Package ‘geophys’

October 13, 2022

Type Package

Title Geophysics, Continuum Mechanics, Gravity Modeling

Version 1.4-1

Date 2018-01-18

Depends R (>= 2.15)

Imports RPMG, RSEIS, RFOC, GEOmap, cluster

Suggests stats

Author Jonathan M. Lees [aut, cre],
Jake Anderson [ctb],
Francois Beauducel [ctb],
Dmitry Nicolsky [ctb]

Maintainer Jonathan M. Lees <jonathan.lees@unc.edu>

Description
Codes for analyzing various problems of geophysics, continuum mechanics and gravity models.

NeedsCompilation no

Repository CRAN

Date/Publication 2018-01-21 19:39:01 UTC

R topics documented:

---

geophys-package *Geophysics*

Description

Geophysics, Continuum Mechanics, Mogi Models, Mohr's Circles, Okada Model

Details

Package:	geophys
Type:	Package
Version:	1.2-0
Date:	2010-05-12
License:	GPL
LazyLoad:	yes

Author(s)

Jonathan M. Lees

Maintainer: Jonathan M. Lees <jonathan.lees@unc.edu>

References

Mogi

See Also

RSEIS, RFOC, GEOmap, zoeppritz

Examples

```
## Not run:

stress()


## End(Not run)
```

annotatebox — *Annotate Box*

Description

Show a box with corner points annotated

Usage

```
annotatebox(Rbox = matrix(ncol = 4, nrow = 8), add = TRUE)
```

Arguments

Rbox	Box matrix
add	logical, TRUE= add to existing plot

Value

Graphical Side effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

stress, stressSETUP, annotateplane

Examples

```
annotatebox()
```

annotateplane — *Annotate Plane*

Description

Show a box and plane with the three points annotated

Usage

```
annotateplane(Rp = matrix(ncol = 4, nrow = 3), add = TRUE)
```

Arguments

`Rp`	Points matrix
`add`	logical, TRUE= add to existing plot

Value

Graphical Side effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

stress, stressSETUP, annotatebox

Examples

```
annotateplane()
```

`AXB.prod` *Vector Cross Product*

Description

Vector Cross Product

Usage

```
AXB.prod(A, B)
```

Arguments

`A`	vector, 2 or 3 elements
`B`	vector, 2 or 3 elements

Details

Can be 2D or 3D vectors. If lists are input they are converted to vectors.

Value

vector

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

Examples

```
AXB.prod(c(1,0,0), c(0,1,0)) ###  =  c(0, 0 , 1)
AXB.prod(c(0,1,0), c(1,0,0)) ###  =  c(0, 0 ,-1)
```

BMOD *Plot block model for gravity*

Description

Plot block model with several polygonal structures for gravity modeling.

Usage

```
BMOD(bill, nstn = 100, PLOT = TRUE, obs=NULL)
```

Arguments

`bill`	Polygon data from ModelG
`nstn`	number of stations
`PLOT`	logical, TRUE=plot
`obs`	Observed Profile Gravity (x,g)

Details

Used internally in interactive modeling.

Value

graphical Side effects.

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

ModelG

centroid *centroid*

Description

calculate the centroid of a polygon

Usage

```
centroid(p)
```

Arguments

p	list: x,y

Details

same code as centroid.polygon

Value

vector x, y

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

Examples

```
L=list()
L$x=c( 0.2188,-0.6668,-1.0992,-0.5105, 0.5106)
L$y=c( 0.76822, 0.75292,-0.01741,-0.74692,-0.48165)
centroid(L)
```

desh *Display Mesh*

Description

Display Mesh

Usage

```
desh(M, add = TRUE, PTS = TRUE, colmesh = grey(0.8), colpts = grey(0.5),
...)
```

Arguments

M	Mesh output of meshgrid
add	logical, add to plot
PTS	logical, add points
colmesh	color for mesh
colpts	color for points
...	graphical parameters from par

Details

Used for showing strain mesh plots.

Value

graphical side effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

meshgrid

Examples

```
###  warping of the ground from a Mogi source
 P=1e5
 a = 0.01
         f = .1
      nu=0.25
         E = 10e13
     mu = E/(2*(1+nu));
```

```
EX = seq(from=-3, by=.4, to= 3)
     mm = RPMG::meshgrid(EX, EX)
  rho = sqrt(mm$x^2 +  mm$y^2)

cen = list(x=mean(EX), y=mean(EX))

are = sqrt( (mm$x-cen$x)^2  + (mm$y-cen$y)^2 )


 o1 = mogi1(a, P, rho, mu, f)


shiftx =  o1$ur*(mm$x-cen$x)/are
shifty =  o1$ur*(mm$y-cen$y)/are

newmesh = list(x=mm$x+shiftx , y=mm$y+shifty)

desh(mm, add=FALSE, PTS=FALSE, colmesh=grey(.8) )


desh(newmesh, add=TRUE, PTS=FALSE, colmesh=rgb(.6,.8, .6) )
```

DGzx *Gravity anomaly in 2.5D*

Description

Gravity anomaly in 2.5-Dimensions from an arbitrary polynomial at many stations.

Usage

```
DGzx(xs, zs, xv, zv, den)
```

Arguments

xs	station locations in X
zs	station locations in Z
xv	x-vertices
zv	z-vertices
den	density contrast

Details

calculate the 2.5D solution to gravity. Orientation of the vertices should be right handed.

Value

vector of Delta-Gz and Delta-Gx at each station

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Won and Bevis (1987) Computing the gravitational and magnetic anomalies due to a polygon: Algorithms and Fortran subroutines <doi:https://doi.org/10.1190/1.1442298>

Examples

```
nstn = 10
xstart = -10000
xend = 10000
xcen = 0
zcen = 5000
RAD = 2000


xs = seq(from=xstart, by=(xend-xstart)/nstn , length=nstn)
zs = rep(0, length=length(xs))

den = 0.2

Np = 6
theta = seq(from=0, to=2*pi, length=Np)
KZ = list(x=NA, y=NA)
KZ$x = xcen+RAD*cos(theta)
KZ$y = zcen+RAD*sin(theta)




Ngrav = DGzx(xs, zs,  KZ$x, KZ$y, den)
```

dircheck — *Check direction of polygon*

Description

Check direction of polygon for Right Handed orietnation

Usage

```
dircheck(pol)
```

Arguments

pol	list of x,y

Details

Gets cross product of consecutive points in polygon and returns the sign of each. If most are positive, should try to reverse.

Value

vector of signs

Note

Co-linear values should return zero.

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

rev2RH, flipZEE

Examples

```
L=list()
L$x=c( 0.2188,-0.6668,-1.0992,-0.5105, 0.5106)
L$y=c( 0.76822, 0.75292,-0.01741,-0.74692,-0.48165)
dircheck(L)

K = rev2RH(L)
###  after reversing,
dircheck(K)
```

DO.DYKE *Heat for Dyke Intrusion*

Description

Solve heat equation for dyke intrusion.

Usage

```
DO.DYKE(a = a, x = x, t = t, k = k, T0 = T0, NDIM = TRUE)
```

Arguments

a	Thickness of dyke (meters)
x	vector of distances across dyke (m)
t	time in seconds
k	diffusion constant
T0	Temperature at infinity
NDIM	logical, TRUE=use nondimensional temperature for text labels

Value

graphical side effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

get.heat2, DO.HALFSPACE

Examples

```
 a = 10;
k=10^6


  DO.DYKE(NDIM=TRUE)

####  dimensional variable plot
   DO.DYKE(NDIM=FALSE)
```

DO.HALFSPACE *Heat Equation in Halfspace*

Description

Heat Equation in Halfspace

Usage

```
DO.HALFSPACE()
```

Details

Several solution for different times are displayed in succession.

Value

Graphical Side effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

DO.DYKE

Examples

```
DO.HALFSPACE()
```

dofry *Fry Estimation*

Description

Calculate the Fry diagram for estimating shear on random point patterns

Usage

```
dofry(x, y, PLOT = FALSE)
```

Arguments

x	x-coordinates
y	y-coordinates
PLOT	logical, TRUE=plot

Details

The number of points returned can be large and plotting can take a long time, so when plotting can restrict the points by using plotfry.

Value

List:

X	x-coordinates
Y	y
mx	mean X
my	mean Y

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Fry, N., (1979) <http://dx.doi.org/10.1016/0040-1951(79)90135-5> Random point distributions and strain measurement in rocks Tectonophysics, 60:89-105.

