Package 'JFM'

January 20, 2025

Encoding UTF-8

Type Package

Title Rock Mass Structural Analysis from 3D Mesh of Point Cloud

Version 1.0

Date 2022-03-10

Description Provides functions to extract joint planes from 3D triangular mesh derived from point cloud and makes data available for structural analysis.

License GPL

Imports Rcpp (>= 0.12.18), MASS (>= 7.3.50), rgl (>= 0.99.16), RockFab (>= 1.2), Rvcg (>= 0.17), randomcoloR(>= 1.1.0)

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 7.1.2

NeedsCompilation yes

Author Riccardo Campana [aut, cre], Jeffrey R. Webber [ctb]

Maintainer Riccardo Campana <riccarl@hotmail.it>

Repository CRAN

Date/Publication 2022-03-28 08:00:02 UTC

Contents

`JFM-package` *JFM: A package for structural analysis.*

Description

Provides functions to extract joint planes from 3D triangular mesh and makes data available for structural analysis. Below compute_plane_normal function description an example test over all function is done. In your package directory you will find in extdata dir an example of point_cloud.txt file and in test folder two R scripts a test.R file to process point_cloud.txt file and a JFM_workflow.R generic workflow script.

Author(s)

Riccardo Campana <riccarl@hotmail.it>

`build_3d_mesh` *build_3d_mesh*

Description

This function reads a XYZRGB text file, requires a search radius in meters and an output file name to save the resulting mesh. for data format see file in package extdata folder

Usage

```
build_3d_mesh(path2myXYZRGBtxt, search_radius, file_name)
```

Arguments

`path2myXYZRGBtxt`	Path to the XYZRGB.txt input file
`search_radius`	Path to the XYZRGB.txt input file
`file_name`	name of the output .ply mesh file

Value

A 3D triangular mesh

Examples

```
## Not run: path2myXYZRGBtxt<-system.file("extdata", "test.txt", package = "JFM")

file_name<- "test"

mesh3d<-build_3d_mesh(path2myXYZRGBtxt,0.5,file_name)
## End(Not run)
```

`calculate_joints` *calculate_joints*

Description

This function calculates joint orientation with the least square method selecting vertexes of each facet plane

Usage

```
calculate_joints(vertici_tr, indici_tri, normal_from_wild)
```

Arguments

`vertici_tr`	list of facets vertexes coordinates ("vb property of mesh3d object")
`indici_tri`	list of facets indexes ("it property of mesh3d object")
`normal_from_wild`	matrix of data resulting from wildfire search

Value

a matrix of least square plane for each joint

Examples

```
## Not run:

mesh3d<-build_3d_mesh(path2myXYZRGBtxt,0.5,file_name)

normali_recalc<-Rcpp_wildfire_search(7,normals[,1:3],neighbours)

joint_list_Cpp<-calculate_joints(mesh3d,normali_recalc)
## End(Not run)
```

calculate_joints_area *calculate_joints_area*

Description

This function calculates the area of each cluster of facets belonging to the same plane

Usage

```
calculate_joints_area(normal_from_wild)
```

Arguments

normal_from_wild	matrix of data resulting from wildfire search

Value

a list of the area of each plane

Examples

```
## Not run:

normali_recalc<-Rcpp_wildfire_search(7,normals[,1:3],neighbours)

calculate_joints(mesh3d,normali_recalc)


## End(Not run)
```

compute_facets_normal *compute_facets_normal*

Description

This function returns a matrix of the three component vector of the normal of each facet.

