Package 'RGPR'

April 13, 2018

Type Package

Title Ground-penetrating radar (GPR) data visualisation, processing and delineation

Version 0.0.0

Date 2017-10-24

Author Emanuel Huber and Guillaume Hans

Maintainer Emanuel Huber <emanuel.huber@alumni.ethz.ch>

Description S4 classes and R functions to read, write, analyse and visualise ground-penetrating radar (GPR) data.

Depends base

License GPL (>= 2)

URL https://github.com/emanuelhuber/RGPR

BugReports https://github.com/emanuelhuber/RGPR/issues

LazyData true

Collate 'RGPR.R'
'global.R'
'ClassGPR.R'
'ClassGPRsurvey.R'
'ClassMisc.R'
'data.R'

Imports methods,
graphics,
stats,
grDevices,
utils,
sp,
rgdal,
rgeos,
MASS,
signal,
colorspace,

Cairo,
rgl,
mmand,
e1071,
plot3D,
adimpro,
raster,
fields

RoxygenNote 6.0.1

R topics documented:

`RGPR-package` *RGPR: A package for processing and visualising ground-penetrating data radar (GPR) data.*

Description

The RGPR package provides two classes GPR and GPRsurvey

Reading/writing/export functions

- `readGPR()`: format DT1 (Sensors&Software), rds (R-format)
- `writeGPR()`: format DT1 (Sensors&Software), rds (R-format)
- `exportPDF()`
- `exportDelineations()`
- `exportFid()`: ASCII-file
- `exportCoord()`: ASCII, SpatialLines or SpatialPoints
- `exportProc()`: ASCII-file

Plot functions

- plot(): raster or wiggles.
- plot3D():
- plotAmpl()
- plotDelineations()

Coercion

- as.matrix():
- as.numeric():
- as.list():
- as.SpatialPoints():
- as.SpatialLines():

Delineation

- delineate():
- plotDelineations():
- delineations(): list of the delineations
- addDelineation:
- rmDelineations:
- exportDelineations:
- plotDelineations3D:
- identifyDelineation:

Author(s)

Maintainer: Emanuel Huber <emanuel.huber@alumni.ethz.ch>

Authors:

- Guillaume Hans <guillaume.hans@fpinnovations.ca>

References

Several books!

See Also

Useful links:

- https://github.com/emanuelhuber/RGPR
- Report bugs at https://github.com/emanuelhuber/RGPR/issues

ampl *Amplitude of the GPR data*

Description

Amplitude of the GPR data

Usage

```
## S4 method for signature 'GPR'
ampl(x, FUN = mean, ...)
```

ann *Annotations of the GPR data*

Description

Annotations of the GPR data

Usage

```
## S4 method for signature 'GPR'
ann(x)

## S4 replacement method for signature 'GPR'
ann(x) <- value
```

Arith,GPR,ANY-method *Basic arithmetical functions*

Description

Basic arithmetical functions

Usage

```
## S4 method for signature 'GPR,ANY'
Arith(e1, e2)

## S4 method for signature 'GPR,GPR'
Arith(e1, e2)

## S4 method for signature 'ANY,GPR'
Arith(e1, e2)
```

Arguments

e1	An object of the class RGPR.
e2	An object of the class RGPR.

Examples

```
data(frenkeLine00)
A <- exp(frenkeLine00)
B <- A + frenkeLine00
```

---

as.matrix *Coercion to matrix*

Description

Coercion to matrix

Coercion to vector

Coercion to SpatialLines

Coercion to SpatialPoints

Coercion to numeric

Coercion from matrix to GPR

Coercion from list to GPR

Usage

```
## S4 method for signature 'GPR'
as.matrix(x)

## S4 method for signature 'GPR'
as.vector(x, mode = "any")

## S4 method for signature 'GPR'
as.SpatialLines(x)

## S4 method for signature 'GPR'
as.SpatialPoints(x)

## S4 method for signature 'GPR'
as.numeric(x, ...)

as.GPR.matrix(x, ...)

as.GPR.list(x, ...)
```

clip *Clip the amplitude*

Description

Clip the amplitude

Usage

```
## S4 method for signature 'GPR'
clip(x, Amax = NULL, Amin = NULL)
```

CMPAnalysis *Common mid-point (CMP) analysis*

Description

either use 'rec' and 'trans' to compute the distance between the antennas or give the distance between the antennas (asep) or seq(x@antsep, by = x@dx, length.out = length(x))

Usage

```
## S4 method for signature 'GPR'
CMPAnalysis(x, method = c("semblance", "winsemblance",
  "wincoherence", "wincoherence2"), v = NULL, asep = NULL, w = NULL)
```

Arguments

x	An object of the class GPR
method	A length-one character vector
v	A numeric vector defining at which velocities the analysis is performed
asep	A length-n numeric vector defining the antenna separation for each trace (n = number of traces)
w	A length-one numeric vector defining the window length for the methods 'wincoherence' and 'wincoherence2'.

Details

semblance also described as the ratio of input to output energy (Niedell and Taner, 1971)

semblance2 windowed semblance

wincoherence Windowed coherence measure based on eigen-decomposition that estimates the signal-to-noise ratio for high resolution velocity analysis (Sacchi, 2002)

wincoherence2 Windowed coherence measure based on a log-generalized likelihood ratio which tests the hypothesis of equality of eigenvalues (Key and Smithson, 1990)

References

- Neidell and Taner (1971) Semblance and other coherency measures for multichannel data. Geophysics, 36(3):482-497.
- Key and Smithson (1990) New approach to seismic-reflection event detection and velocity determination. Geophysics, 55(8):1057-1069.
- Textbook: Sacchi (2002) Statistical and Transform Methods in Geophysical Signal Processing

---

`colFromPal` *Return color from palette*

Description

Return color from palette

Usage

```
colFromPal(A, col = palGPR(n = 101))
```

`conv1D` *Trace convolution (1D)*

Description

Convolution of the GPR traces with a wavelet

Usage

```
## S4 method for signature 'GPR'
conv1D(x, w)
```

Arguments

`x`	A GPR data
`w`	A numeric vector defining a wavelet or a matrix with number of columns equal to the number of traces.