See Also

Showfry, plotfry, xtractlip

Examples

```
RDAT = randFRY(400, LIM=c(0,0, 200, 200) , rlen=5   )
length(RDAT$x)
plot(RDAT$x, RDAT$y, asp=1, pch=".", cex=2)
flag = sqrt( (RDAT$x-mean(RDAT$x))^2  + (RDAT$y-mean(RDAT$y))^2)<75

DAT = list(x=RDAT$x[flag], y=RDAT$y[flag])
 x = DAT$x
    y = DAT$y
 plot(x,y, asp=1, pch=21, col='red', bg='gold', ann=FALSE, axes=FALSE)

FF = dofry(DAT$x, DAT$y )
AF = plotfry(FF, dis=30)

Z = xtractlip(AF)
```

```
lines(cluster::predict.ellipsoid(Z$hull), col='red')
```

DoMohr *Mohr's Circle*

Description

Mohr's Circle

Usage

```
DoMohr(Stensor=diag(c(3,2,1)), axis=NULL)
```

Arguments

`Stensor`	Stress tensor
`axis`	vector of axes to plot, c(1,2,3,4)

Details

Stress tensor is a 2 by 2 or 3 by 3 symmetric matrix

Value

Graphical Side effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

W.D. Means, Stress and Strain: Basic Concepts of Continuum Mechanics for Geologists, Springer; 1976. 4th ed., 339 p. (ISBN-10:0387075569).

See Also

DoMohrFig1, stress

Examples

```
Stensor = matrix(c(2,.3, .3, 1), ncol=2)

Stensor = matrix(c(50, 40, 40, 10), ncol=2)

DoMohr(Stensor)


Stensor = matrix(c(
-50, 40, 20,
40, 10, 10,
20,  10, 5), ncol=3)

DoMohr(Stensor)
```

DoMohrFig1 *Annotated Stress Tensor*

Description

Annotated plot of a 2D stress tensor

Usage

```
DoMohrFig1(Stensor=matrix(c(5,1, 1, 3), ncol=2), rot1=NULL)
```

Arguments

Stensor	stress tensor
rot1	rotation matrix

Value

Graphical Side effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

DoMohr

Examples

```
Stensor =matrix(
       c(50, 40,
         40, 10), ncol=2)

DoMohrFig1(Stensor)
```

---

draw.brachiopod — *Draw Brachiopod*

Description

Draw Brachiopod at a particular location

Usage

```
draw.brachiopod(BK = matrix(), x = 0, y = 0, col = "black", fill = NULL, ...)
```

Arguments

BK	brachiopod matrix
x	x-coordinate location on plot
y	y-coordinate location on plot
col	col for outline
fill	color for fill
...	other par parameters for plotting

Details

Used for strain illustrations

Value

side effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

get.brachiopod

Examples

```
gb = get.brachiopod()

draw.brachiopod(gb)



BB = get.brachiopod()
phi0=0
phi =   rep(phi0, 6)
shears = seq(from=0, to=1.5, length=6)
shearmats = list()
for(i in 1:length(shears))
{
 shr = shears[i]
  shearmats[[i]] = matrix(c(1, shr, 0,  1), ncol=2)
}

phi0=0
phi =   rep(phi0, 6)
phi=runif(6, 0, 90)
shears = rep(0, 6)


 for(i in 1:length(phi))
        {
          shr = shears[i]
          shearmats[[i]] = matrix(c(1, shr, 0,  1), ncol=2)
        }


x =  rep(c(25,75), times=3)

y = as.vector( t(matrix( rep(c( 75, 50,25 ), times=2) , ncol=2)) )


siz  = rep(30, 6)


plot(c(0, 100), c(0, 100), asp=1, type='n', ann=FALSE, axes=FALSE)
for(i in 1:length(x))
  {
    cosp=cos(phi[i]*pi/180)
    sinp=sin(phi[i]*pi/180)
#######  make a rotation matrix
#### rmat=matrix(c(cosp,-sinp, x, sinp, cosp, y), ncol=2)


 ####
    rmat1=matrix(c(cosp,-sinp, sinp, cosp),  ncol=2)
```

```
    total =  rmat1

    rmat= rbind( total, c(x[i],y[i]) )

    n = length(BB[,1])

    RACK1 = cbind(siz[i]*BB[,1:2], rep(1,n))

   draw.brachiopod(RACK1)

 }
```

`elipfit` *Least Squares Ellipse*

Description

Fit best Ellipse to a set of points using least squares.

Usage

```
elipfit(ex, ey, PLOT = FALSE, add = TRUE, ...)
```

Arguments

`ex`	x-coordinates
`ey`	y-coordinates
`PLOT`	Logical, TRUE plots lines
`add`	logical, add=TRUE adds to existing plot, FALSE starts new plot
`...`	graphical parameters from par

Details

Program removes mean values from x-y points prior to fitting

Value

Vector of semi major axes and angle (radians)

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

ellipsoidhull, xtractlip

Examples

```
EL=list()
EL$x=c(-9.478057124959,-8.887247178079,-7.679217676939,-5.843342603491,
-4.827065854640,-4.524213862912,-3.640069016562,-3.181663570301,
-2.830460290732,-2.276944364728,-0.258719497667, 5.987620544313,
5.262029128773, 4.233925281717, 3.463189680754, 3.295838697506,
1.127493106453, 0.611037858455, 0.308185866727,-1.034364425884,
-1.385567705454,-1.939083631457,-3.957308498518)
EL$y=c(-7.379108143633,-8.077163749968,-8.353322965687,-8.375311165710,
-8.133874868636,-8.988274156710,-7.715168397536,-7.629855105316,
-7.978224323725,-7.658390735870,-5.981004273181, 0.550983019511,
0.717119185592, 1.901196979324, 1.691334007646, 2.938457310537,
1.506850181485, 1.471885910595, 2.326285198669, 0.967866147275,
1.316235365684, 0.996401777829,-0.680984684860)


plot(EL, asp=1)

LL = elipfit(EL$x,EL$y, PLOT=TRUE, add=TRUE, col='purple')
```

---

erf *Error Function*

Description

Error Function

Usage

```
erf(x)
philpotts.erf(x)
erfc(x)
erfinv(x)
```

Arguments

x

Details

Uses pnorm to calculate the error function. erfinv returns the inverse of the erf function.

Value

error function

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

philpotts.erf

Examples

```
j = seq(from=0, to=5.5, by=0.01)
 y = erf(j)
phil = philpotts.erf(j)

 plot(j,y)
points( j, phil, pch=3, col='red'  )


#### plot the difference
plot(j, y-phil)
lines(j, y-phil)


#####  inverse of erf function:

 j = seq(from=0, to=5.5, by=0.1)
 y = erf(j)

for(i in 1:length(j))
   {

     z = erfinv(y[i])
     print(paste(i, y[i], j[i], z, sep=" "))
   }
```

---

flipZEE *Flip Z direction*

Description

Flip the zee direction so negative is down.

Usage

```
flipZEE(pol)
```

Arguments

`pol`	list of x,y

Details

Reverse the sign of the y-direction to achieve negative down Z values.

Value

same as input with -y

Note

Uses the y component as if it were Z.

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

rev2RH, dircheck

Examples

```
L=list()
L$x=c( 0.2188,-0.6668,-1.0992,-0.5105, 0.5106)
L$y=c( 0.76822, 0.75292,-0.01741,-0.74692,-0.48165)
flipZEE(L)
```

`get.brachiopod` *Brachiopod icon*

Description

set Brachiopod icon

Usage

```
get.brachiopod()
```

Details

Program returns the outline of a brachiopod for strain analysis.

Value

2D Matrix of coordinates.

Note

Used in conjunction with draw.brachiopod

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

draw.brachiopod

Examples

```
gb = get.brachiopod()

draw.brachiopod(gb)
```

---

get.heat2 — *Heat Equation Solution*

Description

Solution of the diffusion equyation when contact is made instananeously with an infinite half space. Used for sudden exposure of magma at the surface, as in an oceanic ridge.

Usage

```
get.heat2(x, T0, k, t)
```

Arguments

x	numeric vector, distance
T0	Temperature at infinity
k	diffusion constant
t	time

Details

get.heat2 is used when temperature at the boundary (atmosphere) remains constant for all time. get.heat is used when temperature in half space changes (warms up).

