ICESat-2 Mission Orbits

This first vignette demonstrates how to download and process time specific orbits. We’ll use one of the Reference Ground Track (RGT) cycles and merge it with other data sources with the purpose to visualize specific areas.

We’ll load one of the latest which is “RGT_cycle_14” (from December 22, 2021 to March 23, 2022). The documentation of the “RGT_cycle_14” data includes more details on how a user can come to the same data format for any of the RGT Cycles.

```r
pkgs = c('IceSat2R', 'magrittr', 'mapview', 'sf', 'rnaturalearth', 'data.table', 'DT', 'stargazer')
load_pkgs = lapply(pkgs, require, character.only = TRUE) # load required R packages

sf::sf_use_s2(use_s2 = FALSE) # disable 's2' in this vignette
mapview::mapviewOptions(leafletHeight = '600px', leafletWidth = '700px') # applies to all leaflet maps
```

# load the 'RGT_cycle_14' data

```r
data(RGT_cycle_14)
res_rgt_many = sf::st_as_sf(x = RGT_cycle_14, coords = c('longitude', 'latitude'), crs = 4326)
```

```r
## Simple feature collection with 131765 features and 6 fields
## Geometry type: POINT
## Dimension: XY
## CRS: EPSG:4326
## First 10 features:
## day_of_year Date hour minute second RGT geometry
## 1 356 2021-12-22 7 57 49 1 POINT (-0.1318472 0.02795893)
## 2 356 2021-12-22 7 58 49 1 POINT (-0.5162124 3.868758)
## 3 356 2021-12-22 7 59 49 1 POINT (-0.901809 7.709809)
## 4 356 2021-12-22 8 0 49 1 POINT (-1.289879 11.55065)
## 5 356 2021-12-22 8 1 49 1 POINT (-1.681755 15.39082)
## 6 356 2021-12-22 8 2 49 1 POINT (-2.078916 19.2299)
## 7 356 2021-12-22 8 3 49 1 POINT (-2.483051 23.06748)
## 8 356 2021-12-22 8 4 49 1 POINT (-2.896146 26.90316)
## 9 356 2021-12-22 8 5 49 1 POINT (-3.3206 30.73662)
## 10 356 2021-12-22 8 6 49 1 POINT (-3.759374 34.56754)
```

ICESat-2 and Countries intersection

We’ll proceed to merge the orbit geometry points with the countries data of the rnaturalearth R package (1:110 million scales) and for this purpose, we keep only the “sovereign” and “sov_a3” columns,
cntr = rnaturalearth::ne_countries(scale = 110, type = 'countries', returnclass = 'sf')

cntr = cntr[, c('sovereign', 'sov_a3')]

cntr

## Simple feature collection with 177 features and 2 fields
## Geometry type: MULTIPOLYGON
## Dimension: XY
## Bounding box: xmin: -180 ymin: -90 xmax: 180 ymax: 83.64513
## CRS: +proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0
## First 10 features:
## sovereign sov_a3 geometry
## 0 Afghanistan AFG MULTIPOLYGON (((61.21082 35...)
## 1 Angola AGO MULTIPOLYGON (((16.32653 -5...)
## 2 Albania ALB MULTIPOLYGON (((20.59025 41...)
## 3 United Arab Emirates ARE MULTIPOLYGON (((51.57952 24...)
## 4 Argentina ARG MULTIPOLYGON (((-65.5 -55.2...)
## 5 Armenia ARM MULTIPOLYGON (((43.58275 41...)
## 6 Antarctica ATA MULTIPOLYGON (((-59.57209 -...)
## 7 France FRA MULTIPOLYGON (((68.935 -48...))
## 8 Australia AUS MULTIPOLYGON (((145.398 -40...)
## 9 Austria AUT MULTIPOLYGON (((16.97967 48...)