Value

The convolved GPR data.

conv2D *2D onvolution*

Description

Convolution of the GPR data with a kernel

Usage

```
## S4 method for signature 'GPR'
conv2D(x, w)
```

Arguments

x	A GPR data
w	A numeric matrix with smaller dimension than the GPR data.

Value

The convolved GPR data.

convolution *Linear convolution based on FFT*

Description

If A (or B) is a numeric vector, it is converted into a one-column matrix. Then if A and B do not have the same number of column, then the first column of the matrix with the smallest number of column is repeated to match the dimension of the other matrix. match the dimension of the other matrix.

Usage

```
convolution(A, k)
```

Arguments

A	A numeric vector or matrix.
A	B numeric vector or matrix.

convolution2D *Two-dimensional convolution*

Description

The convolution is performed with 2D FFT

Usage

```
convolution2D(A, k)
```

coord *Coordinates of the GPR data*

Description

Coordinates of the GPR data

Usage

```
## S4 method for signature 'GPR'
coord(x, i, ...)

## S4 replacement method for signature 'GPR'
coord(x) <- value
```

coordref *Define a local reference coordinate*

Description

Define a local reference coordinate

Usage

```
## S4 method for signature 'GPRsurvey'
coordref(x)
```

coords *Return coordinates*

Description

Return coordinates

Set coordinates

Usage

```
## S4 method for signature 'GPRsurvey'
coords(x, i)

## S4 replacement method for signature 'GPRsurvey'
coords(x) <- value
```

crs *Coordinate reference system (CRS) of the GPR data*

Description

Coordinate reference system (CRS) of the GPR data

Usage

```
## S4 method for signature 'GPR'
crs(x)

## S4 replacement method for signature 'GPR'
crs(x) <- value

## S4 method for signature 'GPRsurvey'
crs(x)

## S4 replacement method for signature 'GPRsurvey'
crs(x) <- value
```

dcshift *Direct-Current shift removal*

Description

The direct-current shift is estimated for each traces based on a specified number of time samples (normally the samples before time-zero). Then, the direct-current shift of every trace is substracted from every trace.

Usage

```
## S4 method for signature 'GPR'
dcshift(x, u = 1:10, FUN = mean)
```

Arguments

x	An object of the class 'GPR'.
u	Number of time samples used to evaluate the DC-shift.
FUN	A function to apply on the first 'u' time samples (default is 'mean'; alternatively 'median' could be used or any user defined function).
n	[integer(1)] My argument. Default is 1.

deconv *Deconvolution*

Description

A generic function to perform different types of convolution

Usage

```
## S4 method for signature 'GPR'
deconv(x, method = c("spiking", "wavelet", "min-phase",
  "mixed-phase"), ...)
```

Arguments

method	Type of deconvolution method.
...	additional arguments, see details.

Value

A list containing the deconvolued GPR data (and possibly other variables.

Spiking and mixed-phase deconvolution

The required arguments for `method = "spiking"` and `method = "mixed-phase"` are:

- `W`: A length-two numeric vector defining the time/depth window for which the wavelet is estimated
- `wtr`: A length-one numeric vector defining the number of neighorough traces to be combine into a "super trace" (the total number of traces is `2*wtr + 1`).
- `nf`: A length-one numeric vector defining the filter length.
- `mu`: A length-one numeric vector defining the amount of noise.

Wavelet deconvolution

The required arguments for `method = "wavelet"` are:

- `h`: A numeric vector corresponding to the wavelet used to deconvolve the GPR data.
- `mu`: A length-one numeric vector defining the amount of noise.

`delineate` *Delineate structure on GPR data*

Description

Delineate structure on GPR data

Usage

```
## S4 method for signature 'GPR'
delineate(x, name = NULL, type = c("raster", "wiggles"),
  addTopo = FALSE, nupspl = NULL, n = 10000, ...)

## S4 method for signature 'GPR'
addDelineation(x, itp, name = NULL, type = c("raster",
  "wiggles"), addTopo = FALSE, ...)

## S4 replacement method for signature 'GPR'
rmDelineations(x) <- value

## S4 method for signature 'GPR'
delineations(x, sel = NULL, ...)

## S4 method for signature 'GPR'
exportDelineations(x, dirpath = "")

## S4 method for signature 'GPR'
plotDelineations3D(x, sel = NULL, col = NULL, add = TRUE,
  ...)
```

```
## S4 method for signature 'GPR'
plotDelineations(x, sel = NULL, col = NULL, ...)

## S4 method for signature 'GPR'
identifyDelineation(x, sel = NULL, ...)
```

---

depth *Depth/time of the GPR data*

Description

Depth/time of the GPR data

Usage

```
## S4 method for signature 'GPR'
depth(x)

## S4 replacement method for signature 'GPR'
depth(x) <- value
```

---

depth0 *Depth zero*

Description

Depth zero

Usage

```
depth0(time_0, v = 0.1, antsep = 1, c0 = 0.299)
```

---

depthToTime *Depth to time conversion*

Description

Depth to time conversion

Usage

```
depthToTime(z, time_0, v = 0.1, antsep = 1, c0 = 0.299)
```

depthunit *Depth unit of the GPR data*

Description

Depth unit of the GPR data

Usage

```
## S4 method for signature 'GPR'
depthunit(x)

## S4 replacement method for signature 'GPR'
depthunit(x) <- value
```

description *Description of the GPR data*

Description

Description of the GPR data

Usage

```
## S4 method for signature 'GPR'
description(x)

## S4 replacement method for signature 'GPR'
description(x) <- value
```

dewow *Trace dewowing*

Description

dewow remove the low-frequency component (the so-called 'wow') of every trace..