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

Examples

```
 k = 1*10^(-6)
dt = 3600
 dz = 20*10^(-2)

T0=25

T1 = 1200



x =  seq(from=0, to=80, by=0.5)/100

i = 1
Tx =  get.heat2(x, T1-T0, k, i*dt)

plot(Tx, max(x)-x, type='n', xlim=c(700, 1200) , axes=FALSE, xlab="Temp", ylab="Depth, cm")
axis(3)
axis(2, at=pretty(x), labels=100*(max(x)-pretty(x)))

for(j in 1:5)
{
Tx =  get.heat2(x, T1-T0, k, j*dt)
Tx = Tx+T0
lines(Tx, max(x)-x, lty=2, col=j)
}
```

`Glines` *Igneous Petrology Lines*

Description

Scanned lines from igneous petrology

Usage

```
data(Glines)
```

Format

The format is: List of 10 $:List of 2 ..$ x: num [1:77] 55.2 56.1 57.2 58.5 59.5 ... ..$ y: num [1:77] 1.62 1.67 1.82 2.15 2.35 ... $:List of 2 ..$ x: num [1:6] 50.7 51.7 53.1 54.4 56.3 ... ..$ y: num [1:6] 9.21 9.16 9.24 9.21 9.09 ... $:List of 2 ..$ x: num [1:14] 41.7 41.9 42.8 44.9 48.4 ... ..$ y: num [1:14] 3.35 3.55 4.22 5.82 8.49 ... $:List of 2 ..$ x: num [1:13] 52.1 52.2 52.2 52.2 52.3 ... ..$ y: num [1:13] 1.72 2.7 3.77 5.27 5.67 ... $:List of 2 ..$ x: num [1:20] 45.5 46.8 47.8 48.8 49.7 ... ..$ y: num [1:20] 5.74 5.69 5.69 5.69 5.64 ... $:List of 2 ..$ x: num [1:2] 63.2 63.4 ..$ y: num [1:2] 6.64 3.45 $:List of 2 ..$ x: num [1:2] 55.2 55.2 ..$ y: num [1:2] 5.34 1.75 $:List of 2 ..$ x: num [1:2] 46.7 53.2 ..$ y: num [1:2] 7.04 7.07 $:List of 2 ..$ x: num [1:2] 49.7 55.1 ..$ y: num [1:2] 14.7 11.2 $:List of 2 ..$ x: num [1:5] 58.8 60 61.8 62.3 62.5 ..$ y: num [1:5] 11.5 12.3 13.4 13.8 13.9

Details

Can click on here with a mixture to get a location on the plot.

Source

scanned figure

References

Principles of Igneous and Metamorphic Petrology, A. R. Philpotts, 1990, Prentice Hall (ISBN-13:978-0521880060)

Examples

```
data(Glines)
data(PPoints)

plot(Glines[[1]], type='n', xlab="SiO2", ylab="NA2O+K2O")
for(i in 1:length(Glines))
{

lines(Glines[[i]]$x, Glines[[i]]$y)

}

for(i in 1:length(PPoints$x))
{
text(PPoints$x[i],PPoints$y[i], label=PPoints$lab[i], srt=PPoints$rots[i])
}
```

`heat.sol` *Heat Equation Solution*

Description

Heat Equation Solution

Usage

```
heat.sol(x, T0, k, t)
```

Arguments

`x`	distance
`T0`	temperature
`k`	diffusion constant
`t`	time vector

Details

INteractive code to see evolution

Value

Graphical Side Effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

get.heat, get.heat2,DO.HALFSPACE

Examples

```
## Not run:
k = 1e-6
 t  = 1*365*24*60*60
T0 = 1000
x = seq(0,20, length=1000)
T =  heat.sol(x, T0, k, t)


## End(Not run)
```

labelLine *Label a Line Segment*

Description

Label a Line Segment

Usage

```
labelLine(P1, P2, above = TRUE, dinch = 0.2, lab = "text",
 acode = 3, alength = 0.06, aty = 1, acol = "black", bty = 1,
 bcol = "black", tcol = "black", font = 1, cex = 1)
```

Arguments

P1	Point 1 c(x,y)
P2	Point 2 c(x,y)
above	logical, TRUE=label above the line, else below
dinch	length the legs
lab	character, text label
acode	code for arrows, see arrows
alength	length for arrows
aty	lty for arrows
acol	color for arrows
bty	style for legs
bcol	color for legs
tcol	color for text
font	font for text
cex	character expansion for text, see par

Details

Two short lines are drawn perpendicular to the line between the points, the length of this line is dinch. The arrow is drawn between the legs, using the parameters provided.

Value

graphical side effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

par, arrows, text

Examples

```
 V1 = c(runif(1) ,  runif(1),runif(1) , runif(1))


 P1 = c(V1[1], V1[3])
P2 =  c(V1[2], V1[4])
 plot(c(P1[1], P2[1]), c(P1[2], P2[2] ), asp=1, type='n' )
   arrows(P1[1], P1[2], P2[1], P2[2], length=.04, col='red')

   labelLine( P1, P2 , lab="ABOVE", dinch = .5,
           aty=2, acol='blue'  )
   labelLine( P1, P2 , above=FALSE, lab="below",
         dinch = .5, aty=2, acol='green', tcol="magenta"  )
```

`lipper` *Ellipse calculator*

Description

Return parametric version of ellipse from equation of an ellipse.

Usage

```
lipper(a, b, c, d = 0, f = 0, g = -1)
```

Arguments

a	numeric
b	numeric
c	numeric
d	numeric
f	numeric
g	numeric

Details

General quadratic equation is, $ax^2 + 2bxy + cy^2 + 2dx + 2fy + g = 0$

Value

vector=c(ap, bp, phi)

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

<http://mathworld.wolfram.com/Ellipse.html>

Examples

```
LIP = lipper(10, 3, 9)

theta  = seq(from=0, to=2*pi, length=360)

        phi = LIP[3]
        px = LIP[1]*cos(theta)*cos(phi)-  LIP[2]*sin(theta)*sin(phi)
        py = LIP[1]*cos(theta)*sin(phi) + LIP[2]*sin(theta)*cos(phi)


plot(px, py, type='p', asp=1)
lines(px, py)
```

`Maxstress` *Maximum Stress*

Description

Maximum Normal and Shear stress in a plane

Usage

```
Maxstress(NN, Stensor)
```

Arguments

`NN`	Normal Vector
`Stensor`	Stress tensor

Details

Stress tensor can be entered as a vector of 3 or six values or as a 3 by 3 matrix. If three values are provided provided they are assumed to be the principal stresses. If six values are provided, they are c(x11, x12, x13, x22, x33). In either case eigen values and eigen vectors are calculated and used in the following analysis.

Value

NN	Normal vector to plane is returned
sigNORMmax	maximum normal stress
tauSHEARmax	maximum shear stress in the plane

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

stress

Examples

```
  S = stressSETUP( )

Nvec = NORMvec(S$PPs, S$xscale, S$Rview, S$aglyph , add = FALSE)

Stensor = matrix(c(
15, 0, 0,
0, 8, 0,
0,  0, 5), ncol=3)



Mstress  = Maxstress(Nvec, Stensor)

DoMohr(Stensor)
 axis(1)
axis(2)


points(Mstress$sigNORMmax , Mstress$tauSHEARmax, pch=21, col='blue'  , bg='gold' )

u=par('usr')

segments(0, Mstress$tauSHEARmax, Mstress$sigNORMmax ,
Mstress$tauSHEARmax, lty=2, col='green'  )

text(mean(c(0, Mstress$tauSHEARmax)),  Mstress$tauSHEARmax,
"MaxShear in Plane", pos=3)


segments(Mstress$sigNORMmax , u[3] , Mstress$sigNORMmax ,
Mstress$tauSHEARmax, lty=2, col='purple'  )

text(Mstress$sigNORMmax , u[3], "MaxNormal stress", adj=c(0,-1) )
```

ModelG *Interactive 2.5D Gravity Modeling*

Description

Interactive 2.5D Gravity Modeling

Usage

```
ModelG(Data, labs = c("Done"), obs=NULL, ZCOLS=RPMG::pastel.colors(24, seed=2 ))
```

Arguments

`Data`	list: xmin, xmax, zmin, zmax
`labs`	Labels
`obs`	Observed Profile Gravity (x,g)
`ZCOLS`	Color Palette for plotting blocks

Details

Forward modeling using the code of Won and Bevis.

For input, at a minimum, one must provide the boundary of the model region, xmin, xmax, ymin, ymax (or depth). more comprehensive input, i.e. with a starting model, will provide a model in addition: xmin, xmax, zmin, zmax, mod, cens, n. The mod is a list of polygons, each with xy coordinates and a delta-rho value.

Value

Model

Note

You type in the polygons interactively and execute the gravity forward modeling.