We then merge the orbit points with the country geometries and specify also “left = TRUE” to keep also observations that do not intersect with the rnaturalearth countries data,

dat_both = suppressMessages(sf::st_join(x = res_rgt_many,
                               y = cntr,
                               join = sf::st_intersects,
                               left = TRUE))

dat_both

## Simple feature collection with 131765 features and 8 fields
## Geometry type: POINT
## Dimension: XY
## CRS: EPSG:4326
## First 10 features:
## day_of_year Date hour minute second RGT sovereign sov_a3 geometry
## 1 356 2021-12-22 7 57 49 1 <NA> <NA> POINT (-0.1318472 0.02795893)
## 2 356 2021-12-22 7 58 49 1 <NA> <NA> POINT (-0.5162124 3.868758)
## 3 356 2021-12-22 7 59 49 1 Ghana GHA POINT (-0.901809 7.709809)
## 4 356 2021-12-22 8 0 49 1 Burkina Faso BFA POINT (-1.289879 11.55065)
## 5 356 2021-12-22 8 1 49 1 Mali MALI POINT (-1.681755 15.39082)
## 6 356 2021-12-22 8 4 49 1 Algeria DZ POINT (-2.078916 19.2299)
The unique number of RGT's for "RGT_cycle_14" are

\[
\text{length(unique(dat_both$RGT))}
\]

## [1] 1387

We observe that from December 22, 2021 to March 23, 2022,

```r
df_tbl = data.frame(table(dat_both$sovereignt), stringsAsFactors = F)
colnames(df_tbl) = c('country', 'Num_IceSat2_points')

df_subs = dat_both[, c('RGT', 'sovereignt')]
df_subs$geometry = NULL
colnames(df_subs) = c('RGT', 'country')
df_subs = split(df_subs, by = 'country')
df_subs = lapply(df_subs, function(x) {
  unq_rgt = sort(unique(x$RGT))
  items = ifelse(length(unq_rgt) < 5, length(unq_rgt), 5)
  concat = paste(unq_rgt[1:items], collapse = '-')
  iter_dat = data.table::setDT(list(country = unique(x$country),
                                    Num_RGTs = length(unq_rgt),
                                    first_5_RGTs = concat))

  iter_dat
})

df_subs = data.table::rbindlist(df_subs)

df_tbl = merge(df_tbl, df_subs, by = 'country')
df_tbl = df_tbl[order(df_tbl$Num_IceSat2_points, decreasing = T), ]

DT_dtbl = DT::datatable(df_tbl, rownames = FALSE)
```
all RGT’s (1387 in number) intersect with “Antarctica” and almost all with “Russia”.

‘Onshore’ and ‘Offshore’ Points ICESat-2 coverage

The onshore and offshore number of ICESat-2 points and percentages for the “RGT_cycle_14” equal to

```r
num_sea = sum(is.na(dat_both$sovereign))
num_land = sum(!is.na(dat_both$sovereign))

perc_sea = round(num_sea / nrow(dat_both), digits = 4) * 100.0
perc_land = round(num_land / nrow(dat_both), digits = 4) * 100.0
```

dtbl_land_sea = data.frame(list(percentage = c(perc_sea, perc_land),
                           Num_Icesat2_points = c(num_sea, num_land)))

row.names(dtbl_land_sea) = c('sea', 'land')

```r
stargazer::stargazer(dtbl_land_sea,
                      summary = FALSE,
                      rownames = TRUE,
                      header = FALSE,
                      float = FALSE,
                      table.placement = '!h',
                      title = 'Land and Sea Proportions')
```

---

<table>
<thead>
<tr>
<th>percentage</th>
<th>Num_Icesat2_points</th>
</tr>
</thead>
<tbody>
<tr>
<td>sea</td>
<td>67.070</td>
</tr>
<tr>
<td>land</td>
<td>32.930</td>
</tr>
</tbody>
</table>

Global glaciated areas and ICESat-2 coverage

We can also observe the ICESat-2 “RGT_cycle_14” coverage based on the 1 to 10 million large scale Natural Earth Glaciated Areas data,
data(ne_10m_glaciated_areas)