Usage

```
## S4 method for signature 'GPR'
dewow(x, type = c("MAD", "Gaussian"), w)
```

Arguments

x	An object of the class GPR.
type	A length-one character vector, either MAD (Median Absolute Deviation filter) or Gaussian (Gaussian filter)
w	A length-one numeric vector equal to the window length of the filter. Per default, the filter length is five times the GPR pulse width.

Value

An object of the class GPR whose traces are dewowed.

Examples

```
data(frenkeLine00)
A <- dewow(frenkeLine00, type = "Gaussian")
A
```

---

displacement *Displacement to align two matrix*

Description

Displacement to align two matrix

Usage

```
displacement(x, y, method = c("phase", "WSSD"), dxy = NULL)
```

---

distTensors *Distance between structure tensors*

Description

Distance between structure tensors

Usage

```
distTensors(J1, J2, method = c("geodesic", "log-Euclidean", "angular", "L2"),
  normalise = FALSE)
```

exportCoord *Export the trace coordinates.*

Description

Export the trace coordinates.

Usage

```
## S4 method for signature 'GPR'
exportCoord(x, type = c("SpatialPoints", "SpatialLines",
  "ASCII"), fPath = NULL, driver = "ESRI Shapefile", ...)
```

exportFid *Export fiducial markers*

Description

Export fiducial markers

Usage

```
## S4 method for signature 'GPR'
exportFid(x, fPath = NULL)
```

exportPDF *Export a PDF showing the GPR profile.*

Description

Export a PDF showing the GPR profile.

Usage

```
## S4 method for signature 'GPR'
exportPDF(x, fPath = NULL, addTopo = FALSE, clip = NULL,
  normalize = NULL, nupspl = NULL, type = "wiggles", ...)
```

exportProc *Export the process steps.*

Description

Export the process steps.

Usage

```
## S4 method for signature 'GPR'
exportProc(x, fPath = NULL, sep = "\t",
  row.names = FALSE, col.names = FALSE, ...)
```

fFilter *Frequency filter*

Description

Frequency filter

Usage

```
## S4 method for signature 'GPR'
fFilter(x, f = 100, type = c("low", "high", "bandpass"),
  L = 257, plotSpec = FALSE)
```

fid *Fiducial markers of the GPR data*

Description

Fiducial markers of the GPR data

Usage

```
## S4 replacement method for signature 'GPR'
fid(x) <- value

## S4 method for signature 'GPR'
fid(x)
```

filepath *Filepath of the GPR data*

Description

Filepath of the GPR data

Usage

```
## S4 method for signature 'GPR'
filepath(x)

## S4 replacement method for signature 'GPR'
filepath(x) <- value
```

filter1D *One dimensional filters*

Description

One dimensional filters

Usage

```
## S4 method for signature 'GPR'
filter1D(x, type = c("median", "hampel", "Gaussian"), ...)
```

filter2D *Two-dimensional filters*

Description

Two-dimensional filters

Usage

```
## S4 method for signature 'GPR'
filter2D(x, type = c("median3x3", "adimpro"), ...)
```

firstBreak *First wave break*

Description

Compute the first wave break.

Usage

```
## S4 method for signature 'GPR'
firstBreak(x, method = c("coppens", "coppens2", "threshold",
  "MER"), thr = 0.12, w = 11, ns = NULL, bet = NULL)
```

Arguments

x	An object of the class GPR
method	A length-one character vector. "coppens" corresponds to the modified Coppens method, "threshold" to the threshold method, and "MER" to the modified energy ratio method.
thr	A length-one numeric vector defining the threshold signal amplitude (in %) at which time zero is picked (only for the threshold method).
w	A length-one numeric vector defining the length of leading window (only for the modified Coppens and modified energy ratio methods). Recommended value: about one period of the first-arrival waveform.
ns	A length-one numeric vector defining the length of the edge preserving smoothing window (only for the modified Coppens method). Recommended value: between one and two signal periods. When ns = NULL the value of ns is set to 1.5 * w.
bet	A length-one numeric vector defining the stabilisation constant (only for the modified Coppens method). Not critical. When bet = NULL the value of bet is set to 20% of the maximal signal amplitude.

References

Modified Coppens method Sabbione J.I. and Velis D. (2010) Automatic first-breaks picking: New strategies and algorithms. Geophysics, 75(4): 67-76.

Modified Energy Ratio (MER) method Han L., Wong J., and John C. (2010) Time picking on noisy microseismograms. In: Proceedings of the GeoCanada 2010 Convention - Working with the Earth, Calgary, AB, Canada, p. 4

See Also

time0 to set time zero and time0Cor to shift the traces such that they start at time zero.

firstBreakToTime0 *Convert first wave break to time-zero*

Description

Account for the delay time between time of wave emission and time of first wave break recording due to the antenna separation (offset).

Time correction for each trace to compensate the offset between transmitter and receiver antennae (it converts the trace time of the data acquired with a bistatic antenna system into trace time data virtually acquiered with a monostatic system under the assumption of horizontally layered structure). If all the traces have the same time-zero, this function does not change the trace but only the time (time scale). If the traces have different time-zero, the traces are first aligned to have the same time-zero (spline interpolation)

Usage

```
firstBreakToTime0(fb, x, c0 = 0.299)

## S4 method for signature 'GPR'
timeCorOffset(x, t0 = NULL)
```

Arguments

x	A object of the class GPR
c0	Propagation speed of the GPR wave through air (used only when keep = NULL).
t0	A numeric vector with length equal either to NULL, or one or to the number traces. If t0 = NULL ‘time0(x)‘ will be used.

