



RGPR - an open-source package to process and visualize GPR data

Swiss National Science Foundation
(grants no. CRSI22_132249/1
and P2BSP2_161955)



funding/sponsor

6th top programming
language (IEEE ranking 2017)

key features

- freely available and open-source
- multi-platform
- reproducible data processing
- easily expandable
- high-quality plot
- R package

source code
+ documentation
+ tutorials



emanuelhuber.github.io/RGPR

easy to install

install RGPR from github

```
# install "devtools" if not already done
if(!require("devtools")){
  install.packages("devtools")
}
devtools::install_github("emanuelhuber/RGPR")
```

develop version,
work in progress

read several GPR data files

```
mySurvey <- GPRsurvey(c('XLINE0.dt1',
  'XLINE1.dt1',
  ...
  'YLINE2.dt1',
  'XLINE3.dt1'))
```

add topographic data

```
mySurvey <- interpPos(mySurvey, FID_markers)
```

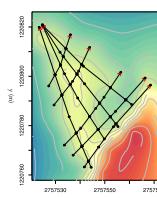
georeferencing

```
mySurvey <- georef(mySurvey, alpha = pi/2,
  creg = c(622259, 256895))
```

class GPRsurvey

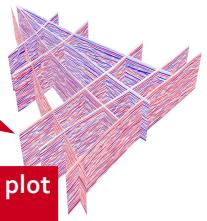
plot survey lines

```
plot(raster_topo)
plot(mySurvey, add = TRUE)
```



3D interactive plot

```
plot3DRGL(mySurvey)
```



3D interactive plot
with open GL

to process GPR data
(objects can be
manipulated as matrix)

extract GPR data 'XLINE1.dt1'
for processing

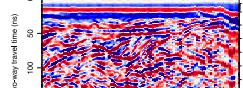
class GPR

read single data file

```
x <- readGPR('XLINE1.dt1')
```

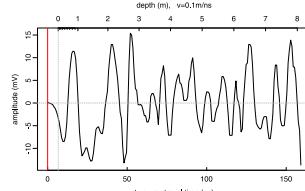
plot radargram

```
plot(x)
```



plot trace #14

```
plot(x[,14])
```



processing

```
x <- dcshift(x, u = 1:110)
x <- dewow(x, type = 'MAD',
  w = 50)
x <- fFilter(x, f = c(150, 260),
  type = 'low')
```

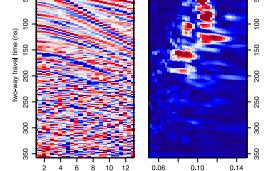
processing track

```
x
*** Class GPR ***
name      = LINE01
filepath  = rawGPR/LINE01.DT1
description =
survey date = 2014-04-25
Reflection, 100 MHz, Window
length = 399.6 ns, dz = 0.4 ns
46 traces, 11.25 m
> PROCESSING
1. coord<- ...
2. dcshift//u=1:110
3. dewow//type=MAD+w=50
4. fFilter//f=150,260+type=low
*****
```

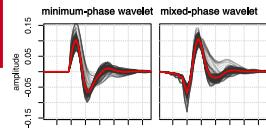
CMP analysis

```
CMPA <- CMPanalysis(
  CMP, w = 10,
  v = seq(0.05, 0.15,
  length = 40))
plot(CMPA)
```

3D interpolation



deconvolution



keep track of
processing

1D processing	2D processing
dewow	f-k filter
DC-shift	2D convolution
trace average	3x3-median filter
amplitude correction	2D adaptative smoothing
frequency filter	Kirchhoff migration
constant-offset correction	
deconvolution	

available processing
functions

what's next?

short term perspectives

- transform functions
- CMP/velocity modeling
- delineations

long term perspectives

- support for cross-borehole GPR
- forward GPR simulator

collaborative development
on github

Yes, you can... contribute

- share your user experience
- report bugs
- ask questions
- propose modifications on github