We'll restrict the processing to the major polar glaciers (that have a name included),

```r
ne_obj_subs = subset(ne_10m_glaciated_areas, !is.na(name))
ne_obj_subs = sf::st_make_valid(x = ne_obj_subs)  # check validity of geometries
```

```
## Simple feature collection with 68 features and 5 fields
## Geometry type: MULTIPOLYGON
## Dimension: XY
## Bounding box: xmin: -180 ymin: -89.99993 xmax: 180 ymax: 82.96573
## CRS: 4326
## First 10 features:
## recnum scalerank featurecla name min_zoom
## 143 143 3 Glaciated areas Mount Brown Icefield 2.1
## 148 148 5 Glaciated areas Braithwaite Icefield 5.0
## 152 152 3 Glaciated areas Hooker Icefield 2.1
## 206 206 5 Glaciated areas Homathko Icefield 5.0
## 214 214 6 Glaciated areas Clachnacudainn Icefield 5.7
## 215 215 6 Glaciated areas Albert Icefield 5.7
## 228 228 3 Glaciated areas Plateau Icefield 2.1
## 230 230 5 Glaciated areas Pemberton Icefield 5.0
## 256 256 3 Glaciated areas Cambria Icefield 2.1
## 273 0 3 Glaciated areas Lyell Icefield 2.1
```

and we'll visualize the subset using the `mapview` package,

```r
mpv = mapview::mapview(ne_obj_subs,
            color = 'cyan',
            col.regions = 'blue',
            alpha.regions = 0.5,
            legend = FALSE)
```

mpv
We will see which orbits of the ICESat-2 "RGT_cycle_14" intersect with these major polar glaciers,

```r
res_rgt_many$id_rgt = 1:nrow(res_rgt_many)  # include 'id' for fast subsetting
dat_glac_sf = suppressMessages(sf::st_join(x = ne_obj_subs, y = res_rgt_many, join = sf::st_intersects))
dat_glac = data.table::data.table(sf::st_drop_geometry(dat_glac_sf), stringsAsFactors = F)
dat_glac = dat_glac[complete.cases(dat_glac), ]  # keep non-NA observations
```

<table>
<thead>
<tr>
<th>recnum</th>
<th>scalerank</th>
<th>featurecla</th>
<th>name</th>
<th>min_zoom</th>
<th>day_of_year</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>952</td>
<td>4 Glaciated areas</td>
<td>Jostedalsbreen</td>
<td>3.0</td>
<td>40</td>
</tr>
<tr>
<td>2</td>
<td>1696</td>
<td>3 Glaciated areas</td>
<td>Agassiz Ice Cap</td>
<td>2.1</td>
<td>357</td>
</tr>
<tr>
<td>3</td>
<td>1696</td>
<td>3 Glaciated areas</td>
<td>Agassiz Ice Cap</td>
<td>2.1</td>
<td>358</td>
</tr>
<tr>
<td>4</td>
<td>1696</td>
<td>3 Glaciated areas</td>
<td>Agassiz Ice Cap</td>
<td>2.1</td>
<td>361</td>
</tr>
<tr>
<td>5</td>
<td>1696</td>
<td>3 Glaciated areas</td>
<td>Agassiz Ice Cap</td>
<td>2.1</td>
<td>362</td>
</tr>
<tr>
<td>13245:</td>
<td>0</td>
<td>3 Glaciated areas</td>
<td>Kluane Ice Cap</td>
<td>2.1</td>
<td>42</td>
</tr>
<tr>
<td>13246:</td>
<td>0</td>
<td>3 Glaciated areas</td>
<td>Kluane Ice Cap</td>
<td>2.1</td>
<td>44</td>
</tr>
</tbody>
</table>
We'll split the merged data by the 'name' of the glacier,

```r
dat_glac_name = split(x = dat_glac, by = 'name')

sum_stats_glac = lapply(dat_glac_name, function(x) {
    dtbl_glac = x[, .(name_glacier = unique(name),
                  Num_unique_Dates = length(unique(Date)),
                  Num_unique_RGTs = length(unique(RGT)))]
    dtbl_glac
})
```

```r
sum_stats_glac = data.table::rbindlist(sum_stats_glac)
sum_stats_glac = sum_stats_glac[order(sum_stats_glac$Num_unique_RGTs, decreasing = T), ]
```