See Also

time0 to set time zero and firstBreakToTime0 to convert the first wave break into time zero.

fkFilter *Frequency-wavenumber filter*

Description

Frequency-wavenumber filter

Usage

```
## S4 method for signature 'GPR'
fkFilter(x, fk = NULL, L = c(5, 5), npad = 1)
```

frenkeLine00 *Ground-penetrating radar data.*

Description

Surface ground-penetrating radar data recorded the 25 April 2014 in Frenkental (Swizerland). Coordinates: CH1903+/LV95:2'622'209.66, 1'256'907.54 Elevation: 345.8 m

Usage

```
frenkeLine00
```

Format

An object of the class RGPR

Source

University of Basel (Switzerland)

gain *Gain compensation*

Description

Gain compensation

Usage

```
## S4 method for signature 'GPR'
gain(x, type = c("power", "exp", "agc"), ...)
```

gammaCorrection *Gamma correction of the amplitude*

Description

Gamma correction of the amplitude

Usage

```
## S4 method for signature 'GPR'
gammaCorrection(x, a = 1, b = 1)
```

georef *Georeferencing*

Description

Perform on a set of x,y coordinates (1) a translation by -cloc, then (2) a rotation by alpha (radian), and (3) a translation by creg. If creg is NULL, then creg is set equal to cloc.

Arguments

x	A matrix with the first two columns corresponding to coordinates.
alpha	A length-one numeric vector corresponding to the rotation angle in radians. If alpha = NULL, alpha is estimated from the pairs of points in the local reference system (ploc) and in the regional reference system (preg).
cloc	A length-two numeric vector corresponding to the coordinate center of the local reference system
creg	A length-two numeric vector corresponding to the coordinate center of the regional reference system. Setting creg = NULL (default) is equivalent to apply a rotation of angle alpha and center cloc.
ploc	A matrix with the first two columns corresponding to coordinates in the local reference system.
preg	A matrix with the first two columns corresponding to coordinates in the regional reference system.
FUN	If alpha = NULL, a function to estimate the rotation angle from the angles computed for each pairs of coordinates of ploc-preg.

getGPR *Extract GPR object from GPRsurvey object*

Description

Extract GPR object from GPRsurvey object

Usage

```
## S4 method for signature 'GPRsurvey'
getGPR(x, id)
```

gethd *Return data header*

Description

Return data header

Usage

```
## S4 method for signature 'GPR'
gethd(x, hd = NULL)
```

gkernel *Gaussian 2d-kernel*

Description

Gaussian 2d-kernel

Gaussian x-derivative kernel (edge detector)

Gaussian y-derivative kernel (edge detector)

Usage

```
gkernel(n, m, sd = 1)

dx_gkernel(n, m, sd = 1)

dy_gkernel(n, m, sd = 1)
```

GPR-class *Class GPR*

Description

An S4 class to represent a ground-penetrating radar (GPR) data.

Slots

`version` A length-one character vector indicating the version of RGPR

`data` A $m \times n$ numeric matrix consiting of a cross-section of signal amplitudes as a function of the GPR position. The columns of `data` correspond to the GPR traces and the row of `data` to the time/depth samples.

`traces` A length-m numeric vector corresponding to the trace number.

`depth` A length-n numeric vector indicating the sampling time or the vertical position of the trace samples.

`pos` A length-m numeric vector indicating the relative position of the trace along the survey profile.

`time0` A length-m numeric vector containing the 'time-zero' of every trace.

`time` A length-m numeric vector containing the recording time of every trace.

`fid` A length-m character vector containing fiducial markers associated with the traces.

`ann` A length-m character vector containing annotations associated with the traces.

`coord` A $m \times 3$ matrix containing the (x, y, z) positions of every trace.

`rec` A $m \times 3$ matrix containing the (x, y, z) positions of the receiver for every trace.

`trans` A $m \times 3$ matrix containing the (x, y, z) positions of the transmitter for every trace.

`coordref` A length-3 numeric vector containing the coordinates of a local reference.

`freq` A length-one numeric vector corresponding to the GPR antennae frequency (in MHz).

`dz` A length-one numeric vector corresponding to the time or depth sampling step.

`dx` A length-one numeric vector corresponding to the trace step.

`antsep` A length-one numeric vector corresponding to the antenna separation.

`name` A length-one character vector containing the name of the GPR data.

`description` A length-one character vector containing the description of the GPR data.

`filepath` A length-one character vector containing the file path of the original GPR data.

`depthunit` A length-one character vector corresponding to the time/depth unit (e.g., "ns", "m").

`posunit` A length-one character vector corresponding to the (x, y)-unit (e.g., "m").

`surveymode` A length-one character vector containing the survey mode (e.g., "Reflection", "CMP")

`date` A length-one character vector containing the date of the survey in the format "yyyy-mm-dd".

`crs` A length-one character vector containing the coordinate reference system following the R notation of proj4string from the PROJ.4 library.

`proc` A length-varying character vector whose each element correspond to a processing step applied to the data.

`vel` A list containing the velocity model.

`delineations` A list containing delineated structures.

`hd` A list containing less relevant additional informations.

GPR-subset *extract parts of GPR*

Description

Object of the class GPR can be manipulated as matrix

Usage

```
## S4 method for signature 'GPR,ANY,ANY,ANY'
x[i, j, drop]

## S4 replacement method for signature 'GPR,ANY,ANY,ANY'
x[i, j] <- value
```

Arguments

x	Object of class GPR
i	integer
j	integer

GPRsurvey *Create an object of the class GPRsurvey*

Description

Create an object of the class GPRsurvey using a vector of GPR data filepath

Usage

```
GPRsurvey(LINES)
```

GPRsurvey-subset *extract parts of GPRsurvey*

Description

Return an object of class GPRsurvey

Usage

```
## S4 method for signature 'GPRsurvey,ANY,ANY,ANY'
x[i, j, drop]
```

GPRsurvey.as.SpatialLines *Coerce to SpatialLines*

Description

Coerce to SpatialLines

Coerce to SpatialPoints

Usage

```
## S4 method for signature 'GPRsurvey'
as.SpatialLines(x)

## S4 method for signature 'GPRsurvey'
as.SpatialPoints(x)
```

inPoly *Return points that are within a polygon*

Description

Return points that are within a polygon

Usage

```
inPoly(x, y, vertx, verty)
```

interpPos *Interpolate trace positions from measurement (e.g., GPS).*

Description

Interpolate trace positions from measurement (e.g., GPS).

