Package ‘unfoldr’

May 23, 2019

Title Stereological Unfolding for Spheroidal Particles

Version 0.7

Date 2019-05-22

Author Markus Baaske [aut, cre],
Felix Ballani [ctb]

Maintainer Markus Baaske <mafr2007@gmail.com>

Description Stereological unfolding of the joint size-shape-orientation distribution of spheroidal shaped particles.

Depends R (>= 3.3.0)

Suggests rgl, plotrix

License GPL (>= 3)

Repository/R-Forge/Project unfoldr

Repository CRAN

LazyData no

RoxygenNote 6.1.1

Repository/R-Forge/Revision 238

Repository/R-Forge/DateTimeStamp 2019-05-22 20:28:49

Date/Publication 2019-05-23 08:40:03 UTC

NeedsCompilation yes

R topics documented:

`unfoldr-package` *Stereological Unfolding for Spheroidal Particles*

Description

Stereological unfolding as implemented in this package consists of the estimation of the joint size-shape-orientation distribution of spheroidal shaped particles based on the same measured quantities of corresponding vertical section profiles. A single trivariate discretized version of the (stereological) integral equation in the case of prolate and oblate spheroids is solved numerically by a variant of the well-known Expectation Maximization (EM) algorithm. In addition, routines for estimating the empirical diameter distribution of spheres from planar sections (better known as the Wicksell's corpuscle problem [3]) is also implemented. The package also provides functions for the simulation of Poisson germ-grain processes with either spheroids, spherocylinders or spheres as grains including functions for planar and vertical sections and digitization of section profiles.

References

1. Beneš, V. and Rataj, J. Stochastic Geometry: Selected Topics Kluwer Academic Publishers, Boston, 2004
2. Ohser, J. and Schladitz, K. 3D images of materials structures Wiley-VCH, 2009
3. Ohser, J. and Muecklich, F. Statistical analysis of microstructures in materials science J. Wiley & Sons, 2000
4. C. Lantuéjoul. Geostatistical simulation. Models and algorithms. Springer, Berlin, 2002. Zbl 0990.86007
5. Müller, A., Weidner, A., and Biermann, H. (2015). Influence of reinforcement geometry on the very high-cycle fatigue behavior of aluminum-matrix-composites. Materials Science Forum, 825/826:150-157

`binning1d` *Binning numeric values*

Description

Vector of count data

Usage

```
binning1d(x, bin, na.rm = FALSE)
```

Arguments

`x`	numeric values to be binned
`bin`	non-decreasingly sorted breaks vector
`na.rm`	logical, default `FALSE`, whether to remove missing values, including `NaN` in `x`

Details

The function provides basic binning (grouping) of numeric values into classes defined by the breaks vector `bin`. The values are binned according to $bin[i[j]] < x[j] \leq bin[i[j]+1]$ for intervals $i = 1, ..., N-1$ and `length(bin)=N` of values $x[j], j = 1, ..., |x|$. If $x[j] > bin[N]$ or $x[j] < bin[1]$ then $x[j]$ is not counted at all.

Value

Vector of count data

Author(s)

M. Baaske

Examples

```
x <- runif(100,0,1)
bin <- seq(0,1,by=0.1)
binning1d(x,bin)
```

`binning3d` *Histogram data*

Description

Count data of size, shape and orientation

Usage

```
binning3d(size, angle, shape, breaks, check = TRUE, na.rm = TRUE)
```

Arguments

`size`	vector of sizes
`angle`	vector of angles
`shape`	vector of shape factors
`breaks`	list of bin vectors
`check`	logical, default is `TRUE`
`na.rm`	logical, whether `NA`s should be removed, default `TRUE`

Details

For each value of planar or spatial measured quantities `size`, `shape` and `orientation` the function counts the number of observations falling into each class (bin). The list `breaks` can be obtained by the function `setbreaks`. If `check=TRUE`, some checks on `breaks` are done. For an example please see the file 'almmc.R'.

Value

3D array of binned count data

Author(s)

M. Baaske

`coefficientMatrixSpheres` *Coefficients for Expectation Maximization algorithm*

Description

Matrix of coefficients for Wicksell's corpuscle problem

Usage

```
coefficientMatrixSpheres(bin)
```

Arguments

`bin`	non-decreasing vector of class limits

Details

The function computes the matrix of coefficients for solving the discretized integral equation of the Wicksell's corpuscle problem.

Value

array of coefficients

Author(s)

M. Baaske

References

Ohser, J. and Muecklich, F. Statistical analysis of microstructures in materials science J. Wiley & Sons, 2000

`coefficientMatrixSpheroids` *Coefficients for trivariate unfolding*

Description

Compute coefficients of the discretized integral equation for unfolding

Usage

```
coefficientMatrixSpheroids(breaks, stype = c("prolate", "oblate"),
  check = TRUE, nCores = getOption("par.unfoldr", 1))
```

Arguments

`breaks`	list of bin vectors, see `setbreaks`
`stype`	type of spheroid, either "`prolate`" or "`oblate`"
`check`	logical, whether to run some input checks
`nCores`	number of cpu cores used to compute the coefficients

Details

In order to apply the Expectation Maximization (EM) algorithm to the stereological unfolding procedure for the joint size-shape-orientation distribution in 3D one first has to compute the coefficients of a discretized integral equation. This step is the most time consuming part of unfolding and therefore has been separated in its own function and can be called separately if needed. The number of bin classes for the size, shape and orientation do not need to be the same, but the given class limits are also used for binning the spatial values. One can define the number of cpu cores by the global option `par.unfoldr` or passing the number of cores `nCores` directly to the function.

