x                    an object of class `varietal`.  
 bycol              logical. If TRUE, returns a list of distributional objects (one for each element). If FALSE, returns a single distributional object (containing the PC1 scores for each sample).  
 plot                logical. If TRUE, shows the PCA biplot that is used when `bycol` is FALSE.

**Examples**

```
Apfile <- system.file("apatite.csv", package="provenance")
Ap <- read.varietal(fn=Apfile, snames=3)
varietal2distributional(Ap, bycol=FALSE, plot=TRUE)
```

# Index

`_PACKAGE` (provenance), 40

ALR, 3

`amalgamate`, 4, 52

`as.acomp`, 5

`as.compositional`, 6

`as.counts`, 6

`as.data.frame`, 7

`as.varietal`, 8

`botev`, 9

`bray.diss`, 9

CA, 10

`central`, 11

CLR, 12

`combine`, 13, 52

`densities`, 13

`diss`, 14

`endmembers`, 15

`get.f`, 16

`get.n`, 16

`get.p`, 17

GPA, 18

`indscal`, 18

KDE, 20

KDEs, 21, 52

`KS.diss`, 22

`Kuiper.diss`, 23

`lines` (`points.ternary`), 39

`lines.ternary`, 23

MDS, 24

`minsorting`, 25

Namib, 27

PCA, 28

`plot.CA`, 28

`plot.compositional`, 29

`plot.distributional`, 30

`plot.GPA`, 31

`plot.INDSCAL`, 32

`plot.KDE`, 33

`plot.KDEs`, 34

`plot.MDS`, 34

`plot.minsorting`, 36

`plot.PCA`, 37

`plot.ternary`, 38

`points.ternary`, 39

`procrustes`, 39

provenance, 40

`provenance-package` (provenance), 40

`radialplot`, 41

`read.compositional`, 43

`read.counts`, 44

`read.densities`, 45

`read.distributional`, 46

`read.varietal`, 48

`restore`, 49

`SH.diss`, 50

SNSM, 51

`subset`, 51

`summaryplot`, 52

ternary, 53

`ternary.ellipse`, 54

`text` (`points.ternary`), 39

`text.ternary`, 54

`varietal2distributional`, 55