The next table shows the total number of days and RGTs for each one of the major polar glaciers,

```r
stargazer::stargazer(sum_stats_glac,
                      summary = FALSE,
                      rownames = FALSE,
                      header = FALSE,
                      float = FALSE,
                      table.placement = 'h',
                      title = 'Days and RGTs')
```
<table>
<thead>
<tr>
<th>name_glacier</th>
<th>Num_unique_Dates</th>
<th>Num_unique_RGTs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Antarctic Ice Sheet</td>
<td>92</td>
<td>1,387</td>
</tr>
<tr>
<td>Greenland Ice Sheet</td>
<td>91</td>
<td>352</td>
</tr>
<tr>
<td>Agassiz Ice Cap</td>
<td>56</td>
<td>58</td>
</tr>
<tr>
<td>Academy of Sciences Ice Cap</td>
<td>34</td>
<td>34</td>
</tr>
<tr>
<td>Manson Icefield</td>
<td>14</td>
<td>19</td>
</tr>
<tr>
<td>Müller Ice Cap</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>Kluane Ice Cap</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>Sydkap Ice Cap</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>Southern Patagonian Ice Field</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Stikine Icecap</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Vestfonna</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Brasvellbreen</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Northern Patagonian Ice Field</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Jostedalsbreen</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

We can restrict to one of the glaciers to visualize the ICESat-2 “RGT_cycle_14” coverage over this specific area ('Southern Patagonian Ice Field'),

```r
sample_glacier = 'Southern Patagonian Ice Field'
dat_glac_smpl = dat_glac_name[[sample_glacier]]
cols_display = c('name', 'day_of_year', 'Date', 'hour', 'minute', 'second', 'RGT')

stargazer::stargazer(dat_glac_smpl[, ..cols_display],
  summary = FALSE,
  rownames = FALSE,
  header = FALSE,
  float = FALSE,
  table.placement = 'h',
  title = 'Southern Patagonian Ice Field')
```

<table>
<thead>
<tr>
<th>name</th>
<th>day_of_year</th>
<th>Date</th>
<th>hour</th>
<th>minute</th>
<th>second</th>
<th>RGT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Southern Patagonian</td>
<td>357</td>
<td>2021-12-23</td>
<td>0</td>
<td>40</td>
<td>43</td>
<td>11</td>
</tr>
<tr>
<td>Southern Patagonian</td>
<td>2</td>
<td>2022-01-02</td>
<td>12</td>
<td>28</td>
<td>4</td>
<td>171</td>
</tr>
<tr>
<td>Southern Patagonian</td>
<td>20</td>
<td>2022-01-20</td>
<td>23</td>
<td>16</td>
<td>46</td>
<td>453</td>
</tr>
<tr>
<td>Southern Patagonian</td>
<td>49</td>
<td>2022-02-18</td>
<td>21</td>
<td>52</td>
<td>48</td>
<td>895</td>
</tr>
<tr>
<td>Southern Patagonian</td>
<td>64</td>
<td>2022-03-05</td>
<td>9</td>
<td>31</td>
<td>50</td>
<td>1,116</td>
</tr>
</tbody>
</table>

and we gather the intersected RGT coordinates points with the selected glacier,

```r
subs_rgts = subset(res_rgt_many, id_rgt %in% dat_glac_smpl$id_rgt)

set.seed(1)
samp_colrs = sample(x = grDevices::colors(distinct = TRUE),
                     size = nrow(subs_rgts))
subs_rgts$color = samp_colrs

ne_obj_subs_smpl = subset(ne_obj_subs, name == sample_glacier)

mpv_glacier = mapview::mapview(ne_obj_subs_smpl,
                                color = 'cyan',
                                col.regions = 'blue',
                                alpha.regions = 0.5)
```
and visualize both the glacier and the subset of the intersected RGT coordinate points (of the different Days) in the same map. The clickable map and point popups include more information,

```r
lft = mpv_glacier + mpv_RGTs
lft
```