Usage

```
compute_facets_normal(vertici_tr, indici_tri)
```

Arguments

vertici_tr	list of facets vertexes coordinates ("vb property of mesh3d object")
indici_tri	list of facets indexes ("it property of mesh3d object")

Value

matrix of the three component of the normal vector and area of each face

Examples

```
## Not run: indici_tri<-t(mesh3d[['it']])

vertici_tr<-t(mesh3d[["vb"]])

normals<-compute_facets_normal(vertici_tr,indici_tri)
## End(Not run)
```

---

compute_plane_area_rcpp

returns the sum of the area of facets belonging to the same plane

---

Description

returns the sum of the area of facets belonging to the same plane

Arguments

`tr_area`	a matrix with the first column facet area and second column the ID of plane it belows
`id_fam_no_zero`	the list of planes ID

Value

the sum of the area of facets belonging to the same plane given `tr_area` and `id_fam_no_zero`

---

compute_plane_normal

returns the least square plane from the vertexes of facets of the same plane (nested in calculate_joints function)

Description

returns the least square plane from the vertexes of facets of the same plane (nested in calculate_joints function)

Usage

```
compute_plane_normal(it_id_plane_points, vb_facets, id_fam_no_zero)
```

Arguments

it_id_plane_points	the "it"property of mesh object binded with ID column of widfire search
vb_facets	the vb property of mesh object (vertexes coordinates)
id_fam_no_zero	the list of planes ID

Value

returns the least square plane from the vertexes of facets of the same plane given it_id_plane_points the list of planes ID id_fam_no_zero, the matrix of vertexes coordinates vb_facets

Examples

```
 #This is an example of workflow script in test folder

path2myXYZRGBtxt<-system.file("test", "test.txt", package = "JFM")

file_name<- "test"

mesh3d<-build_3d_mesh(path2myXYZRGBtxt,0.5,paste0(tempdir(),"/",file_name))

vertici_tr<-t(mesh3d[["vb"]])

indici_tri<-t(mesh3d[['it']])

neighbours<-find_neighbours_rcpp(indici_tri)

### find neighbours of each triangle facet using a Rcpp function

neighbours<-find_neighbours_rcpp(indici_tri)

### or a hybrid R-Rcpp function

#### core number to dedicate to computational processes; check with
#### detectCores() function how many cores your pc owns

require("parallel")

detectCores()

### use only 2 cores

no_cores <- 2

neighbours<-findNeighbourFacets(no_cores,indici_tri)

### compute normal of each triangle facet

normals<-compute_facets_normal(vertici_tr,indici_tri)
```

```
### apply wildfire search

normali_recalc<-Rcpp_wildfire_search(7,normals[,1:3],neighbours)

### plot search result and if not satisfied repeat search increasing/decreasing tolerance angle

plotrand_col_planes(mesh3d,normali_recalc)

### calculate least square plane for each group of facets

joint_list_Cpp<-calculate_joints(vertici_tr,indici_tri,normali_recalc)

### calculate area for each group of facets

val_area<-calculate_joints_area(normali_recalc)

### extract pole maxima setting your minimum contour density
### and area to filter data, plot and save them

poles_maxima<-plot_joint_poles(normali_recalc,joint_list_Cpp,
                  val_area,paste0(tempdir(),"/",file_name),0.3,1)

##### plot and save great circle of pole maxima

azi_dip_maxima<-plot_joint_great_circles(poles_maxima, paste0(tempdir(),"/",file_name))

### plot colors of pole maxima onto mesh facets

plot_maxima2mesh(mesh3d,azi_dip_maxima,normali_recalc,10)

remove()
```

`compute_triangle_area_rcpp` *returns the area of a mesh facet*

Description

returns the area of a mesh facet

Arguments

`tr_vertex_coords` A 3x3 matrix of the coordinates of facet vertexes

findNeighbourFacets *findNeighbourFacets*

Description

This function finds the IDs of each mesh facet It requires number of cores of your pc to use and list of facets indexes corresponding to the "it" property of mesh3d object.