Value

numeric 6D array of coefficients

Author(s)

M. Baaske

Examples

```
## Not run:
## Comment: Set breaks vector and compute the coefficient matrix
## for spheroid unfolding

options(par.unfoldr=2L)
breaks <- setbreaks(c(6,5,6),maxSize=0.37,kap=1.25)
breaks

P <- coefficientMatrixSpheroids(breaks,check=FALSE)
c(min(P),max(P),sum(P))


## End(Not run)
```

`cylinders3d` *Cylinder system 3D*

Description

Draw spherocylinders in 3D

Usage

```
cylinders3d(S, box, draw.axes = FALSE, draw.box = TRUE,
  clipping = FALSE, ...)
```

Arguments

`S`	list of cylinders, see `simPoissonSystem`
`box`	simulation box
`draw.axes`	logical, whether to show the axes
`draw.box`	logical, whether to show the bounding box
`clipping`	logical, whether to clip all lateral planes of the simulation box
`...`	further material properties passed to 3d plotting functions

Details

The function requires the package '`rgl`' to draw spherocylinders into a 3D '`rgl`' image. For an example please see the file 'simCylinders.R'.

Author(s)

M. Baaske

`data15p` *Intersection ellipses parameters*

Description

The data set consists of section profile parameters (assumed to come from prolate spheroids) of fitted ellipses based on measured section particles of an aluminium matrix composite [5] from metallographic analysis. The data set can be used to reconstruct the trivariate spatial (prolate) spheroid distribution.

Usage

```
data(data15p)
```

Format

A matrix of columns named `A` (major semi-axis length), `C` (minor semi-axis length), `S=C/A` (shape factor), `alpha` (polar angle in the intersecting plane) and coordinates `(x,y)` of the centers of fitted ellipses.

Author(s)

M. Baaske

digitizeProfiles *Digitization*

Description

Digitize 2D section profiles

Usage

```
digitizeProfiles(sp, delta = 0.01, win = NULL)
```

Arguments

`sp`	list of section profiles, see `intersectSystem`
`delta`	lattice constant for discretization of section profiles
`win`	list of length two, the intersection window, default `NULL`

Details

A list of section profiles, e.g. discs, ellipses or segments from intersected spherocylinders, is digitized according to a given resolution according to the lattice constant `delta` such that the result is an integer matrix which can be interpretated as a binary image. An intersection window can be either provided by the user w.r.t. $[l, u]^2$ where `l,u` are lower, respectively, upper bounds of corresponding to a simulation box, or, it is taken from the section profiles `sp` which stores the intersection window as an attribute.

Value

binary (integer) matrix as an image

Author(s)

M. Baaske

Examples

```
# simulation box
box <- list("xrange"=c(0,5),"yrange"=c(0,5),"zrange"=c(0,5))
# (exact) bivariate size-shape (isotropic) orientation distribution (spheroids)
theta <- list("size"=list("mx"=-2.5,"my"=0.5, "sdx"=0.35,"sdy"=0.25,"rho"=0.15),
              "orientation"=list("kappa"=1))

# return only 3D system
S <- simPoissonSystem(theta,lam=100,size="rbinorm",box=box,type="prolate",
      intersect="original",n=c(0,1,0),mu=c(0,0,1),perfect=TRUE,pl=1)

# vertical intersection w.r.t. 'mu' (z axis, see above)
sp <- intersectSystem(S, 2.5)
```

```
# show intersecting window
win <- attr(sp,"win")

# digitize (could also pass some 'win' as an argument)
W <- digitizeProfiles(sp, delta=0.01, win = NULL)
image(1:nrow(W),1:ncol(W),W,col=gray(1:0))
```

`drawSpheroidIntersection` *Spheroid intersections 3D*

Description

Draw section profiles of spheroids in 3D

Usage

```
drawSpheroidIntersection(E, n = c(0, 1, 0), np = 25)
```

Arguments

`E`	a list of spheroid intersections, see `intersectSystem`
`n`	the normal vector of the intersecting plane
`np`	number of points for a polygon approximation of the ellipses

Details

The function requires the package ’`rgl`’ for drawing section profiles (ellipses) into a 3D ’`rgl`’ image. For a full example please see the file ’simSpheroids.R’.

Author(s)

M. Baaske

`em.saltykov` *Expectation Maximization (EM) algorithm*

Description

Estimation of the empirical sphere diameter distribution

Usage

```
em.saltykov(y, bin, maxIt = 32)
```

Arguments

`y`	vector of observed absolute frequencies of circle diameters
`bin`	non-decreasing vector of class limits
`maxIt`	maximum number of iterations to be used

Details

The function performs the EM algorithm, see reference below.