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Won and Bevis (1987) Computing the gravitational and magnetic anomalies due to a polygon: Algorithms and Fortran subroutines <doi:https://doi.org/10.1190/1.1442298>

See Also

DGzx

Examples

```
## Not run:
blum = list(xmin=0, xmax=26000, zmin=0, zmax=-1000)
gill = ModelG(blum)



## End(Not run)
```

`mogi1` *Mogi's model*

Description

Mogi's model (point source in elastic half-space)

Usage

```
mogi1(d = 1, f = 1, a = 0.1, P = 1e+05, mu = 4e+09, nu = 0.25)
```

Arguments

`d`	distance along surface, m
`f`	depth to source, m
`a`	radius of sphere injected, m
`P`	hydrostatic pressure of injection, Pa
`mu`	shear modulus
`nu`	Poisson's ratio

Details

Units should be constistent, e.g.: R, F, A, Ur and Uz in m imply V in m3; E, mu and P in Pa; Dt in rad, Er, Et and nu dimensionless.

Value

list

`ur`	radial displacement
`uz`	vertical displacement

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Mogi, K., Relations between the eruptions of various volcanoes and the deformations of the ground surfaces around them, Bull. Earthquake Res. Inst. Univ. Tokyo, 36, 99-134, 1958.

Examples

```
P=1e5
a = 0.1
    f = 1

 nu=0.25
    E = 10e9

mu = E/(2*(1+nu));

#############
rho = seq(from=-3, to=3, by=0.1)
o1 = mogi1(a, P, rho, mu, f)
plot(rho, o1$uz)

plot(rho, o1$ud)







EX = seq(from=-3, by=.1, to= 3)

mm = RPMG::meshgrid(EX, EX)

rho = sqrt(mm$x^2 +  mm$y^2)

o1 = mogi1(a, P, rho, mu, f)

#############
image(EX, EX, o1$uz, asp=1)


contour(    EX, EX, o1$uz, add=TRUE)
#############
image(EX, EX, o1$ur, asp=1)


contour(    EX, EX, o1$ur, add=TRUE)

#############
image(EX, EX, o1$uz, asp=1)
contour(    EX, EX, o1$ur, add=TRUE)
```

mogiM *Mogi Model*

Description

Mogi model deformation returns the deformation from a point source presurized inflation in an elastic medium.

Mogi's model (point source in elastic half-space). computes radial and vertical displacements Ur and Uz, ground tilt Dt, radial and tangential strain Er and Et on surface, at a radial distance R from the top of the source due to a hydrostatic pressure inside a sphere of radius A at depth F, in a homogeneous, semi-infinite elastic body and approximation for A « F (center of dilatation). Formula by Anderson [1936] and Mogi [1958].

Usage

```
mogiM(R = 1, F = 1, A = 0.1, P = 1e+05, E = 1e+10, nu = 0.25)
```

Arguments

`R`	Hoirizontal Distance frm source, m
`F`	Depth below surface, m, positive down
`A`	radius of magma chamber
`P`	hydrostatic pressure change in the sphere
`E`	elasticity (Young's modulus)
`nu`	Poisson's ratio

Details

Original paper by Mogi used poisson's ratio equale to 0.25, i.e. lame parameters lambda and nu were equal.

Value

list:

`ur`	radial displacements Ur
`uz`	vertical displacements Uz, Uz > 0 = UP
`dt`	ground tilt Dt
`er`	radial strain Er
`et`	tangential strain Et on surface

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Anderson, E.M., Dynamics of the formation of cone-sheets, ring-dikes, and cauldron-subsidences, Proc. R. Soc. Edinburgh, 56, 128-157, 1936.

Mogi, K., Relations between the eruptions of various volcanoes and the deformations of the ground surfaces around them, Bull. Earthquake Res. Inst. Univ. Tokyo, 36, 99-134, 1958.

Examples

```
data(PXY)

delV = 2.3E13/(100^3)    ##### (convert to meter^3 from cm^3)
F = 2.8E5/100      ##### (convert to meter from cm    )

EX = seq(from=0, by=100, to= 9000)


Atest = mogiM(R=EX,F=F,A=delV)



 plot(PXY, pch=6, col='purple', xlim=c(0,9), ylim=c(0, 1) )
    ###  model
    lines(EX/1000, Atest$uz/max(Atest$uz))


############ best fit   optimization

library(stats)

    fr<-function(x)
      {

        Atest = mogiM(R=PXY$x*1000 ,F=x[1],A=x[2])


        rms = sum ( (PXY$y - Atest$uz/max(Atest$uz))^2 )

        return(rms)
      }
xin = c(2600, 2.0e+07)

FOUT = stats::optim(xin , fr)

  Btest = mogiM(R=EX,F=FOUT$par[1] ,A=FOUT$par[2])

   plot(PXY, pch=6, col='purple', xlim=c(0,9), ylim=c(0, 1) )

 lines(EX/1000, Btest$uz/max(Btest$uz))
```

`mohrleg` *Legend for Mohr*

Description

Legend for Mohr

Usage

```
mohrleg(ES)
```

Arguments

`ES`	Eigen Value Decomposition, output of function eigen

Details

Add notes to plots of Mohr's circles. Uses the eigenvalues of the decomposition.

Value

Graphical Side Effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

DoMohr

Examples

```
Stensor = matrix(c(50, 40, 40, 10), ncol=2)

DoMohr(Stensor)
```

NinePointCircle *Nine Point Circle*

Description

Nine Point cirlce of a triangle

Usage

```
NinePointCircle(P1, P2 = c(0, 1), P3 = c(1, 0), add = FALSE, SHOW = TRUE)
```

Arguments

P1	vector, Point 1
P2	vector, Point 1
P3	vector, Point 1
add	add to existing plot
SHOW	create a new plot and add

Details

circle passes through nine points that can be calculated for any triangle. Also known as Feuerbach's circle, Euler's circle, Terquem's circle, the six-point circle, the twelve-point circle, the n-point circle, the medioscribed circle, the mid circle, the circum-midcircle.

Value

list of essential points:

A	2-vector, vertex point 1
B	2-vector, vertex point 2
C	2-vector, vertex point 3
D	2-vector, mid-point opposite A
E	2-vector, mid-point opposite B
F	2-vector, mid-point opposite C
G	2-vector, foot altitude point opposite A
H	2-vector, foot altitude point opposite B
I	2-vector, foot altitude point opposite C
J	2-vector, mid point from S-A
K	2-vector, mid point from S-B
L	2-vector, mid point from S-C
S	2-vector, Intersection point of altitudes
CEN	2-vector, center of nine point circle
R	radius of nine point circle

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

<http://en.wikipedia.org/wiki/Nine-point_circle>

See Also

TriangleInfo, TriangleCenter

Examples

```
P1 = 10*runif(2)
    P2 =  10*runif(2)
    P3 =  10*runif(2)

    TRI =  NinePointCircle(P1, P2, P3, add=TRUE, SHOW=TRUE)
```

NORMvec *Plot Normal Vector*

Description

Calculate and plot a normal vecotor to a plane

Usage

```
NORMvec(PPs, xscale, Rview, aglyph = list(), add = TRUE)
```

Arguments

PPs	Points for plotting, these define the plane
xscale	scale for the figure
Rview	viewing matrix
aglyph	glyph for plotting the normal vector in 3D
add	logical, whether to add to current plot

Value

Graphical Side effects

Note

Used internally

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

pstart, PLOTbox, PLOTplane, stress

Examples

```
#######    set colors
 axcol = 'black'
 boxcol = 'blue'
 planecol = 'brown'

####   view angle
 Rview  =    RFOC::ROTZ(-130)

###   set arbitrary scale
    xscale = 100

###  set glphy for plotting 3D vectors
 headlen =xscale* .3/6
       len =xscale* .7/6
       basethick =xscale* 0.05/2
       headlip =xscale* .02/2
       aglyph = RFOC::Z3Darrow(len = len , basethick =basethick ,
             headlen =headlen , headlip=headlip )

####   set three points and combine them with rbind
     P1 = xscale*c(.2, 1,1,0)
    P2 = xscale*c(1, .1,1,0)
    P3 = xscale*c(1, 1,.4,0)

    PPs = rbind(P1, P2, P3)

  ###  convert points for view
Rp = PPs

###   create box  and scale it
   BOX <-matrix(c(0,0,0,0,
            0, 1, 0,0,
            0, 1, 1,0,
            0, 0, 1,0,
            1,0,0,0,
            1, 1, 0,0,
            1, 1, 1,0,
            1, 0, 1,0), ncol=4, byrow=TRUE)
```

```
    BOX = xscale*BOX

    ##############   create axes and scale them
AX = matrix(c(0,0,0,0,
            1, 0, 0,0,
            0, 0, 0,0,
            0, 1, 0,0,
            0,0,0,0,
            0, 0, 1,0), ncol=4, byrow=TRUE)

    AX = 1.5*xscale*AX

 ##############   rotate axes and box
    Rax =  AX


          Rbox =   BOX

 ##############  start the figure
pstart(xscale=xscale)
 ######  plot the box
PLOTbox(Rax, Rbox, axcol= 'black', boxcol= 'blue')
 ##############  plot the plane
PLOTplane(Rp)

#######  plot the normal
 NN = NORMvec(PPs, xscale, Rview,aglyph=aglyph, add=TRUE)
```

---

okada85 *Okada Fault Deformation Model*

Description

Calculate the deformation due to a fault based on Okada 1985 code.