Usage

```
findNeighbourFacets(no_cores, indici_tri)
```

Arguments

no_cores	number of core to use in search computation
indici_tri	list of facets indexes ("it property of mesh3d object")

Value

a matrix of indexes of facets neighbours of target face saved on working dir

Examples

```
## Not run: indici_tri<-t(mesh3d[['it']])

require("parallel")

detectCores()

no_cores <- detectCores() - 4 ### keep free some cores

neighbours<-findNeighbourFacets(no_cores,indici_tri)
## End(Not run)
```

find_neighbours_rcpp *This function finds the rows IDs of neighbours of each mesh facet. It requires a list of facets indexes corresponding to the "it" property of mesh3d object*

Description

This function finds the rows IDs of neighbours of each mesh facet. It requires a list of facets indexes corresponding to the "it" property of mesh3d object

Usage

```
find_neighbours_rcpp(indici_tr)
```

Arguments

indici_tr	matrix of facets ID the "it" property of a mesh3D

Value

this function returns the rows IDs of neighbours of each mesh facet given a list of facets indexes indici_tri

Examples

```
indici_tri<-matrix(data = c(1, 2, 3 ,5, 6,
3, 2, 3, 5,7, 8 ,1),
nrow = 4,ncol = 3, byrow = TRUE)
find_neighbours_rcpp(indici_tri)
```

---

find_triangles_rcpp *returns the row indexes of the neighbour facets of a target facet (nested in findNeighbourFacets R function)*

Description

returns the row indexes of the neighbour facets of a target facet (nested in findNeighbourFacets R function)

Usage

```
find_triangles_rcpp(indici_tr, r)
```

Arguments

indici_tr	matrix of facets ID the "it" property of a mesh3D
r	index of the row of the target facet

Value

returns the row indexes of the neighbour facets of the facet at r row of indici_tr facet indexes matrix

Examples

```
indici_tri<-matrix(data = c(1, 2, 3 ,5, 6,
3, 2, 3, 5,7, 8 ,1),
nrow = 4,ncol = 3, byrow = TRUE)
row_index<-1
find_triangles_rcpp (indici_tri,row_index)
```

`least_square_plane_rcpp` *returns the coefficients of the least square plane and the relative mean square error*

Description

returns the coefficients of the least square plane and the relative mean square error

Usage

```
least_square_plane_rcpp(PointsXYZ)
```

Arguments

`PointsXYZ`	matrix of coordinates of point

Value

returns the coefficients of the least square plane and the relative mean square error of a set of 3d points `PointsXYZ`

Examples

```
list_xyz<-matrix(data = c(-10.0, -10.0, -15.0 ,10.0, -10.0,
-5.0, -10.0, 10.0, 5.0, 10.0, 10.0 ,15.0),
nrow = 4,ncol = 3, byrow = TRUE)
least_square_plane_rcpp(list_xyz)
```

`plotrand_col_planes` *plotrand_col_planes*

Description

This function returns a 3d plot of mesh where facets of the same plane are of same color.

Usage

```
plotrand_col_planes(mesh_tr, normal_from_wild)
```

Arguments

`mesh_tr`	an object of type mesh3d
`normal_from_wild`	the output matrix resulting from wildfire search

Value

a 3d plot of mesh with facets of the same plane

Examples

```
## Not run:

mesh3d<-build_3d_mesh(path2myXYZRGBtxt,0.5,file_name)

normali_recalc<-Rcpp_wildfire_search(7,normals[,1:3],neighbours)

plotrand_col_planes(mesh3d,normali_recalc)
## End(Not run)
```

---

plot_joint_great_circles

plot_joint_great_circles

Description

This function loads joint maxima poles, convert them to great circles and plot them on Schmidt stereogram. Data are also saved in working folder.

Usage

```
plot_joint_great_circles(giac_max, file_name)
```

Arguments

giac_max	Joint maxima poles returned from plot_joint_poles function
file_name	Name of the output data file

Value

A plot with great circles of joint maxima saved in working dir

Examples

```
## Not run:

poles_maxima<-plot_joint_poles(normali_recalc,joint_list_Cpp,val_area,file_name,max_pole,min_area)

azi_dip_maxima<-plot_joint_great_circles(poles_maxima, file_name)
## End(Not run)
```

plot_joint_poles *plot_joint_poles*

Description

This function plots on schmidt stereogram selected joints poles, draws density contour lines and retrieves poles maxima. Selected joints and poles maxima are saved in working folder.

