

使用fsbrain对FreeSurfer数据进行读写和可视化

Alex / 2023-10-22 / free_learner@163.com / AlexBrain.cn

本文介绍使用R环境下的fsbrain包对FreeSurfer格式的数据进行读写和可视化的基本方法。所有内容参考fsbrain的[官方文档](#)。

一、安装fsbrain和下载测试数据

```
install.packages('fsbrain')
```

在实际安装过程中，根据系统环境不同可能会遇到各种报错，解决方案请参考官方文档。我这里使用的R版本是3.6.3，fsbrain的版本是0.5.3。

fsbrain提供了样例数据，通过下面的方式进行下载：

```
library(fsbrain)
fsbrain::download_optional_data()
```

样例数据包括两个被试（这两个被试数据完全一样，用于测试使用多个被试数据的情形），被试数据按照FreeSurfer (recon-all)的固定文件夹格式进行组织。

```
subjects_dir <- fsbrain::get_optional_data_filepath('subjects_dir')
subjects_list <- c('subject1', 'subject2')
subject_id <- 'subject1'
```

就像使用FreeSurfer的时候，我们通常需要设置 `SUBJECTS_DIR` 和 `SUBJECT` 两个变量，在上面的代码中，`subjects_dir` 表示存放FreeSurfer (recon-all)结果的路径（对应于 `SUBJECTS_DIR`），`subjects_list` 和 `subject_id` 分别表示所有被试文件夹的名字和单个被试文件夹的名字（对应于 `SUBJECT`）。

二、读写数据

`group.morph.native` 用于读取一组被试个体空间（native space）的形态特征数据，比如 `surf/lh.thickness`；`group.morph.standard` 用于读取一组被试标准空间（standard space）的形态特征数据；`write.group.morph.standard.sf` 将所有被试的标准空间的形态特征数据保存为一

个四维的mgz格式的文件，第四个维度表示被试；`group.morph.standard.sf` 用于读取四维的mgz格式的文件；`group.label` 用于读取一组被试的标签（label）文件，比如`label/lh.cortex.label`；`group.annot` 用于读取一组被试的注释（annotation）文件，比如`label/lh.aparc.annot`。

除了上述可以读写一组被试数据的函数，还存在对应的读取单个被试的函数，比如`subject.morph.native`、`subject.label`。fsbrain还提供了一些统计方面的函数，比如根据一个分区模板计算每个脑区的均值，可以使用`group.agg.atlas.native`。fsbrain底层调用`freesurferformats`包来实现各种数据格式的读写，所以如果不是针对FreeSurfer的结果，也许直接用`freesurferformats`包更灵活一些。

```
## Read left-hemi thickness data (lh.thickness) for a group of subjects in native space
groupdata_nat = group.morph.native(subjects_dir, subjects_list, 'thickness', 'lh')
## Read left-hemi thickness data (lh.thickness.fwhm10.fsaverage.mgh) for a group of subjects in
standard space
groupdata_std = group.morph.standard(subjects_dir, subjects_list, 'thickness', 'lh', fwhm='10')
## Write left-hemi thickness data for a group of subjects into one mgz file for future faster
access
write.group.morph.standard.sf('group_thickness_lh_fwhm10.mgz', groupdata_std)
## Read the all-in-one mgz file created in the last step
groupdata_std = group.morph.standard.sf('group_thickness_lh_fwhm10.mgz')
## Read label data (lh.cortex.label) for a group of subjects in native space
grouplabels = group.label(subjects_dir, subjects_list, 'cortex.label', hemi='lh')
## Read annotation data (lh.aparc.annot) for a group of subjects in native space
groupannot = group.annot(subjects_dir, subjects_list, 'lh', 'aparc')
```