Usage

```
okada85(e = 0, n = 0, depth = 0, strike = 20,
dip = 20, L = 5, W = 3, rake = 20, slip = 1, U3 = 1, nu = 0.25)
```

Arguments

e	vector or matrix, Easting Observation, km
n	vector or matrix, Northing Observation, km
depth	depth of fault
strike	degrees, strike of fault, measured from north

dip	degrees, strike of fault, measured from horizontal
L	Length, km
W	Width, km
rake	rake (degrees)
slip	slip
U3	dilitation (m)
nu	Poisson's ration, default=0.25

Details

Computes displacements, tilts and strains at the surface of an elastic half-space, due to a dislocation defined by RAKE, SLIP, and OPEN on a rectangular fault defined by orientation STRIKE and DIP, and size LENGTH and WIDTH. The fault centroid is located (0,0,-DEPTH).

R-Code is a translation of original Matlab code by Francois Beauducel. UNC student Lan Ngo worked on the first version and J. M. Lees tested, verified and finalized the package.

Value

list:

uE	East deformation
uN	North deformation
uZ	Vertical deformation
uZE	tilt vertical east
uZN	tilt vertical north
uNN	horizontal strains North-North
uNE	horizontal strains North-East
uEN	horizontal strains East-North
uEE	horizontal strains East-East

Note

Units should be in km.

Note that vertical strain components can be obtained with following equations: uNZ = -uZN; uEZ = -uZE; uZZ = -(uEE + uNN)*NU/(1-NU);

Be careful about the definition of strike: seems that here the strike may be measured positive north of east, not east of north as stated in the documentation above.

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Aki K., and P. G. Richards (1980) (ISBN-13: 978-1891389634) Quantitative seismology, Freemann & Co, New York.

Okada Y., Surface deformation due to shear and tensile faults in a half-space, Bull. Seismol. Soc. Am., 75:4, 1135-1154, 1985.

Acknowledgments: Francois Beauducel, Dmitry Nicolsky, University of Alaska

See Also

mogi1, mogiM

Examples

```
library(RPMG)

EX = seq(from=-10, to=10, length=50)
WHY = seq(from=-10, to=10, length=50)
M = RPMG::meshgrid(EX, WHY )

OKout  = okada85(M$x, M$y,2,30,70,5,3,-45,1,1)
old.par <- par(no.readonly = TRUE)

par(mfrow=c(2, 2))

image(EX, WHY, OKout$uN, col=terrain.colors(100) ,
asp=1, main="uN" );contour(EX, WHY, OKout$uN, add=TRUE)

image(EX, WHY, OKout$uE, col=terrain.colors(100) ,
asp=1 , main="uE" );contour(EX, WHY, OKout$uE, add=TRUE)

image(EX, WHY, OKout$uZ, col=terrain.colors(100) ,
asp=1, main="uZ" );contour(EX, WHY, OKout$uZ,  add=TRUE)

plot(EX, OKout$uZ[, 25 ])


par(old.par)
```

---

perpproj *Perpendicular Projection*

Description

Perpendicular Projection in two-dimensions

Usage

```
perpproj(V1, V2, add = FALSE)
pointproj(P1, VEC )
```

Arguments

V1	vector 1
V2	vector 1
add	logical, TRUE add to plot
P1	point coordinates,
VEC	vector coordinates,

Details

V1 and V2 are arrays with, for example, V1=c(x1 , x2 , y1, y2) or lists with V2=list(x=c(x1, x2), y-c(y1, y2)) which gets converted into the first format.

The points are from the head of each vector projected onto the the line formed by the other vector. If the the x1 y1 of each vector do not coincide, the interesections of two lines is determined and the tail of each vector is moved to that point for determining the projection positions.

for the pointproj the input can be a 2-element vector or a list with (x,y) elements. The VEC should be organized, as (x1,x2, y1, y2) or as an (x,y) list. The P1 vector can include many points, so a large number of projections are doable.

Value

list:

P1	x, y
P2	x, y

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

vecproj

Examples

```
V1 = c( 2,  6, 1, 9)
  V2 = c( 0, 5,  1, 2)


  PP = perpproj( V1, V2, add=FALSE  )
```

```
R = range(c(V1, V2, unlist(PP) ))

plot(R, R, type='n', asp=1)

arrows(V1[1], V1[3], V1[2], V1[4], length=.08 )
arrows(V2[1], V2[3], V2[2], V2[4], length=.08 )

points(PP$P1[1],PP$P1[2], col='red')
   points(PP$P2[1],PP$P2[2], col='blue')



arrows(V2[2], V2[4],PP$P1[1],PP$P1[2] , length=.08, col='red' )

arrows(V1[2], V1[4], PP$P2[1],PP$P2[2], length=.08, col='blue' )
```

PLOTbox *Plot 3D box*

Description

Plot 3D box

Usage

```
PLOTbox(Rax, Rbox, axcol = "black", boxcol = "blue")
```

Arguments

Rax	rotated axes
Rbox	rotated box
axcol	axes color
boxcol	box color

Value

Graphical Side effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

pstart, PLOTplane, stress, NORMvec

Examples

```
#######    set colors
 axcol = 'black'
 boxcol = 'blue'
 planecol = 'brown'

####   view angle
 Rview  =    RFOC::ROTZ(-130)

###   set arbitrary scale
    xscale = 100


###   create box  and scale it
   BOX <-matrix(c(0,0,0,0,
            0, 1, 0,0,
            0, 1, 1,0,
            0, 0, 1,0,
            1,0,0,0,
            1, 1, 0,0,
            1, 1, 1,0,
            1, 0, 1,0), ncol=4, byrow=TRUE)


    BOX = xscale*BOX

    ##############   create axes and scale them
AX = matrix(c(0,0,0,0,
            1, 0, 0,0,
            0, 0, 0,0,
            0, 1, 0,0,
            0,0,0,0,
            0, 0, 1,0), ncol=4, byrow=TRUE)

    AX = 1.5*xscale*AX

 ##############   rotate axes and box
    Rax =  AX


          Rbox =   BOX

 ##############  start the figure
pstart(xscale=xscale)
 ######  plot the box
PLOTbox(Rax, Rbox, axcol= 'black', boxcol= 'blue')
```

`plotfry` *Plot Fry Output*

Description

Plot Fry Output limited by a specified distance.

Usage

```
plotfry(fry, dis, col = grey(0.7), ann = FALSE, axes = FALSE)
```

Arguments

`fry`	list output of dofry
`dis`	distance to limit plotting to the center
`col`	color for points
`ann`	logical, annotation of axes, see par
`axes`	logical, whether to plot the axes, see par

Details

Used for plotting the points attained through the fry method

Value

`x`	x-coordinates
`y`	y-coordinates
`mx`	x center of plot
`my`	y center of plot
`dis`	distance for limiting plot

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Fry, N., (1979) <http://dx.doi.org/10.1016/0040-1951(79)90135-5> Random point distributions and strain measurement in rocks Tectonophysics, 60:89-105.