Value

vector of count data of absolute frequenties of sphere diameters

Author(s)

M. Baaske

References

Ohser, J. and Muecklich, F. Statistical analysis of microstructures in materials science J. Wiley & Sons, 2000

Examples

```
## Not run:
## Comment: Simulate a Poisson sphere system,
##  intersect, discretize and display results

library(unfoldr)
library(rgl)
library(plotrix)

spheres <- function(spheres, box=NULL, draw.box=FALSE, draw.axis=FALSE, ...) {
xyz <- apply(sapply(spheres, "[[", "center"),1,function(x) x)
sizes <- unlist(lapply(spheres,function(x) x$r))
spheres3d(xyz,radius=sizes,...)
```

```
if(draw.axis) {
axes3d(c('x','y','z'), pos=c(1,1,1), tick=FALSE)
#title3d('','','x','y','z')
}
axes3d(edges = "bbox",labels=TRUE,tick=FALSE,box=TRUE,nticks=0,
expand=1.0,xlen=0,xunit=0,ylen=0,yunit=0,zlen=0,zunit=0)

if(draw.box) {
x <- box$xrange[2]
y <- box$yrange[2]
z <- box$zrange[2]
c3d.origin <- rgl::translate3d(rgl::scale3d(rgl::cube3d(col="gray", alpha=0.1),
x/2,y/2,z/2),x/2,y/2,z/2)
rgl::shade3d(c3d.origin)
}

}

col <- c("#0000FF","#00FF00","#FF0000","#FF00FF","#FFFF00","#00FFFF")

##################################################################
## `beta` distribution for radii
##################################################################

lam <- 50
## parameter beta distribution (radii)
theta <- list("size"=list("shape1"=1,"shape2"=10))

# simulation bounding box
box <- list("xrange"=c(-1,4),"yrange"=c(-1.5,3.5),"zrange"=c(0,2))

## simulate and return full spheres system
## intersect with XZ plane and return full list of intersection profiles

## section plane xy
S <- simPoissonSystem(theta,lam,size="rbeta",box=box,type="spheres",
intersect="full",n=c(0,0,1),dz=0,pl=1)

# check resulting distribution
length(S$S)
summary(sapply(S$S,"[[","r"))
theta$size[[1]]/(theta$size[[1]]+theta$size[[2]]) # mean

## interior spheres:
## the ones which intersect one of the lateral planes (without top/bottom planes)
## showing spheres with color intersect
notIn <- sapply(S$S,function(x) !attr(x,"interior"))
spheres(S$S[notIn],box,FALSE,TRUE,color=col)

# not intersecting
In <- sapply(S$S,function(x) attr(x,"interior"))
spheres(S$S[In],box,color="gray")
```

```
## full sphere system
# open3d()
# spheres(S$S,box,FALSE,TRUE,color=col)
# planes3d(0,0,1,0,col="darkgray",alpha=1)

## draw intersections
sp <- S$sp
id <- sapply(sp,"[[","id")
open3d()
spheres(S$S[id],box,FALSE,TRUE,color=col)
planes3d(0,0,1,0,col="darkgray",alpha=1)

XYr <- t(sapply(sp,function(s) cbind(s$center[1],s$center[2],s$r)))
# centers
x <- XYr[,1]
y <- XYr[,2]
r <- XYr[,3]
win <- attr(sp,"win")
xlim <- win[[1]]
ylim <- win[[2]]

dev.new()
plot(x,y,type="n",xaxs="i", yaxs="i", xlab="x",ylab="y",xlim=xlim,ylim=ylim)
for(i in 1:nrow(XYr))
 draw.circle(x[i],y[i],r[i],nv=100,border=NULL,col="black")

## digitize inersections
## `win` can also be omitted if not different
## from original itersection window (derived from the box)
W <- digitizeProfiles(sp, delta=0.01, win=win)
dim(W)
dev.new()
image(1:nrow(W),1:ncol(W),W,col=gray(1:0))

#################################################################
## Exact simulation of spheres with log normal radii
#################################################################

lam <- 100
## parameter rlnorm distribution (radii)
theta <- list("size"=list("meanlog"=-2.5,"sdlog"=0.5))
# simulation bounding box
box <- list("xrange"=c(0,5),"yrange"=c(0,5),"zrange"=c(0,5))
## simulate only 3D system
S <- simPoissonSystem(theta,lam,size="rlnorm",box=box,type="spheres",
intersect="original", perfect=TRUE, pl=101)
## show
open3d()
spheres(S[1:2000],box,TRUE,TRUE,color=col)

## check!
mean(log(sapply(S,"[[","r")))
```

```
sd(log(sapply(S,"[[","r")))

##################################################################
## Planar section
##################################################################

# planar section of exact simulated `rlnorm` sphere system:
# returns diameters for those section profiles
# which have their centers inside the intersection window
sp <- planarSection(S,d=2.5,intern=TRUE,pl=1)
hist(sp)
summary(sp)
mean(log(sp/2))
sd(log(sp/2))

##################################################################
## General intersection, all objects (inter=FALSE)
##################################################################

SP <- intersectSystem(S, 2.5, n=c(0,0,1), intern=FALSE, pl=1)

## show in 3D
id <- sapply(SP,"[[","id")
open3d()
spheres(S[id],box,TRUE,color=col)
planes3d(0,0,-1,2.5,col="black",alpha=1)

## 2D sections
XYr <- t(sapply(SP,function(s) cbind(s$center[1],s$center[2],s$r)))
# centers
x <- XYr[,1]
y <- XYr[,2]
r <- XYr[,3]
xlim <- box$xrange
ylim <- box$yrange

dev.new()
plot(x,y,type="n",xaxs="i", yaxs="i", xlab="x",ylab="y",xlim=xlim,ylim=ylim)
for(i in 1:nrow(XYr))
draw.circle(x[i],y[i],r[i],nv=100,border=NULL,col="black")

# digitize
W <- digitizeProfiles(SP, delta=0.01)
dev.new()
image(1:nrow(W),1:ncol(W),W,col=gray(1:0))


##################################################################
## Unfolding
##################################################################

ret <- unfold(sp,nclass=25)
```

```
## Point process intensity
cat("Intensities: ", sum(ret$N_V)/25, "vs.",lam,"\n")

## original diameters
r3d <- unlist(lapply(S,function(x) 2*x$r))
rest3d <- unlist(lapply(2:(length(ret$breaks)),
function(i) rep(ret$breaks[i],sum(ret$N_V[i-1]))))

op <- par(mfrow = c(1, 2))
hist(r3d[r3d<=max(ret$breaks)], breaks=ret$breaks, main="Radius 3D",
freq=FALSE, col="gray",xlab="r")
hist(rest3d, breaks=ret$breaks,main="Radius estimated",
freq=FALSE, col="gray", xlab="r")
par(op)


#################################################################
## Update intersection: find objects which intersect bounding box
#################################################################

idx <- updateIntersections(S)
sum(!idx) # objects intersecting
id <- which( idx != 1)
open3d()
spheres(S[id],box,TRUE,TRUE,color=col)

#################################################################
## user-defined simulation function
#################################################################

lam <- 50
theta <- list("p1"=-2.5,"p2"=0.5)
myfun <- function(p1,p2) { c("r"=rlnorm(1,p1,p2)) }

box <- list("xrange"=c(0,5),"yrange"=c(0,5),"zrange"=c(0,5))

S <- simPoissonSystem(theta,lam,rjoint=myfun,box=box,type="spheres",pl=1)
r <- sapply(S,"[[","r")
mean(log(r))
sd(log(r))

open3d()
spheres(S,box,TRUE,TRUE,color=col)

## End(Not run)
```