Usage

```
## S4 method for signature 'GPR'
interpPos(x, topo, plot = FALSE, r = NULL, tol = NULL,
  method = c("linear", "spline", "pchip"), ...)
```

Arguments

x	An object of the class GPR.
topo	A $m \times 4$ numeric matrix, with m the number of traces in 'x'. The columns names of topo must be "E", "N", "Z" and "TRACE".
plot	A length-one boolean vector. If TRUE some control plots are displayed.
r	A 'RasterLayer' object from the package 'raster' from which trace elevation z will be extracted based on the trace position (x, y) on the raster.
tol	Length-one numeric vector: if the horizontal distance between two consecutive trace positions is smaller than tol, then the traces in between as well as the second trace position are removed. If tol = NULL, tol is set equal to sqrt(.Machine$double.eps).
method	A length-three character vector defining the interpolation methods (same methods as in signal::interp1: "linear", "nearest", "pchip", "cubic", and "spline"). First element for the interpolation of the inter-trace distances, second element for the interpolation of the horizontal trace positions, and third element for the interpolation of the vertical trace positions.

---

intersections *Return intersection from GPRsurvey*

Description

Return intersection from GPRsurvey

Usage

```
## S4 method for signature 'GPRsurvey'
intersections(x)
```

---

latlongToUTM *latitude-longitude to UTM*

Description

see https://stackoverflow.com/a/30225804

Usage

```
latlongToUTM(lat, long, zone = NULL, south = FALSE)
```

lines *Add a GPR trace on a plot*

Description

Add a GPR trace on a plot

Plot the GPR survey as lines

Usage

```
## S3 method for class 'GPR'
lines(x, ...)

## S3 method for class 'GPRsurvey'
lines(x, ...)
```

linkCoordFid *Link coordinates to fiducial marker*

Description

To interpolate topo

Usage

```
linkCoordFid(y, xyz, pcode, tol = 0.1)
```

Arguments

y	object of class GPSsurvey
xyz	matrix of coordinates
pcode	character vector (length(pcode) = nrow(xyz)) indicating which coordinates to which GPR data belongs
tol	Tolerance to detect duplicates from topo data

Math,GPR-method *Basic mathematical functions*

Description

Methods for the base Math methods S4groupGeneric

Usage

```
## S4 method for signature 'GPR'
Math(x)
```

Arguments

x	An object of the class RGPR.

Details

Currently implemented methods include:

- "abs", "sign", "sqrt", "ceiling", "floor", "trunc", "exp", "expm1", "log", "log10", "log2", "log1p", "cos", "cosh", "sin", "sinh", "tan", "tanh"

Examples

```
data(frenkeLine00)
A <- exp(frenkeLine00)
```

migration *Migration of the GPR data*

Description

Migration of the GPR data

Usage

```
## S4 method for signature 'GPR'
migration(x, type = c("static", "kirchhoff"), ...)
```

name *Name of the GPR data*

Description

Name of the GPR data

Usage

```
## S4 method for signature 'GPR'
name(x)

## S4 replacement method for signature 'GPR'
name(x) <- value
```

NMOCor *Normal Move-Out correction*

Description

Remove the Normal Move-Out (NMO) from the trace given a constant velocity: this is a non-linear correction of the time axis that require interpolation. Note that only the conventional NMO correction is currently implemented. The conventional NMO introduces a streching effect. A nonstretch NMO will be implemented in a near future.

Usage

```
## S4 method for signature 'GPR'
NMOCor(x, v = NULL, asep = NULL)
```

Arguments

x	An object of the class GPR
v	A length-one numeric vector defining the radar wave velocity in the ground
asep	A length-n numeric vector defining the antenna separation for each trace (n = number of traces; for example: seq(x@antsep, by = x@dx, length.out = length(x))). If NULL, the slots rec and trans of x are used to compute the distance between the antennas

Details

Assuming a horizontal reflecting plane and homogeneous medium, the two-way bistatic travel time of the reflected wave for an antenna separation x follows directly from the Pythagorean theorem:

$$t_{TWT}(x,z) = \sqrt{\frac{x^2}{v^2} + \frac{4z^2}{v^2}}$$

where $t_{TWT}(x)$ is the two-way travel time at antenna separation x of the wave reflected at depth z with propagation velocity v. This equation defines an hyperbola (keep z constant, increase the antenna separation x and you obtain a hyperbola similar to the reflection signals you obtain with common-mid point survey). The idea behind NMO-correction is to correct the signal for the antenna separation (offset) and therefore to transform the signal to the signal we would have recorded with zero offset ($x = 0$). We write the vertical two-way traveltime at zero offset

$$t_0 = t_{TWT}(x = 0) = \frac{2z}{v}$$

Therefore, the NMO-correction Δ_{NMO} is

$$\Delta_{NMO} = t_{TWT}(x) - t_0$$

$$\Delta_{NMO} = t_0(\sqrt{1 + \frac{x^2}{v^2 t_0^2}} - 1)$$

References

- Tillard and Dubois (1995) Analysis of GPR data: wave propagation velocity determination. Journal of Applied Geophysics, 33:77-91
- Shatilo and Aminzadeh (2000) Constant normal-moveout (CNMO) correction: a technique and test results. Geophysical Prospecting, 473-488