See Also

dofry, Showfry

Examples

```
RDAT = randFRY(400, LIM=c(0,0, 200, 200) , rlen=5   )
length(RDAT$x)
plot(RDAT$x, RDAT$y, asp=1, pch=".", cex=2)
flag = sqrt( (RDAT$x-mean(RDAT$x))^2  + (RDAT$y-mean(RDAT$y))^2)<75

DAT = list(x=RDAT$x[flag], y=RDAT$y[flag])
 x = DAT$x
    y = DAT$y
 plot(x,y, asp=1, pch=21, col='red', bg='gold', ann=FALSE, axes=FALSE)

FF = dofry(DAT$x, DAT$y )
AF = plotfry(FF, dis=30)

Z = xtractlip(AF)

lines(cluster::predict.ellipsoid(Z$hull), col='red')
```

PLOTplane *Plot Plane on box*

Description

Plot Plane on box

Usage

```
PLOTplane(Rp, planecol = "brown")
```

Arguments

`Rp`	Rotated points
`planecol`	color for plotting plane

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

pstart, PLOTbox, stress

Examples

```
#######    set colors
 axcol = 'black'
 boxcol = 'blue'
 planecol = 'brown'

####   view angle
 Rview  =    RFOC::ROTZ(-130)

###   set arbitrary scale
    xscale = 100


####   set three points and combine them with rbind
     P1 = xscale*c(.2, 1,1,0)
    P2 = xscale*c(1, .1,1,0)
    P3 = xscale*c(1, 1,.4,0)

    PPs = rbind(P1, P2, P3)

  ###  convert points for view
Rp = PPs

###   create box  and scale it
   BOX <-matrix(c(0,0,0,0,
            0, 1, 0,0,
            0, 1, 1,0,
            0, 0, 1,0,
            1,0,0,0,
            1, 1, 0,0,
            1, 1, 1,0,
            1, 0, 1,0), ncol=4, byrow=TRUE)


    BOX = xscale*BOX

    ##############   create axes and scale them
AX = matrix(c(0,0,0,0,
            1, 0, 0,0,
            0, 0, 0,0,
            0, 1, 0,0,
            0,0,0,0,
            0, 0, 1,0), ncol=4, byrow=TRUE)

    AX = 1.5*xscale*AX

 ##############   rotate axes and box
    Rax =  AX


          Rbox =   BOX
```

```
 ##############  start the figure
pstart(xscale=xscale)
 ######  plot the box
PLOTbox(Rax, Rbox, axcol= 'black', boxcol= 'blue')
 ##############  plot the plane
PLOTplane(Rp)

     P1 = xscale*c(0, 0,.4,0)
    P2 = xscale*c(0.8, 0,0,0)
    P3 = xscale*c(0, .7, 0 ,0)

    PPs = rbind(P1, P2, P3)

  ###  convert points for view
Rp = PPs

PLOTplane(Rp, planecol = "gold" )
```

---

points2line *Points to Line*

Description

Given a point and a matrix of lines, calculate the projection of the vector of the point to the first coordinate of each line and the perpendicular distance.

Usage

```
points2line(Lp, VL)
```

Arguments

Lp	Point list (x,y)
VL	Matrix of points, N by 4 as X1, Y1, X2, Y2

Details

The first point in the matrix is considered the base.

Value

list:

rat	cosine projection
srat	sine projection

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

REplane

Examples

```
S= stressSETUP()
      pstart()
PLOTbox(S$Rax, S$Rbox, axcol= 'green', boxcol= 'purple')
      PLOTplane(S$Rp, planecol="brown")

 basepoint = 3

        legpoints = c(7,4,2)



        VL =   cbind( rep(S$Rbox[basepoint,1] , length(legpoints)),
     rep(S$Rbox[basepoint,2] , length(legpoints)),
          S$Rbox[legpoints,1], S$Rbox[legpoints,2])

      Lp=list()
Lp$x=c(40.0180732557)
Lp$y=c(40.4847345741)

 G = points2line(Lp, VL )
```

PolarDecomp *Polar Decomposition*

Description

Polar Decomposition for Strain

Usage

```
PolarDecomp(A)
```

Arguments

A	Strain Matrix

Details

Polar decomposition uses the svd to extract 2 matrices that represent the stretch and rotation of a strain: A = UP. U is orthogonal rotation matrix and P is the stretch tensor. These are extracted from the singular value decomposition.

Value

P	matrix, stretch tensor
U	matrix, orthogonal rotation matrix

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

<http://en.wikipedia.org/wiki/Finite_strain_theory>

See Also

svd

Examples

```
A = matrix(runif(4, -1, 1), ncol=2)
PD = PolarDecomp(A)

E = svd(A)

###  W         S           V
 E$u

 ###t(E$v) %*% diag(E$d)%*%  (E$u)

P = E$v
U =  E$u


U
```

PPoints *Igneous Petrology Points*

Description

Scanned points from igneous petrology

Usage

```
data(PPoints)
```

Format

The format is: List of 4 $ x : num [1:16] 45.4 56.4 64.1 71.1 66.3 ... $ y : num [1:16] 9.39 13.56 11.36 8.36 5.84 ... $ lab : chr [1:16] "minor varieties" "phonolites" "trachytes" "rhyolites" ... $ rots: num [1:16] 45 0 0 0 0 0 0 -90 0 0 ...

Details

Can click on here with a mixture to get a location on the plot.

Source

scanned figure

References

Principles of Igneous and Metamorphic Petrology, A. R. Philpotts, 1990, Prentice Hall <ISBN-13:978-0521880060>.

Examples

```
data(Glines)
data(PPoints)

plot(Glines[[1]], type='n', xlab="SiO2", ylab="NA2O+K2O")
for(i in 1:length(Glines))
{

lines(Glines[[i]]$x, Glines[[i]]$y)

}

for(i in 1:length(PPoints$x))
{
text(PPoints$x[i],PPoints$y[i], label=PPoints$lab[i], srt=PPoints$rots[i])
}
```

`pstart` *Start an empty plot*

Description

Start an empty plot

Usage

```
pstart(xscale = 30, expand=1.2)
```

Arguments

`xscale`	Arbitray scale for plot
`expand`	factor to expand the plotting region, default=1.2

Details

Creates a plotting region with no drawing. Default region is defined by: xlim= c(-expand*xscale, expand*xscale) and ylim= c(-expand*xscale, expand*xscale).

Value

Graphical Side effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

Examples

```
pstart()

axis(1)
axis(2)
```

PXY *Deformation of a volcano*

Description

Deformation of Pozzuoli Italy 1982

Usage

```
data(PXY)
```

Format

The format is: List of 2 $ x: num [1:75] 0.0237 0.3478 0.5252 0.8689 0.8387 ... $ y: num [1:75] 1 0.991 0.975 0.928 0.962 ...

Details

Used as example data for Mogi Source inversion.

Source

Digitized from publication listed in reference.

References

publication

Berrino, G., Corrado, G., Luongo, G., and Toro, B., Ground deformation and gravity changes accompanying the 1982 Pozzuoli Uplift, in Monograph:On the 1982-1984 bradyseismic crisis at Phlegraean Fields (Italy) eds: Barberi, F., Hill, D P., Innocenti, F., Luongo, G., Treuil, M., Bulletin Volcanologique, vol.47, no.2, special issue, pp.187-200, 1984 <doi:https://doi.org/10.1007/BF01961548>.

Examples

```
data(PXY)

plot(PXY)
```

randFRY *Random data for Fry analysis*

Description

Random distribution of points that have a specified radial spacing.

Usage

```
randFRY(N = 200, LIM = c(0, 0, 100, 100), rlen = 5, PLOT=FALSE)
```

Arguments

N	integer, number of points
LIM	Limit in x-y direction
rlen	distance length. points are selected if they are greater than this distance from each other.
PLOT	logical, TRUE=make a plot of data

Details

A spatial data set is generated that has random distribution with points no closer than a specified distance apart.

Value

x	x-coordinate
y	y-coordinate

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Fry, N., (1979) <http://dx.doi.org/10.1016/0040-1951(79)90135-5> Random point distributions and strain measurement in rocks Tectonophysics, 60:89-105.

See Also

dofry

Examples

```
RDAT = randFRY(100, LIM=c(0,0, 100, 100) , rlen=5   )
length(RDAT$x)
plot(RDAT$x, RDAT$y, asp=1, pch=".", cex=2)
```

`randpoles` *Random Poles*

Description

Create a set of random poles distributed around a given pole.

Usage

```
randpoles(az, iang, alphadeg, opt = "unif", BALL.radius = 1, N = 10, add = TRUE, ...)
```

Arguments

`az`	azimuth of given direction
`iang`	inclination angle of given direction
`alphadeg`	standard error for distribution
`opt`	Optional distribution, "unif" or "norm"
`BALL.radius`	radius of small circle to draw
`N`	Number of points to create
`add`	logical, TRUE=add points to net plot
`...`	graphical parameters for plotting

Details

A given geographic pole is supplied and N normally or uniformly distributed poles are created.