em.spheroids *Trivariate stereological unfolding*

Description

Estimate the joint size-shape-orientation distribution of spheroids

Usage

```
em.spheroids(P, F, maxIt, nCores = getOption("par.unfoldr", 2L))
```

Arguments

`P`	coefficient array
`F`	input histogram
`maxIt`	maximum number of EM iterations
`nCores`	number of cpu cores to be used

Details

Given an array of coefficients `P`, see `coefficientMatrixSpheroids` and an input histogram `F` of measured planar characteristics of section profiles, the function estimates the spatial joint size-shape-orientation distribution of the corresponding spheroids in 3D by a discretized version of the *Expectation Maximization* (EM) algorithm. A number of cpu cores can be set by the option '`par.unfoldr`' for parallel computations. The function is also internally called by `unfold` in case of spheroids.

Value

trivariate histogram

Author(s)

M. Baaske

References

Beneš, V. and Rataj, J. Stochastic Geometry: Selected Topics Kluwer Academic Publishers, Boston, 2004

Examples

```
## Comment: Trivariate unfolding of spheroid distribution

## set number of cpu cores (optional)
options(par.unfoldr=2L)

## Intensity: mean number of spheroids per unit volume
lam <- 1000

## simulation parameters
theta <- list("size"=list("meanlog"=-2.5,"sdlog"=0.5),
      "shape"=list(0.5),"orientation"=list("kappa"=2))
```

```
## simualtion
set.seed(1234)

S <- simPoissonSystem(theta,lam,size="rlnorm",
orientation="rbetaiso",box=list(c(0,5)),type="prolate",pl=1)

## unfolding
sp <- verticalSection(S,2.5)
ret <- unfold(sp,c(7,6,5),kap=1.25)
cat("Intensities: ", sum(ret$N_V)/25, "vs.",lam,"\n")
```