`optPhaseRotation` *Optimum Phase Rotation*

Description

Optimum Phase Rotation

Usage

```
optPhaseRotation(x, rot = 0.01, plot = TRUE)
```

Arguments

`x`	any data that can be converted into a numeric vector with as.vector.
`rot`	The phase rotation increment.
`plot`	A lenth-one boolean vector. If TRUE, the kurtosis as a function of phase angle is plotet.

padmat *pad a matrix*

Description

pad a matrix

Usage

```
padmat(x, n, m, what = 0)
```

palGPR *Plot single colour palette*

Description

source: vignette of the R-package "colorspace" (Color Space Manipulation)

Colour palette

Usage

```
palGPR(colPal = "default", n = 101, power = 1, returnNames = FALSE)

plotPal(col, border = NA)

displayPalGPR()
```

Examples

```
plotPal(palGPR("hcl_5"))
displayPalGPR()
```

papply *Apply processing to GPRsurvey object*

Description

Apply processing to GPRsurvey object

Usage

```
## S4 method for signature 'GPRsurvey'
papply(x, prc = NULL)
```

phaseRotation *Phase rotation shift the phase of signal by phi (in radian)*

Description

Phase rotation

shift the phase of signal by phi (in radian)

Usage

```
phaseRotation(x, phi)
```

plot *Plot the GPR object.*

Description

If the GPR object consists of a single trace, wiggle plot is shown.

Plot GPR suvey lines

Usage

```
## S3 method for class 'GPR'
plot(x, y, ...)

## S3 method for class 'GPRsurvey'
plot(x, y, ...)
```

plot3DRGL *Three-dimensional plot of the GPR data with Open-GL*

Description

Three-dimensional plot of the GPR data with Open-GL

Usage

```
## S4 method for signature 'GPR'
plot3DRGL(x, addTopo = FALSE, clip = NULL,
  normalize = NULL, nupspl = NULL, add = TRUE, xlim = NULL,
  ylim = NULL, zlim = NULL, ...)
```

plotAmpl *Plot the trace amplitude*

Description

Plot the amplitude estimated over the whole GPR data as a function of time/depth.

Usage

```
## S4 method for signature 'GPR'
plotAmpl(x, FUN = mean, add = FALSE, all = FALSE,
  plotLog = TRUE, ...)
```

Arguments

x	An object of the class GPR.
FUN	A function to be applied on each row of the GPR data to estimate the wave amplitude as a function of time/depth.
add	A length-one boolean vector. If TRUE the amplitude is plotted on the previous plot. If FALSE (default) a new plot is created.
all	A length-one boolean vector. If TRUE the logarithm of the amplitude of every trace is ploted on the estimate amplitude. Default is FALSE. processing functions with their arguments applied previously on the GPR data.

Examples

```
data(frenkeLine00)
plotAmpl(frenkeLine00, FUN = median)
```

plotRaster *Plot GPR as image (raster)*

Description

Plot GPR as image (raster)

Usage

```
plotRaster(z, x = NULL, y = NULL, main = "", xlim = NULL, note = NULL,
  ratio = 1, time_0 = 0, antsep = 1, v = 0.1, surveymode = NULL,
  addFid = TRUE, fid = NULL, ylim = NULL, addAnn = TRUE,
  annotations = NULL, depthunit = "ns", posunit = "m",
  rasterImage = TRUE, resfac = 1, clab = "mV", add = FALSE,
  barscale = TRUE, addGrid = FALSE, col = palGPR(n = 101), yaxt = "s",
  bty = "o", relTime0 = TRUE, clim = NULL, pdfName = NULL, ...)
```

plotTensor *Plot structure tensor on GPR data*

Description

Plot structure tensor on GPR data

Usage

```
plotTensor(x, 0, type = c("vectors", "ellipses"), normalise = FALSE,
  spacing = c(6, 4), len = 1.9, n = 10, ratio = 1, ...)
```

plotTensor0 *Plot structure tensor on GPR data*

Description

Plot structure tensor on GPR data

Usage

```
plotTensor0(alpha, l1, l2, x, y, col = NULL, type = c("vectors",
  "ellipses"), normalise = FALSE, spacing = c(6, 4), len = 1.9, n = 10,
  ratio = 1, ...)
```

points *Add a GPR trace points on a plot*

Description

Add a GPR trace points on a plot

Usage

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

pos *Position of the GPR traces*

Description

Position of the GPR traces

Usage

```
## S4 method for signature 'GPR'
pos(x)

## S4 replacement method for signature 'GPR'
pos(x) <- value
```

posLine *Position on a multiline*

Description

Position on a multiline

Usage

```
posLine(loc, last = FALSE)
```

posunit *Position unit of the GPR data*

Description

Position unit of the GPR data

Usage

```
## S4 method for signature 'GPR'
posunit(x)

## S4 replacement method for signature 'GPR'
posunit(x) <- value
```

print *Print GPR*

Description

Print GPR

Identical to print().

Print GPR survey

Identical to print().

Usage

```
## S3 method for class 'GPR'
print(x, ...)

## S4 method for signature 'GPR'
show(object)

## S3 method for class 'GPRsurvey'
print(x, ...)

## S4 method for signature 'GPRsurvey'
show(object)
```

proc *Processing steps applied to the data*

Description

`processing` returns all the processing steps applied to the data.

Add a processing step

Usage

```
## S4 method for signature 'GPR'
proc(x)

## S4 replacement method for signature 'GPR'
proc(x) <- value
```

Arguments

x	An object of the class GPR.

Value

A character vector whose elements contain the name of the processing functions with their arguments applied previously on the GPR data.

Examples

```
data(frenkeLine00)
A <- dewow(frenkeLine00, type = "Gaussian")
proc(A)
```

---

processing *DEPRECATED - Processing steps applied to the data*

Description

DEPRECATED - use proc instead! processing returns all the processing steps applied to the data.