Value

`az`	azimuths of new poles
`dip`	dips of new poles
`x`	x position on plot
`y`	y position on plot

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

net

Examples

```
RFOC::net();
GG = randpoles(30, 40, 10, opt="norm", N=20)

RFOC::addsmallcirc(30, 40, 10)
```

`REplane` *Replot plane*

Description

Replot the plane after points have moved

Usage

```
REplane(m1, Lp, PPs, Rbox, Rview, xscale)
```

Arguments

`m1`	axis chosen
`Lp`	x,y location to move to.
`PPs`	Points matrix
`Rbox`	Box matrix
`Rview`	Viewing matrix
`xscale`	arbitrary scale

Details

Used internally in stress program. The new locations are constrained to lie on associated axis.

Value

graphical side effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

stress

---

rev2RH *Reverse to Right Hand*

Description

Reverse orientation of the vectors to a right handed polygon.

Usage

```
rev2RH(pol)
```

Arguments

pol	List with x and y components

Details

program reverses the order of the x and y components so they traverse in the oposite direction.

Value

list

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

dircheck, flipZEE

Examples

```
L=list()
L$x=c( 0.2188,-0.6668,-1.0992,-0.5105, 0.5106)
L$y=c( 0.76822, 0.75292,-0.01741,-0.74692,-0.48165)
rev2RH(L)
```

`rot2Zplane` *Rotate to Z-plane*

Description

Rotation matrix to project a set of 3D points to z-plane

Usage

```
rot2Zplane(vec, p)
```

Arguments

vec	3D normal vector
p	translation from this point

Details

provides a matrix to rotate and translate a set of points to the X-Y plane. Used for internal calculations

Value

Matrix for projection

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

gmat

Examples

```
P1=runif(3)
P2=runif(3)
P3=runif(3)

PP = rbind(P1, P2, P3, rep(1, 3) )

    g1 = PP[1, 1:3] - PP[3, 1:3]
    g2 = PP[2, 1:3] - PP[3, 1:3]


 B = unlist(AXB.prod(list(x = g1[1], y = g1[2], z = g1[3]),
        list(x = g2[1], y = g2[2], z = g2[3])))
```

```
    B = B/sqrt(sum(B * B))

MF = rot2Zplane(B, P3)

UMAT = t(PP)
```

`Sect2vex` *Intersect 2 Vectors*

Description

Finder intersection point of two vectors in a plane

Usage

```
Sect2vex(V1, V2)
```

Arguments

`V1`	vector of 2 points
`V2`	vector of 2 points

Details

if vectors are parallel return NULL

Value

x-y location of intersection

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

AXB.prod

Examples

```
v1 = matrix(runif(4), ncol=2)
v2 =  matrix(runif(4), ncol=2)


pt = Sect2vex(v1, v2)

plot(c(v1[,1],v2[,1],pt[1] ) , c(v1[,2],v2[,2],pt[2]) , asp=1)

arrows(v1[1,1], v1[1,2], v1[2,1], v1[2,2])
arrows(v2[1,1], v2[1,2], v2[2,1], v2[2,2])

points(pt[1], pt[2], col='red', cex=2, pch=3)
```

`setSTRESS` *Set Stress Tensor*

Description

Set Stress Tensor: given either 3-vector, or 6-vector, create the symmetric stress tensor matrix.

Usage

```
setSTRESS(Stensor)
```

Arguments

`Stensor`	stress tensor as a vector or a 3 by 3 matrix, a 3-vector (principal stresses) or a 6-vector (principal and off diagonal in row-order.

Details

Can be either three components of diagonal, or six components of the stress tensor. If the number of elements is neither 3 nor 6, returns NULL and an error message.

Value

List: eigen value decomposition

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

Examples

```
ES = setSTRESS(c(12, -6, 3) )

ES = setSTRESS(c(12, -6, 3, 3, 6, 1) )
```

Showfry *Show Fry Diagrams*

Description

Show Fry Diagrams with random data

Usage

```
Showfry(RDAT, shear = matrix(c(1, 1.2, 0, 1)), rad = 75)
```

Arguments

RDAT	data list
shear	2D shearing matrix
rad	radius of points from the center from which to select the points for analysis.

Details

The original data is deformed and a circular subset is extracted for analysis. This is to show that the shape of the data does not affect the assessment of the fry ellipse. Plots input data and fry diagram, side by side.

Value

graphical side effects

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Fry, N., (1979) <http://dx.doi.org/10.1016/0040-1951(79)90135-5> Random point distributions and strain measurement in rocks Tectonophysics, 60:89-105.

See Also

dofry, plotfry, xtractlip

Examples

```
RDAT = randFRY(400, LIM=c(0,0, 200, 200) , rlen=5   )
length(RDAT$x)
plot(RDAT$x, RDAT$y, asp=1, pch=".", cex=2)

u = par( no.readonly = TRUE)


par(mfrow=c(3,2))
    shr = 0.0
simpleshear = matrix(c(1, shr, 0,  1), ncol=2)

Showfry(RDAT, simpleshear, 75)
    shr = 1.2
simpleshear = matrix(c(1, shr, 0,  1), ncol=2)

Showfry(RDAT, simpleshear, 75)

epsilon1 = 0.4
H = matrix(c(1+epsilon1, 0, 0,  1/(1+epsilon1) ), ncol=2)

Showfry(RDAT, H, 75)

par(u)
```

---

stress *Stress Box*

Description

Display interactive Stress Box illustrating normal and shear stress on an arbitrary plane given a stress tensor.

Usage

```
stress(PPs = matrix(ncol = 4, nrow = 3), Rview = c(-130, -50),
xscale = 100, Stensor = matrix(ncol = 3, nrow = 3))
```

Arguments

PPs	3Points in 3 by 4 matrix
Rview	Viewing matrix

xscale	arbitrary scale
Stensor	stress tensor

Details

Interactive program to examine stress. To change the plane orientation, click on one of the points and then on a new position along the axis selected.

If a stress tensor is provided, then a second graphics device is required to show the Mohr's circle.

Value

Graphical Side effects

Note

Uses RPMG for interaction.

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

pstart, PLOTbox, PLOTplane, NORMvec

Examples

```
## Not run:
#####   no stress tensor provided => no mohr's circle
stress()

#########################  run program with a stress tensor
Stensor = matrix(c(
15, 0, 0,
0, 10, 0,
0,  0, 5), ncol=3)


stress(Stensor=Stensor)

P1 = c(0.2, 1, 1, 0)
P2 = c(1, 0.1, 1, 0)
P3 = c(1, 1, 0.4, 0)

  S = stressSETUP(P1, P2, P3, xscale=30   )

stress(PPs = S$PPs, Rview =S$Rview,
     xscale = S$xscale, Stensor=Stensor )
```

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

`stressSETUP` *Setup Stress Box*

Description

Setup default stress box, plane information and normal vector 3d plotting glyph

Usage

```
stressSETUP(P1=c(.2, 1,1,0 ), P2=c(1, .1,1,0), P3=c(1, 1,.4,0), xscale=30)
```

Arguments

	Arguments are optional
	4-vector, point 1
`P2`	4-vector,point 2
`P3`	4-vector,point 3
`xscale`	scale factor for plotting

Details

Initialize the stress set up.

Value

List:

`xscale`	arbitrary scale
`Rview`	viewing rotation matrix
`BOX`	Box coordinates, scaled
`AX`	XYZ Axis coordinates
`Rbox`	rotated box
`Rax`	XYZ rotated axes
`PPs`	Points along the Box axes
`Rp`	rotated points
`aglyph`	3D glyph for plotted norm

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

stress, PLOTbox, PLOTplane, NORMvec

Examples

```
S = stressSETUP()
pstart()

PLOTbox(S$Rax, S$Rbox, axcol= 'green', boxcol= 'purple')

pstart()

PLOTplane(S$Rp, planecol="brown")
PLOTbox(S$Rax, S$Rbox, axcol= 'green', boxcol= 'purple')


NORMvec(S$PPs, S$xscale, S$Rview, S$aglyph, add=TRUE)
```

---

tauline *Shear Stress along Line*

Description

Calculate the shear stress along an arbitrary line in a plane with stress orientation

Usage

```
tauline(Rp, P1, P2, Rview, ES, NN)
tauplane(Rp, L, Rview, ES, NN)
```

Arguments

Rp	rotated points describing plane
P1	point 1 extracted from screen (locator)
P2	point 2 extracted from screen
Rview	rotation matrix for viewing
ES	eigen value decomposition from eigen
NN	normal vector to plan in unrotated coordinates
L	list locations (x,y) in the figure, projected to the plane

Details

Used internally in stress. When the plan is plotted, if two points are located on the figure, the points are positions on the plan and un-rotated using the Rview matrix. Then the shear stress in the plan along that line is calculated and returned.