`intersectSystem` *Intersection in 3D*

Description

Intersect a system of spheres, spheroids or spherocylinders by a plane

Usage

```
intersectSystem(S, d, n = c(0, 1, 0), intern = FALSE, pl = 0)
```

Arguments

`S`	list of spheres, spheroids or spherocylinders, see `simPoissonSystem`
`d`	distance of the the box-aligned intersecting plane from the origin
`n`	normal vector which defines the intersecting plane
`intern`	logical, `FALSE` (default), return all section profiles otherwise only those which have their centers inside the corresponding intersection window (if the intersected system had been simulated using exact simulation this makes sense)
`pl`	integer, `pl=0` (default), for no verbose output and otherwise a full specification list of section profiles in case of sphere and spheroid intersections

Details

The function intersects a given (Poisson) system made of spheres, spheroids or cylinders as grains by a plane defined by a normal vector, e.g. `n=c(0,1,0)`, perpendicular to one of the bounding planes of the simulation box. For a print level `pl>=0` some verbose output is given. Also it sets the type of return value. In case of spheroid intersections, setting `pl=10`, leads to a short version of the full specification return list of section profiles with elements named `A` (major semi-axis), `C` (minor semi-axis), `S` (the shape factor as the ratio of these two) and `phi` as the angle in the intersecting plane between $[0, 2\pi]$ w.r.t. the 'x' axis. Otherwise additional components are returned such as the ellipse‘s rotation matrix, also named `A`, the center point `center` and a constant `type=10` (defining the object of full ellipses among other types of intersection objects) of the section profiles. For sphere intersections only a numeric vector of disc radii are returned as a short version of return values.

Value

For spheroid intersections the function returns a list of size, shape and angle of section profiles or a short version of it; for sphere intersections either radii or lists containing the centers of discs and the object number.

Author(s)

M. Baaske

Examples

```
box <- list("xrange"=c(0,5),"yrange"=c(0,5),"zrange"=c(0,5))

# constant size-shape orientation distribution (spheroids)
theta <- list("size"=list(0.1),"shape"=list(0.5), "orientation"=list("kappa"=10))

S <- simPoissonSystem(theta,lam=100,box=box,type="prolate",pl=1)

# return short version of section profiles
sp <- intersectSystem(S, 2.5, pl=10)
```

---

parameterEstimates *Spatial histogram*

Description

Characteristics for a spatial histogram

Usage

```
parameterEstimates(H, breaks)
```

Arguments

`H`	trivariate (output) histogram, estimated by `unfold`
`breaks`	breaks vector from `setbreaks`

Details

Based on the estimated joint distribution after unfolding the function replicates the entries of the vector `breaks`, see `setbreaks`, equal to the number of estimated count data for each class. For an example please see the file 'unfold.R'.

Value

list of the following entries: either major or minor semi-axis length `a`, (polar) angle `Theta` and shape factor `s`, see `parameters3d`

Author(s)

M. Baaske

---

parameters3d *3D characteristics of spheroids*

Description

Extract size, shape and orientation angle

Usage

```
parameters3d(S)
```

Arguments

`S`	list of spheroids

Details

The function extracts the characteristics of a 3D spheroid system, either oblate or prolate spheroids, and returns a list of the following elements: major semi-axis length '`a`', shape factor '`s`' and polar angle '`Theta`'. For a full example please see the file 'unfold.R'.

Value

a list of spheroids‘ 3D characteristics

Author(s)

M. Baaske

---

planarSection *Sphere planar sections*

Description

Planar intersection of spheres

Usage

```
planarSection(S, d, intern = FALSE, pl = 0)
```

Arguments

`S`	list of spheres of class `sphere`, see `simPoissonSystem`
`d`	distance of the (planar) xy-plane to the origin
`intern`	logical, `FALSE` (default), whether to return all discs or only those which have their centers inside the intersecting window
`pl`	print level, `pl>0` for some verbose output

Details

The function computes the planar intersection of a sphere system, i.e. an intersection with the plane whose normal vector is given by `c(0,0,1)` and returns the diameters of the resulting discs.

Value

numeric vector of disc diameters

Author(s)

M. Baaske

Examples

```
lam <- 100

# parameter rlnorm distribution (radii)
theta <- list("size"=list("meanlog"=-2.5,"sdlog"=0.5))

# simulation bounding box
box <- list("xrange"=c(0,5),"yrange"=c(0,5),"zrange"=c(0,5))

# simulate only 3D system
S <- simPoissonSystem(theta,lam,size="rlnorm",box=box,type="spheres",
  intersect="original", pl=1)

# return only objects whose centers are within
# the intersection window
sp <- planarSection(S,d=2.5,intern=TRUE,pl=1)

# histogram of diameters
hist(sp)
summary(sp)

# distribution of radii
mean(log(sp/2))
sd(log(sp/2))
```

`sectionProfiles` *Construct section profiles*

Description

Set up section profiles of spheroids for unfolding

Usage

```
sectionProfiles(size, alpha, type = c("prolate", "oblate"))
```

Arguments

`size`	matrix of lengths of the semi-axes
`alpha`	angle of section profiles in the plane (see details)
`type`	name of the spheroid type, either "`prolate`" or "`oblate`" from which the section profiles are assumed to come from

Details

The function aggregates the necessary information for trivariate unfolding of spheroids' joint size-shape orientation distribution of type either "`prolate`" or "`oblate`". The argument `size` is a numeric matrix of semi-axis lengths where the first column corresponds to the major semi-axis and the second one to minor semi-axis. The orientation of an ellipse is assumed to be measured as the angle between its major axis and vertical axis of the coordinate system in the intersection plane ('z' axis in 3D). For values in $[0, 2\pi]$ these angles are automatically transformed to $[0, \pi/2]$ as required by the unfolding procedure.

