# Package 'RGPR'

July 28, 2016

**Type** Package

**Title** GPR processing and visualisation

**Version** 0.0.0

**Date** 2015-05-06

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

**Depends** methods,
graphics

**License** MIT License

**URL** http://stanford.edu/~ehuber/RGPR.html

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

**LazyData** true

**Collate** 'global.R'
'ClassGPR.R'
'ClassGPRsurvey.R'
'data.R'

**Imports** methods,
graphics,
sp,
rgdal,
rgeos,
MASS,
signal,
colorspace,
Cairo,
rgl,
EMD,
mmand,
e1071,
plot3D

**RoxygenNote** 5.0.1

# R topics documented:

---

`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, values) <- value
```

## Arith *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)
```

---

## conv1D *Trace convolution (1D)*

---

### Description

Convolution of the GPR traces with a wavelet

Convolution of the GPR data with a kernel

### Usage

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

## S4 method for signature 'GPR'
conv2D(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. |
| x | A GPR data |
| w | A numeric matrix with smaller dimension than the GPR data. |

### Value

The convolved GPR data.

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, B)
```

### 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, values) <- value
```

## coordref *Define a local reference coordinate*

### Description

Define a local reference coordinate

### Usage

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

## 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 removal*

---

## Description

Direct-current removal

Compute the first wave break

## Usage

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

## S4 method for signature 'GPR'
firstBreack(x, w = 11, ns = NULL, bet = NULL)
```

## Details

Jaun I. Sabbione and Danilo Velis (2010). Automatic first-breaks picking: New strategies and algorithms. Geophysics, 75 (4): v67-v76 -> modified Coppens's Method nl = length leading window: about one period of the firs-arrival waveform ns = length eps (edge preserving smoothing) window: good results with ns between one and two signal periods -> default values ns= 1.5*nl bet = stabilisation constant, not critical, set to 0.2*max(amplitude)

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, values = NULL) <- 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, ...)
```

## 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
```

---

## distTensors *Distance between structure tensors*

---

### Description

Distance between structure tensors

### Usage

```
distTensors(a1, b1, c1, a2, b2, c2, method = c("geodesic", "log-Euclidean"))
```

## 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, folder = NULL, sep = "\t",
  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, ...)
```

## 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, values) <- 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"), ...)
```

## 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)
```

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

## 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.

---

`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)
```

## interpPos *Interpolate the trace position.*

### Description

Interpolate the trace position.

### Usage

```
## S4 method for signature 'GPR'
interpPos(x, topo, plot = FALSE, ...)
```

### Arguments

| | |
|---|---|
| x | An object of the class GPR. |
| topo | A (mx4) 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. |

## lines *Add a GPR trace on a plot*

### Description

Add a GPR trace on a plot

### Usage

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

## Math *Basic mathematical functions*

### Description

Basic mathematical functions

### Usage

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

### Arguments

| | |
|---|---|
| x | An object of the class RGPR. |

### 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
```

## 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. |

## palGPR *Colour palette*

### Description

Colour palette

### Usage

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

plotPal(col, border = NA)

displayPalGPR()
```

### Examples

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

## plot *Plot the GPR object.*

### Description

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

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

### 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, ...)
```

### 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)
```

---

## 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, ...)
```

---

## 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)
```

---

## processing *Processing steps applied to the data*

---

### Description

`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)
```

## readGPR *Read a GPR data file*

### Description

Read a GPR data file

### Usage

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

### 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
```

## 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.

## Usage

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

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

## 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:

### References

Several books!

---

## 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.

---

## 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, ...)
```

---

## strucTensor *Structure tensor of GPR data*

---

### Description

Structure tensor of GPR data

### Usage

```
strucTensor(P, winBlur = c(3, 3), winEdge = c(7, 7), winTensor = c(5, 10),
  sdTensor = 2, dxy = c(1, 1), blksze = c(2, 2), thresh = 0.1, ...)
```

## 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

firstBreack to estimate the first wave break.

### Examples

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

## timeCorOffset *Constant-offset correction (time) of the GPR data*

### Description

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).

### Usage

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

## traceScaling *Trace scaling*

### Description

Trace scaling

### Usage

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

## traceShift *Shift the traces*

### Description

Shift the traces

### Usage

```
## S4 method for signature 'GPR'
traceShift(x, t0, keep = 10, delete0 = TRUE)
```

## upsample *Up-sample the GPR data (sinc-interpolation)*

### Description

Up-sample the GPR data (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, values) <- value
```

## writeGPR *Write the GPR object in a file.*

### Description

Write the GPR object in a file.

### Usage

```
## S4 method for signature 'GPR'
writeGPR(x, fPath, format = c("DT1", "rds"),
  overwrite = FALSE)
```

# Index