Usage

```
## S4 method for signature 'GPR'
processing(x)
```

Arguments

x	An object of the class GPR.

Value

A character vector whose elements contain the name of the processing functions with their arguments applied previously on the GPR data.

Examples

```
data(frenkeLine00)
A <- dewow(frenkeLine00, type = "Gaussian")
processing(A)
```

---

readFID *read fiducial marker files*

Description

read fiducial marker files

Usage

```
readFID(FID, sep = ",")
```

readGPR *Read a GPR data file*

Description

Read a GPR data file

Usage

```
## S4 method for signature 'character'
readGPR(fPath, desc = "")
```

Arguments

fPath	Filepath (character).
desc	Short description of the file (character).
coordfile	Filepath of a text file containing the coordinates (x,y,z) of each traces.
crs	Coordinate reference system (character)
intfile	Filepath of a text file containing the intersection.

Value

The GPR data as object of the class RGPR.

Examples

```
NULL
```

readTopo *read topo file*

Description

read topo file

Usage

```
readTopo(TOPO, sep = ",")
```

regInterpPos *Trace interpolation at regularly spaced positions*

Description

Trace interpolation at regularly spaced positions

Usage

```
## S4 method for signature 'GPR'
regInterpPos(x, type = c("linear", "cosine"), dx = NULL)
```

relPos *Relative trace position on the GPR profile.*

Description

Relative trace position on the GPR profile.

Usage

```
## S4 method for signature 'GPR'
relPos(x)
```

repmat *Repeat matrix*

Description

Repeat a matrix row-wise n times and column-wise m times.

Usage

```
repmat(A, n, m)
```

Details

Source A replication of MatLab repmat function! R FOR OCTAVE USERS version 0.4, Copyright (C) 2001 Robin Hankin http://cran.r-project.org/doc/contrib/R-and-octave.txt

reverse *Reverse the trace position.*

Description

Reverse the trace position.

Reverse the trace position.

Usage

```
## S4 method for signature 'GPR'
reverse(x, id = NULL, tol = 0.3)

## S4 method for signature 'GPRsurvey'
reverse(x, id = NULL, tol = 0.3)
```

rotatePhase *Phase rotation*

Description

Rotate the phase of the GPR data by a given angle `phi`.

Usage

```
## S4 method for signature 'GPR'
rotatePhase(x, phi)
```

Arguments

`x`	A GPR data
`phi`	A length-one numeric vector defining the phase rotation in radian.

Value

The GPR data with rotated phase.

shiftEst *Shift estimation between two GPR profiles.*

Description

Shift estimation between two GPR profiles.

Usage

```
## S4 method for signature 'GPR'
shiftEst(x, y = NULL, method = c("phase", "WSSD"),
  dxy = NULL, ...)
```

shiftmat *shift a matrix by n and m shift a matrix by n and m*

Description

shift a matrix by n and m

shift a matrix by n and m

Usage

```
shiftmat(x, n, m)
```

spec *Return the amplitude spectrum of the GPR object.*

Description

Return the amplitude spectrum of the GPR object.

Usage

```
## S4 method for signature 'GPR'
spec(x, type = c("f-x", "f-k"), plotSpec = TRUE,
  unwrapPhase = TRUE, ...)
```

strTensor *Structure tensor field of GPR data*

Description

Structure tensor field of GPR data

surveyIntersect *Compute the survey intersections*

Description

Compute the survey intersections

Usage

```
## S4 method for signature 'GPRsurvey'
surveyIntersect(x)
```

svDate *Survey date*

Description

Return NULL if no date exists, else an object of the class 'Date'

Usage

```
## S4 method for signature 'GPR'
svDate(x)

## S4 replacement method for signature 'GPR'
svDate(x) <- value
```

Arguments

x	An object of the class 'GPR'
value	An object of the class 'Date'

time0 *'time-zero' of every traces*

Description

time0 returns the 'time-zero' of every traces. Generally, 'time-zero' corresponds to the first wave arrival (also called first wave break).

Usage

```
## S4 method for signature 'GPR'
time0(x)

## S4 replacement method for signature 'GPR'
time0(x) <- value
```

Arguments

x	An object of the class GPR.

Value

A vector containing the time-zero values of each traces.

See Also

firstBreak to estimate the first wave break.

Examples

```
data(frenkeLine00)
time0(frenkeLine00)
```

time0Cor *Time zero correction*

Description

time0Cor shift the traces vertically such that they start at time zero (time zero of the data can be modified with the function)

Usage

```
## S4 method for signature 'GPR'
time0Cor(x, t0 = NULL, method = c("spline", "linear",
  "nearest", "pchip", "cubic", "none"), crop = TRUE, keep = 0)
```

Arguments

x	A object of the class GPR
t0	A numeric vector with length equal either to NULL, or one or to the number traces. The traces will be shifted to t0. If t0 = NULL 'time0(x)' will be used instead. If t0 is the time-zero, set keep = 0.
method	A length-one character vector defining the interpolation method that are from the function 'interp1' from the 'signal' package.
crop	If TRUE (defaults), remove the rows containing only zero's (no data).
keep	A length-one numeric vector indicating in time units how much of the trace has to be kept before time zero.
c0	Propagation speed of the GPR wave through air (used only when keep = NULL).

Details

When keep = NULL the amount of time kept is equal to time taken by the air wave to travel from the transmitter to the receiver.

Value

An object of the class GPR.

See Also

time0 to set time zero and firstBreak to estimate the first wave break. firstBreakToTime0 to convert the first wave break into time zero.