Value

The function returns a list which consists of either the longer or shorter semi-axis length named `A` of section profiles corresponding to the type of spheroids used before and whose joint joint distribution is to be estimated (by unfolding), the shape factor `S` of both semi-axes as the shape factor between $(0, 1]$ and the orientation angle `alpha`, either of class "`prolate`" or "`oblate`".

Author(s)

M. Baaske

Examples

```
# load data set
data(data15p)

# matrix of semi-axes lengths (major,minor)
AC <- data.matrix(data15p[c("A","C")])/1000

# selecting the minor semi-axis for prolate type of spheroids:
```

```
# independent of nomenclature (always named \code{A})
sp <- sectionProfiles(AC,unlist(data15p["alpha"]))

summary(sp$A) # here minor semi-axis because of prolate
summary(sp$S) # shape factor
summary(sp$alpha) # angle assumed to be w.r.t. (vertical) 'z' axis
```

`setbreaks` *Break vectors*

Description

Construct class limits vectors

Usage

```
setbreaks(nclass, maxSize, base = NULL, kap = 1,
  sizeType = c("linear", "exp"))
```

Arguments

`nclass`	number of classes
`maxSize`	maximum of `size` values
`base`	constant for size class construction
`kap`	constant for shape class construction
`sizeType`	either `linear` or `exp`, default is `linear`

Details

The function constructs the class limits for the size, shape and orientation parameters. One can either define "`linear`" class limits of the sizes as $a_i = i\delta, \delta = maxSize/M$ or using exponentially increasing limits like $base^i, i = 1, \dots, M$. The orientation classes are defined by $\theta_j = j\omega, \omega = \pi/(2N), j = 1, \dots, N$ in the range $[0, \pi/2]$, where M, N are the number of size classes and, respectively, the number of orientation classes. The argument `base` must not be `NULL` if `sizeType` equals "`exp`".

Value

list of class limits vectors

Author(s)

M. Baaske

Examples

```
setbreaks(c(8,5,7),0.935,base=0.5,kap=1.25,sizeType="exp")
```

simPoissonSystem *Poisson germ-grain process*

Description

Simulation of Poisson germ-grain processes with either spheres, spheroids or spherocylinders as grains

Usage

```
simPoissonSystem(theta, lam, size = "const", shape = "const",
  orientation = "rbetaiso", type = c("prolate", "oblate", "spheres",
  "cylinders"), rjoint = NULL, box = list(c(0, 1)), mu = c(0, 0, 1),
  dz = 0, n = c(0, 1, 0), intersect = c("original", "full", "only",
  "digi"), delta = 0.01, intern = FALSE, perfect = FALSE, pl = 0,
  label = "N")
```

Arguments

`theta`	list of simulation parameters which must consist of elements: `size`, `shape` and `orientation`
`lam`	mean number of objects per unit volume
`size`	name of the size distribution function
`shape`	name of the shape distribution function
`orientation`	name of direction distribution function
`type`	type of grain, either "`prolate`" or "`oblate`", "`spheres`", "`cylinders`"
`rjoint`	user-defined function, which specifies the (joint) distribution of the size, shape and orientation
`box`	simulation box
`mu`	main orientation axis, `mu=c(0,0,1)` (default)
`dz`	distance of the intersecting plane to the origin
`n`	normal vector defining the intersecting plane
`intersect`	options for type of return values: "`full`" for the simulated system together with section profiles as two lists named `S` and `sp` respectively, choose "`only`" for section profiles only, "`original`" for the 3D system only and "`digi`" for a (binary) integer matrix `W` as a discretized version of section profiles whose resolution depends on the chosen lattice constant `delta`
`delta`	lattice constant for discretization, set to `0.01` (default)
`intern`	logical, `FALSE` (default), whether to return only section profiles with centers inside the simulation window
`perfect`	logical, `FALSE` (default), whether to simulate exactly (also called perfect)
`pl`	integer, print level and return value type, see details
`label`	character, a label set to each generated object, set to '`N`' (default)

Details

The function can simulate a Poisson germ-grain process according to the parameter `theta` within a predefined (3D) box. The positions of the germs follow a uniform distribution according to a Poisson process with mean intensity parameter `lam`. The function either randomly generates `type="prolate"` or `type="oblate"` spheroids, spheres or spherocylinders. The argument `size` sets the name of the distribution function for the size/length of the objects, i.e. the major semi-axis lengths in case of spheroids, radii for spheres or the lengths of the main axis of rotation for spherocylinders including the end caps.