Value

shear stress along the line indicated

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

stress,NORMvec

Examples

```
S= stressSETUP()


pstart()

     PLOTplane(S$Rp, planecol="brown")
     PLOTbox(S$Rax, S$Rbox, axcol= 'green', boxcol= 'purple')




##  L = locator(2)

L=list()
L$x=c(-13.6305297057, 52.6412739525)
L$y=c(26.2697350325,32.4501696158)


Stensor = matrix(c(
15, 0, 0,
0, 10, 0,
0,  0, 5), ncol=3)


P1 = list(x=L$x[1], y=L$y[1])
P2 = list(x=L$x[2], y=L$y[2])


ES = eigen(Stensor)
NN = NORMvec(S$PPs, S$xscale, S$Rview, aglyph=S$aglyph, add=FALSE)
```

```
tauline(S$Rp, P1, P2, S$Rview, ES, NN)
```

`TriangleCenter` *Triangle Center*

Description

Extract Triangle center in 3D

Usage

```
TriangleCenter(P1, P2, P3, A1= 0, A2= 360, KNum=10)
```

Arguments

`P1`	3-vector, point(x,y,z)
`P2`	3-vector, point(x,y,z)
`P3`	3-vector, point(x,y,z)
`A1`	degrees, initial angle in plane(default=0)
`A2`	degrees, final angle in plane(default=360)
`KNum`	Divisor Number to divide range by (default=10)

Details

Program rotates the object to the X-Y plane and does calculations in 2D, then rotates back.

Value

`Center`	x-y of center of the inscribed circle
`r`	radius of inscribed
`Cinscribed`	inscribed circle points around center
`CIRCUM`	x-y of center of the circumscribed circle

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

TriangleInfo

Examples

```
S = stressSETUP()
pstart()

PLOTbox(S$Rax, S$Rbox, axcol= 'green', boxcol= 'purple')

pstart()

PLOTplane(S$Rp, planecol="brown")
PLOTbox(S$Rax, S$Rbox, axcol= 'green', boxcol= 'purple')


NORMvec(S$PPs, S$xscale, S$Rview, S$aglyph, add=TRUE)
   P1 = S$PPs[1, 1:3]
    P2 = S$PPs[2, 1:3]
    P3 = S$PPs[3, 1:3]

BV = TriangleCenter(S$PPs[1,1:3],S$PPs[2,1:3], S$PPs[3,1:3] )

 CIRCview =   BV$Cinscribed

    lines(CIRCview[,1], CIRCview[,2], col='purple')

cview =   BV$Center

    points(cview[1,1], cview[1,2])
```

`TriangleInfo` *Triangle Information*

Description

Given a triangle (three non-colinear points, return info that is useful

Usage

```
TriangleInfo(P1, P2 = c(0, 1), P3 = c(1, 0), add = FALSE)
```

Arguments

P1	vector or list(x,y) of 2D point 1
P2	vector or list(x,y) of 2D point 2
P3	vector or list(x,y) of 2D point 3
add	logical, if TRUE add a variety of lines to the plot

Details

If either P1 P2 or P3 are an x,y list, they are converted to vectors.

If P2 and P3 are missing, P1 contains all the points. Input can be list of 3 x,y values, a 3 by 2 matrix .

Value

list

BI	x,y location of center of inscribed circle
CIRCUM	x,y location of center of circumscribed circle
IH	x,y location of Intersection of bisectors
CEN	centroid location
r	radius of inscribed circle
R	radius of circumscribed circle
AngBis	loci of vectors bisecting each corner on opposite leg
H	lengths of altitudes
M	lengthsof medians
TEE	lengths of perpendiculars
Area	Area of triangle

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

<http://en.wikipedia.org/wiki/Triangle_center>

<http://en.wikipedia.org/wiki/Encyclopedia_of_Triangle_Centers>

See Also

Intersect2points, Sect2vex, AXB.prod, TriangleCenter, rot2Zplane

Examples

```
P1 = runif(2)
P2 = runif(2)
P3 = runif(2)



theX = c(P1[1],P2[1], P3[1])
theY = c(P1[2],P2[2], P3[2])
```

```
        plot(theX , theY, asp=1)


        points(theX , theY)
        text(theX , theY, labels=1:3, pos=3)
        lines(c(theX, theX[1]), c(theY, theY[1]) )

TriangleInfo(P1, P2, P3, add=TRUE)
#####   TriangleInfo(rbind(P1, P2, P3),   add=TRUE)
```

---

vecproj *Vector Projection*

Description

Vector Projection information, such as angle and distances between points

Usage

```
vecproj(P1, P2)
```

Arguments

P1	Point 1
P2	Point 2

Details

The distances returned are the legs of right right triangles where the cosine of the angle is used to get the projection distance of the opposite side on the specified direction. See the example for an illustration.

Value

cang=cang, angrad=angrad, angdeg=angdeg, dis1=d1, dis2=d2

cang	cosine of angle between points
angrad	angle, radians
angdeg	angle, degrees
dis1	distance
dis2	distance

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

See Also

perpproj

Examples

```
P1 = c(2, 3)
P2 = c(5, 2)


I = vecproj(P1, P2)


plot(c(0, P1[1], P2[1]), c(0, P1[2], P2[2]), asp=1, ann=FALSE)

arrows(0, 0, P1[1],  P1[2], length=.1)
arrows(0, 0,  P2[1],  P2[2], length=.1)

text(P1[1],  P1[2], "Point 1", pos=3)
text(P2[1],  P2[2], "Point 2", pos=3)

j1 = atan2(P1[2], P1[1])*180/pi
j2 = atan2(P2[2], P2[1])*180/pi

L1 = vlength(P1)
L2 = vlength(P2)

A = GEOmap::darc(L1*.2, j1, j2, 0, 0, n=1)

lines(A)
an = length(A$x)
arrows(A$x[an-1] , A$y[an-1] ,A$x[an] , A$y[an] ,  length=.08 )

text(A$x[an/2] , A$y[an/2], labels=format(I$angdeg, digits=4)  , pos=4)


 V1 = c( 0,P1[1], 0,  P1[2])
   V2 = c( 0,P2[1], 0,  P2[2])


    PP = perpproj( V1, V2, add=FALSE  )

arrows(P1[1],P1[2],PP$P2[1], PP$P2[2],  length=.07, lty=2, col='red')
arrows(P2[1],P2[2],PP$P1[1], PP$P1[2],  length=.07, lty=2, col='blue')

 labelLine( c(0, 0) , PP$P2 , lab="dis1", dinch = .25, aty=1,
acol='blue', above=FALSE  )
```

```
 labelLine( c(0, 0) , PP$P1 , lab="dis2", dinch = .25, aty=1,
acol='blue'  )
```

vlength *vector length*

Description

calculate euclidian vector length

Usage

```
vlength(a)
```

Arguments

a	vector

Value

Euclidian Length

Author(s)

Jonathan M. Lees<jonathan.lees.edu>

Examples

```
vlength(c(23, 43))
```

`xtractlip` *Extract Convex Hull Ellipse from fry*

Description

Extract Convex Hull Ellipse from fry-diagram

Usage

```
xtractlip(AF)
```

Arguments

AF	x-y List out put of plotfry

Details

To get closest points to center, program cycles around in a circular manner pulling out the closest points. These points are used to estimate the chull ellipse.

Value

List: see output of ellipsoidhull: an object of class '"ellipsoid"', basically a 'list' with several components

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

References

Fry, N., (1979) <http://dx.doi.org/10.1016/0040-1951(79)90135-5> Random point distributions and strain measurement in rocks Tectonophysics, 60:89-105.

See Also

ellipsoidhull

Examples

```
RDAT = randFRY(400, LIM=c(0,0, 200, 200) , rlen=5   )
length(RDAT$x)
plot(RDAT$x, RDAT$y, asp=1, pch=".", cex=2)
flag = sqrt( (RDAT$x-mean(RDAT$x))^2  + (RDAT$y-mean(RDAT$y))^2)<75

DAT = list(x=RDAT$x[flag], y=RDAT$y[flag])
 x = DAT$x
    y = DAT$y
```

```
 plot(x,y, asp=1, pch=21, col='red', bg='gold', ann=FALSE, axes=FALSE)

FF = dofry(DAT$x, DAT$y )
AF = plotfry(FF, dis=30)

Z = xtractlip(AF)

lines(cluster::predict.ellipsoid(Z$hull), col='red')
```

Index