Examples

```
data(frenkeLine00)
tfb <- firstBreak(frenkeLine00)
t0 <- firstBreakToTime0(tfb, frenkeLine00, c0 = 0.299)
time0(frenkeLine00) <- t0
frenkeLine00_2 <- time0Cor(frenkeLine00, method = "pchip")
```

---

timeToDepth *time to depth conversion*

Description

time to depth conversion

Usage

```
timeToDepth(tt, time_0, v = 0.1, antsep = 1, c0 = 0.299)
```

traceAverage *Trace average*

Description

Compute the average trace of a radargram (resulting in a single trace) or a moving average of the traces.

Usage

```
## S4 method for signature 'GPR'
traceAverage(x, w = NULL, FUN = mean, ...)
```

Arguments

x	An object of the class GPR
w	A length-one integer vector equal to the window length of the average window. If w = NULL a single trace corresponding to the average trace of the whole profile is returned.
FUN	A function to compute the average (default is mean)
...	Additional parameters for the FUN functions

Value

An object of the class GPR. When w = NULL, this function returns a GPR object with a single trace corresponding to the average trace of the whole radargram. When w is equal to a strictly positive interger this function returns a GPR object with a size identical to x where each trace corresponds to the average of the w neighbouring traces centered on the considered trace.

Examples

```
data("frenkeLine00")

f0 <- frenkeLine00

f1 <- traceAverage(f0)
plot(f1)
# substract the average trace
plot(f0 - f1)

f2 <- traceAverage(f0, w = 20)
plot(f2)
plot(f0 - f2)

f3 <- traceAverage(f0, w = 20, FUN = median)
plot(f3)
plot(f0 - f3)
```

traceScaling *Trace scaling*

Description

Trace scaling

Usage

```
## S4 method for signature 'GPR'
traceScaling(x, type = c("stat", "min-max", "95", "eq", "sum",
  "rms", "mad", "invNormal"))
```

traceShift *Shift vertically the traces by an amount of depth units.*

Description

Shift vertically the traces by an amount of depth units.

Usage

```
## S4 method for signature 'GPR'
traceShift(x, ts, method = c("spline", "linear", "nearest",
  "pchip", "cubic", "none"), crop = TRUE)
```

Arguments

x	A object of the class GPR
ts	A numeric vector defining the amount of depth the traces have to shifted
method	A length-one character vector indicating the interpolation method. "none" means that the trace is shifted by the amount of points that is the closest to amount of depth ts.
crop	If TRUE (defaults), remove the rows containing only zero's (no data).,

Value

An object of the class GPR.

See Also

time0Cor to shift the traces such that they start at time zero.

trAmplCor *Gain compensation*

Description

Gain compensation

Usage

```
## S4 method for signature 'GPRsurvey'
trAmplCor(x, type = c("power", "exp", "agc"), ...)
```

trimStr *Trim string*

Description

returns string w/o leading or trailing whitespace

Usage

```
trimStr(x)
```

trRmDuplicates *Remove traces with duplicated trace positions*

Description

checks for duplicates trace positions (up to precision defined by 'tol') and remove them from 'x' (object of the class GPR or GPRsurvey). If there is a series of consecutive traces with an inter-distance smaller than the tolerance threshold defined by the computer precision, the function starts by removing traces every two traces and repeat the procedure until the trace inter-distances are larger that the threshold. Example with 5 traces:

- distance between trace 1 and 2 < tol
- distance between trace 2 and 3 < tol
- distance between trace 3 and 4 < tol
- distance between trace 4 and 5 < tol

The algorithm first remove trace 2 and 4 and recompute the inter-trace distances:

- distance between trace 1 and 3 < tol
- distance between trace 3 and 5 > tol

The algorithm remove trace 3. END!

Usage

```
## S4 method for signature 'GPR'
trRmDuplicates(x, tol = NULL)
```

Arguments

x	An object of the class GPR
tol	Length-one numeric vector: if the horizontal distance between two consecutive traces is smaller than tol, then the second trace is removed. If tol = NULL, tol is set equal to sqrt(.Machine$double.eps).

---

trTime *Time of data collection for each trace*

Description

Time of data collection for each trace

---

upsample *Up-sample the GPR data (1D and 2D sinc-interpolation)*

Description

Up-sample the GPR data (1D and 2D sinc-interpolation)

Usage

```
## S4 method for signature 'GPR'
upsample(x, n)
```

---

values *Values of the GPR data*

Description

Values of the GPR data

Usage

```
## S4 method for signature 'GPR'
values(x)

## S4 replacement method for signature 'GPR'
values(x) <- value
```

vel *Velocity model of the GPR data*

Description

Velocity model of the GPR data

Usage

```
## S4 method for signature 'GPR'
vel(x)

## S4 replacement method for signature 'GPR'
vel(x) <- value
```

wapplyRow *Wapply on the row of a matrix (windowed)*

Description

NOT CURRENTLY USED mod by MANU

Usage

```
wapplyRow(x = NULL, width = NULL, by = NULL, FUN = NULL, ...)
```

writeGPR *Write the GPR object in a file.*

Description

Write the GPR object in a file.

Usage

```
writeGPR(x, fPath = NULL, type = c("DT1", "rds", "ASCII", "xyzv"),
  overwrite = FALSE, ...)

## S4 method for signature 'GPR'
writeGPR(x, fPath = NULL, type = c("DT1", "rds", "ASCII",
  "xyzv"), overwrite = FALSE, ...)
```

writeSurvey *Write GPRsurvey object*

Description

Write GPRsurvey object

Usage

```
## S4 method for signature 'GPRsurvey'
writeSurvey(x, fPath, overwrite = FALSE)
```

[[*extract a GPR object from a GPRsurvey object*

Description

Return an object of class GPR

Usage

```
## S4 method for signature 'GPRsurvey,ANY,ANY'
x[[i, j, ...]]

## S4 replacement method for signature 'GPRsurvey,ANY,ANY'
x[[i]] <- value
```

Index