The following direction (orientation) distributions of the spheroids' major-axis, respectively, cylinders' main axis are available: a uniform distribution ("`runifdir`"), distribution ("`rbetaiso`") and the "*von Mises-Fisher*" ("`rvMisesFisher`") distribution. The two last ones depend on the concentration parameter `kappa` which is set as part of the parameter list `theta`, see examples below. The direction distributions generate random spherical coordinates (ϑ, ϕ) w.r.t. a fixed main orientation axis `mu` with polar angle $\vartheta \in [0, \pi/2)$ and azimuthal angle $\phi \in [0, 2\pi)$. The simulations are always performed within a bounding 3D box which consists of a list specifying the ranges of each dimension corresponding to the lower and upper limits of the box in each direction. If the argument `box` contains only a single range, i.e. `box=list(c(0,1))`, this limit isassumed for the remaining dimensions which is then simply replicated. The optional argument `rjoint` defines a (joint) distribution function which can be any function provided by the user in order to generate the required distributional parameters for the spheroids or spherocylinders. For an in-depth example of usage please see the workflow in 'simSpheroids.R' and 'simCylinders.R'.

In addition, the function supports an exact simulation type [2] of the grains. In case of spheroids and spherocylinders setting `size="rbinorm"` declares a bivariate size-shape distribution for which the exact simulation is available. More specifically, for a bivariate normal random vector $[X, Y]$ with correlation parameter ρ, the length of the major semi-axis of a spheroid is given by $a = exp(x)$ with a (logit-transformed) shape parameter as $s = 1/(1 + exp(-y))$ and thus a scaled minor semi-axis length $c = a * s$. The modification leads to a log-normally distributed length of the major semi-axis. Consequently, in case of spherocylinders, the log-normally distributed length is $len = h + 2 * r$ where `h` is the height and $r = len/2 * s$ the radius. The main direction `u` of a spheroid or spherocylinder is determined by the major axis independent of size and shape. Further, the following univariate distributions of the major semi-axis `a`, respectively, length `len` and shape `s` are available: '`rbeta`', '`rgamma`', '`rlnorm`' and '`runif`'. One can also use '`const`' for simulations with constant lengths or shapes. Note that only simulations with size distributions '`rbinorm`' or '`rlnorm`' can use the exact type of simulation.

For spheres any distribution of the radii can be specified as a name of a user-defined function in the argument `size` as long as the formal named function parameters match the actual names of the parameters exactly as defined in `theta`. Besides this, all other distributions given above are also available. Using '`const`' simulates spheres of constant radii.

The argument `pl>=0` denotes both the print level of intermediate output and by the same time the type of the return value. If `pl=10`, then an abbreviated list of spheroids or spheres is returned to speed up computation. Note that, the current implementation does not include routines for unfolding the joint size-shape-orientation distribution of spherocylinders so far.

Value

list of 3D objects depending on the chosen return type defined by the argument `intersect`

Author(s)

M. Baaske

References

- Ohser, J. and Schladitz, K. 3D images of materials structures Wiley-VCH, 2009
- C. Lantuéjoul. Geostatistical simulation. Models and algorithms. Springer, Berlin, 2002. Zbl 0990.86007

Examples

```
# intensity parameter
lam <- 100

# simulation bounding box
box <- list("xrange"=c(0,5),"yrange"=c(0,5),"zrange"=c(0,5))

# log normal size distribution with a constant shape factor and
# concentration parameter (\code{kappa=1}) for the orientation, see reference [1]
theta <- list("size"=list("meanlog"=-2.5,"sdlog"=0.5),
              "shape"=list("s"=0.5),
              "orientation"=list("kappa"=1))

S <- simPoissonSystem(theta,lam,size="rlnorm",box=box,type="oblate",pl=1)
length(S)
```

---

spheroids3d *Spheroid system 3D*

Description

Draw a spheroid system in 3D

Usage

```
spheroids3d(S, box, draw.axes = FALSE, draw.box = TRUE,
  draw.bg = TRUE, bg.col = "white", clipping = FALSE, ...)
```

Arguments

S	list of spheroids, see simPoissonSystem
box	simulation box
draw.axes	logical, whether to show the axes
draw.box	logical, whether to show the bounding box
draw.bg	logical, whether to show a gray background box correpsonding to the simulation box

`bg.col`	background color used showw the box background
`clipping`	logical, whether to clip all lateral planes of the simulation box
`...`	further material properties passed to 3D plotting functions

Details

The function requires the package '`rgl`' to draw spheroids. For an example please see the file 'simSpheroids.R'.

Author(s)

M. Baaske

`trivarHist` *Trivariate histogram*

Description

3D plot of a trivariate histogram

Usage

```
trivarHist(A, main = paste("Trivariate Histogram"), scale = 0.5, col,
  ...)
```

Arguments

`A`	3D array of count data (estimated histogram), see `unfold`
`main`	main title of the plot
`scale`	scaling factor for non-overlapping spheres
`col`	vector of color values repeatedly used for all classes
`...`	graphical parameters passed to `rgl::spheres3d`

Details

The (estimated spatial) joint size-shape-orientation distribution is plotted in a 3D histogram with corresponding axes. The axes intersect in the first class number. The ball volumes visualize the relative frequencies of count data for each class which can be scaled by the user in order to make the spheres non-overlapping. Balls within the same size class have the same color. For an example please see the file 'almmc.R'.