Usage

```
plot_joint_poles(
  normal_from_wild,
  planes_mtx,
  area_ls,
  file_name,
  min_dens,
  plane_area
)
```

Arguments

normal_from_wild	the output matrix resulting from wildfire search
planes_mtx	the list of joints output from calculate_plane function
area_ls	the list of joints area output from calculate_planes_area function
file_name	Name of the output data file containing joint poles
min_dens	the minimum density pole value to be plotted
plane_area	minimum value of joint area to be considered in plot

Value

A Schmidt density plot with maxima values of joints

Examples

```
## Not run:

normali_recalc<-Rcpp_wildfire_search(7,normals[,1:3],neighbours)

joint_list_Cpp<-calculate_joints(mesh3d,normali_recalc)

val_area<-calculate_joints_area(normali_recalc)

file_name<-"my_out_file"

max_pole<-0.3
```

```
min_area<-1

poles_maxima<-plot_joint_poles(normali_recalc,joint_list_Cpp,val_area,file_name,max_pole,min_area)
## End(Not run)
```

---

plot_maxima2mesh *plot_maxima2mesh*

Description

This funtion plots a coloured mesh facets as great circles plot colours

Usage

```
plot_maxima2mesh(mesh_tr, planes_max, normal_from_wild, tol_ang_fam)
```

Arguments

mesh_tr	an object of type mesh3d
planes_max	the output of plot_joint_great_circles function
normal_from_wild	the output matrix iresulting from wildfire search
tol_ang_fam	a tolerance angle to include joints in the same joint set color

Value

A plot with great circles of joint maxima

Examples

```
## Not run:

azi_dip_maxima<-plot_joint_great_circles(poles_maxima, file_name)

plot_maxima2mesh(mesh3d,azi_dip_maxima,normali_recalc,10)

## End(Not run)
```

rcpparma_dotproduct *returns the inner product of ab and ac*

Description

returns the inner product of ab and ac

Arguments

ab	a 3D numeric vector
ac	a 3D numeric vector

Value

the dot product of ab and ac

Examples

```
a1<-c(1,2,3)
a2<-c(3,4,5)
rcpparma_dotproduct(a1,a2)
```

rcpp_crossProd *returns the outer product of ab and ac*

Description

returns the outer product of ab and ac

Usage

```
rcpp_crossProd(ab, ac)
```

Arguments

ab	a 3D numeric vector
ac	a 3D numeric vector

Value

the outer product of ab and ac

Examples

```
a1<-c(1,2,3)
a2<-c(3,4,5)
rcpp_crossProd(a1,a2)
```

Rcpp_wildfire_search *returns a matrix with the 3 components of each face normal vector; the 4th column is the ID of the plane each facet belongs to the 5th colum the area of each facet*

Description

returns a matrix with the 3 components of each face normal vector; the 4th column is the ID of the plane each facet belongs to the 5th colum the area of each facet

Usage

```
Rcpp_wildfire_search(tol_ang, list_of_normals, list_neighbours)
```

Arguments

`tol_ang`	the maximum angle between facets normal belonging to the same plane
`list_of_normals`	the matrix of the components of each facet normal vector
`list_neighbours`	the matrix of facets ID neighbours of each target facet

Value

the IDs of same joint facets given a `tol_angle` between facets normal and 3Dmesh `list_of_normals` and `list_neighbours`

Examples

```
## Not run: neighbours<-find_neighbours_rcpp(indici_tri)
normals<-compute_facets_normal(vertici_tr,indici_tri)
tol_ang<-7
normali_recalc<-Rcpp_wildfire_search(tol_ang,normals[,1:3],neighbours)
## End(Not run)
```

Index