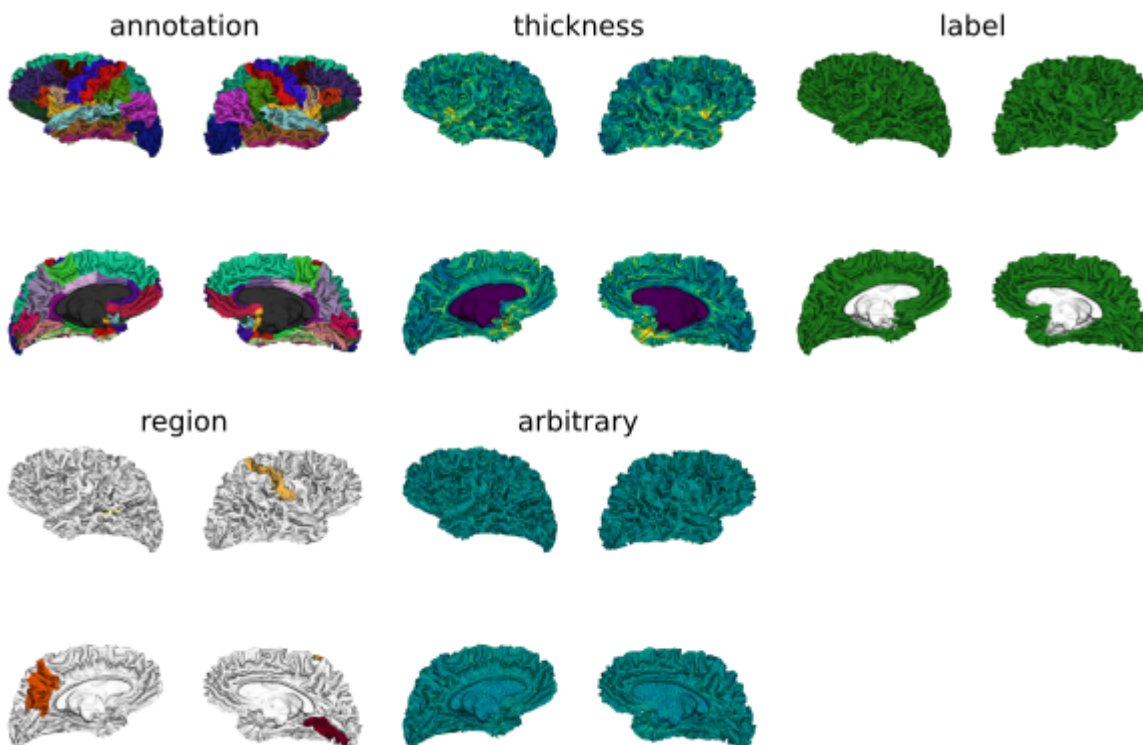
三、可视化

`vis.subject.annot` 用于可视化注释文件；`vis.subject.morph.native` 用于可视化个体空间的形态特征；`vis.subject.label` 用于可视化标签文件；`vis.region.values.on.subject` 用于可视化分区统计结果；`vis.data.on.subject` 用于可视化任意形式的的数据。

```

## Plot parcellation data
vis.subject.annot(subjects_dir, subject_id, 'aparc', 'both', views=c('t4'))
## Plot thickness data
vis.subject.morph.native(subjects_dir, subject_id, 'thickness', hemi='both', views=c('t4'))
## Plot label data
vis.subject.label(subjects_dir, subject_id, 'cortex.label', 'both', views = c('t4'))
## Plot regional values for DK Atlas
lh_region_value_list = list('bankssts'=0.9, 'precuneus'=0.7)
rh_region_value_list = list('postcentral'=0.8, 'lingual'=0.6)
vis.region.values.on.subject(subjects_dir, subject_id, 'aparc', lh_region_value_list,
rh_region_value_list)
## Plot arbitrary data
## Load thickness data and add some random numbers.
lh_data = subject.morph.native(subjects_dir, subject_id, 'thickness', 'lh')
rh_data = subject.morph.native(subjects_dir, subject_id, 'thickness', 'rh')
lh_data = lh_data + rnorm(length(lh_data), mean=10, sd=5)
rh_data = rh_data + rnorm(length(rh_data), mean=10, sd=5)
vis.data.on.subject(subjects_dir, subject_id, lh_data, rh_data, views=c('t4'))

```



通常需要一些参数调试才能实现自己预期的可视化效果（也可能实现不了），fsbrain提供了一个 `export` 函数可以生成一个布局更紧凑、高分辨率、加上了colorbar和图例的图，也许可以直接用在论文中。

```
## Use vis.subject.morph.native as an example, export could be used for other visualization
functions as well
coloredmeshes = vis.subject.morph.native(subjects_dir, subject_id, 'thickness', 'both',
                                         makecmap_options=list('colFn'=squash::jet),
                                         rgloptions=rgloptions, rglactions=list('no_vis'=T))
fig = export(coloredmeshes, colorbar_legend='Cortical Thickness [mm]', output_img='fig1.png')
```