Author(s)

M. Baaske

unfold *Stereological unfolding*

Description

Unfolding the (joint) distribution of planar parameters

Usage

```
unfold(sp, nclass, maxIt = 64, nCores = getOption("par.unfoldr", 2L),
  ...)
```

Arguments

`sp`	section profiles, see `sectionProfiles`
`nclass`	number of classes, see details
`maxIt`	maximum number of EM iterations
`nCores`	number of cpu cores
`...`	optional arguments passed to `setbreaks`

Details

This is a S3 method for either trivariate stereological unfolding or estimation of the 3D diameter distribution of spheres which is better known as the *Wicksell's corpuscle problem*. The function aggregates all intermediate computations required for the unfolding procedure given the data in the prescribed format, see reference of functions below, and returning the characteristics as count data in form of a *trivariate* histogram. The section profile objects `sp`, see `sectionProfiles`, are either of class `prolate` or `oblate` for the reconstruction of the corresponding spheroids or, respectively, spheres. The result of the latter is simply a numeric vector of circle diameters. The number of bin classes for discretization of the underlying integral equations which must be solved is set by the argument `nclass`. In case of Wicksell's corpuscle problem (spheres as grains) this is simply a scalar value denoting the number of bins for the diameter. For spheroids it refers to a vector of length three defined in the order of the number of size, angle and shape class limits which are used. If `sp` is a numeric vector (such as for the estimation of the 3D diameter distribution from a 2D section of spheres) the function calls the EM algorithm as described in [3]. The return value of the function is an object of class "`unfold`" with elements as follows

- N_A (trivariate) histogram of section profile parameters
- N_V (trivariate) histogram of reconstructed parameters
- P array of coefficients
- breaks list of class limits for binning the parameter values

Value

object of class "`unfold`", see details

Author(s)

M. Baaske

See Also

`setbreaks`, `binning3d`

Examples

```
lam <- 100
# parameter rlnorm distribution (radii)
theta <- list("size"=list("meanlog"=-2.5,"sdlog"=0.5))

# simulation bounding box
box <- list("xrange"=c(0,5),"yrange"=c(0,5),"zrange"=c(0,5))
# simulate only 3D system
S <- simPoissonSystem(theta,lam,size="rlnorm",box=box,type="spheres",
  perfect=TRUE, pl=1)

# intersect
sp <- planarSection(S,d=2.5,intern=TRUE,pl=1)

# unfolding
ret <- unfold(sp,nclass=25)
cat("Intensities: ", sum(ret$N_V)/25, "vs.",lam,"\n")
```

`updateIntersections` *Check intersection*

Description

Check itersection of either spheres, spheroids or spherocylinders

Usage

```
updateIntersections(S)
```

Arguments

`S`	list of spheres, spheroids or spherocylinders, see `simPoissonSystem`

Details

For a given list of spheres, spheroids or spherocylinders the function tests whether an object intersects one of the lateral planes (bottom, top plane included) of the simulation box. The vector returned is of length equal to the number of objects in `S` and has entries either `1` for an object which is intersected by and `0` otherwise.

Value

binary integer vector of length equal to the length of `S`

Author(s)

M. Baaske

`verticalSection` *Vertical sections*

Description

Compute vertical section profiles of a spheroid system

Usage

```
verticalSection(S, d, n = c(0, 1, 0), intern = FALSE)
```

Arguments

`S`	list of spheroids, see `simPoissonSystem`
`d`	distance of the intersecting plane from the origin of the box
`n`	normal vector which defines the interecting vertical plane
`intern`	logical, `FALSE` (default), return all section profiles otherwise only those which have their centers inside the correspondig intersection window

Details

The function intersects a spheroid system by a plane defined by the normal vector `n` either equal to `c(0,1,0)` (default) or `c(1,0,0)`, which is called a vertical section. Depending on the type of spheroid (either `"prolate` or `"oblate"`) the returned semi-axis lengths are those corresponding to the minor semi-axis or, respectively, major semi-axis in the way these are required for unfolding.

Value

list of sizes `A`, shape factors `S` and (vertical) angles `alpha` of section profiles in the plane w.r.t the 'z' axis between $[0, \pi/2]$.

Author(s)

M. Baaske

Examples

```
 box <- list("xrange"=c(0,5),"yrange"=c(0,5),"zrange"=c(0,5))

 # (exact) bivariate size-shape (isotropic) orientation distribution (spheroids)
 theta <- list("size"=list("mx"=-2.5,"my"=0.5, "sdx"=0.35,"sdy"=0.25,"rho"=0.15),
"orientation"=list("kappa"=1))

 S <- simPoissonSystem(theta,lam=100,size="rbinorm",box=box,
  type="prolate",perfect=TRUE,pl=1)

 sp <- verticalSection(S,d=2.5,n=c(0,1,0),intern=TRUE)
 summary(sp$alpha)
```

Index