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

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

source code + documentation + tutorials

to manipulate trace coordinates (contains meta-data, coordinates and filepaths only)

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

Swiss National Science Foundation (grants no. CRSI22_132249/1 and P2BSP2_161555)

de Berger & Bears AG

funding/sponsor

6th top programming language (IEEE ranking 2017)

easy to install

install RGPR from github

install `devtools` if not already done

```r
if(!require("devtools")){
  install.packages("devtools")
}
```

devtools::install_github("emanuelhuber/RGPR")

install RGPR from github
easy to install

6th top programming language (IEEE ranking 2017)

read several GPR data files

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

add topographic data

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

plot survey lines

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

3D interactive plot

```r
plot3DRGL(mySurvey)
```

3D interactive plot with open GL

class GPRsurvey

extract GPR data 'XLINE1.dt1' for processing

```r
x <- mySurvey[[2]]
```

class GPR

process single data file

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

processing

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

processing track

```r
*** Class GPR ***
name = LINE01
filepath = rawGPR/LINE01.DT1
description =
survey date = 2014-04-25
Reflection, 100 MHz, Window
46 traces, 11.25 m

> PROCESSING
  1. coord=
  2. dcshift(u=1:110)
  3. dewow(type=MAD,w=50)
  4. fFilter(f=c(150,260),type=low)
```

keep track of processing

class GPRstack

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

collaborative development on github

Yes, you can... contribute
- share your user experience
- report bugs
- ask questions
- propose modifications on github

develop version, work in progress

class GPRcube

```r
class GPRslice
```

3D processing (e.g., migration, attributes)

```r
class GPRSlice
```

structure delineation etc.

horizontal slice

vertical slice

3D interpolation

transform

back-transform

SVD decomposition

Fourier transform

Hilbert transform etc.

available processing functions

1D processing

<table>
<thead>
<tr>
<th>function</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dewow</td>
<td>DC-shift</td>
</tr>
<tr>
<td>DC-shift</td>
<td>trace average</td>
</tr>
<tr>
<td>trace average</td>
<td>amplitude correction</td>
</tr>
<tr>
<td>amplitude correction</td>
<td>frequency filter</td>
</tr>
<tr>
<td>frequency filter</td>
<td>constant-offset correction</td>
</tr>
<tr>
<td>deconvolution</td>
<td>f-k filter</td>
</tr>
</tbody>
</table>

2D processing

<table>
<thead>
<tr>
<th>function</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2D convolution</td>
<td>2D convolution</td>
</tr>
<tr>
<td>2D convolution</td>
<td>2D adaptive smoothing</td>
</tr>
<tr>
<td>2D adaptive smoothing</td>
<td>Kirchhoff migration</td>
</tr>
</tbody>
</table>

what's next?

short term perspectives
- transform functions
- CMP/velocity modeling
- delineations

long term perspectives
- support for cross-borehole GPR
- forward GPR simulator

WATCHES & JEWELLERY

development

source code + documentation + tutorials

emanuelhuber.github.io/RGPR

easy to install

install `devtools` if not already done

```r
if(!require("devtools")){
  install.packages("devtools")
}
```

devtools::install_github("emanuelhuber/RGPR")

install RGPR from github
easy to install

6th top programming language (IEEE ranking 2017)

read several GPR data files

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

add topographic data

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

plot survey lines

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

3D interactive plot

```r
plot3DRGL(mySurvey)
```

3D interactive plot with open GL

class GPRsurvey

extract GPR data 'XLINE1.dt1' for processing

```r
x <- mySurvey[[2]]
```

class GPR

process single data file

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

processing

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

processing track

```r
*** Class GPR ***
name = LINE01
filepath = rawGPR/LINE01.DT1
description =
survey date = 2014-04-25
Reflection, 100 MHz, Window
46 traces, 11.25 m

> PROCESSING
  1. coord=
  2. dcshift(u=1:110)
  3. dewow(type=MAD,w=50)
  4. fFilter(f=c(150,260),type=low)
```

keep track of processing

class GPRstack

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

collaborative development on github

Yes, you can... contribute
- share your user experience
- report bugs
- ask questions
- propose modifications on github

develop version, work in progress

class GPRcube

```r
class GPRslice
```

3D processing (e.g., migration, attributes)

```r
class GPRSlice
```

structure delineation etc.

horizontal slice

vertical slice

3D interpolation

transform

back-transform

SVD decomposition

Fourier transform

Hilbert transform etc.

available processing functions

1D processing

<table>
<thead>
<tr>
<th>function</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dewow</td>
<td>DC-shift</td>
</tr>
<tr>
<td>DC-shift</td>
<td>trace average</td>
</tr>
<tr>
<td>trace average</td>
<td>amplitude correction</td>
</tr>
<tr>
<td>amplitude correction</td>
<td>frequency filter</td>
</tr>
<tr>
<td>frequency filter</td>
<td>constant-offset correction</td>
</tr>
<tr>
<td>deconvolution</td>
<td>f-k filter</td>
</tr>
</tbody>
</table>

2D processing

<table>
<thead>
<tr>
<th>function</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2D convolution</td>
<td>2D convolution</td>
</tr>
<tr>
<td>2D convolution</td>
<td>2D adaptive smoothing</td>
</tr>
<tr>
<td>2D adaptive smoothing</td>
<td>Kirchhoff migration</td>
</tr>
</tbody>
</table>

what's next?

short term perspectives
- transform functions
- CMP/velocity modeling
- delineations

long term perspectives
- support for cross-borehole GPR
- forward GPR simulator

WATCHES & JEWELLERY

development

source code + documentation + tutorials

emanuelhuber.github.io/RGPR

easy to install

install `devtools` if not already done

```r
if(!require("devtools")){
  install.packages("devtools")
}
```

devtools::install_github("emanuelhuber/RGPR")

install RGPR from github
easy to install

6th top programming language (IEEE ranking 2017)

read several GPR data files

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

add topographic data

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

plot survey lines

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

3D interactive plot

```r
plot3DRGL(mySurvey)
```

3D interactive plot with open GL

class GPRsurvey

extract GPR data 'XLINE1.dt1' for processing

```r
x <- mySurvey[[2]]
```