Visualisation of Brain Statistics with R-package `ggseg`

Athanasia M. Mowinckel[1] **and Didac Vidal Piñeiro**[1]

[1]**Center for Lifespan Changes in Brain and Cognition, Univeristy of Oslo, PO. box 1094 Blindern, 0317 Oslo, Norway**

Corresponding author:
Athanasia M. Mowinckel[1]

Email address: `a.m.mowinckel@psykologi.uio.no`

ABSTRACT

The abstract of the article. It can also be on *multiple* lines.

1 INTRODUCTION

Visualization is increasingly important for accurate guidance and interpretation of neuroimaging results, as current research is able to generate a high amount of data and outcomes. For Magnetic Resonance Imaging (MRI), neuroimaging software provides whole-brain information by using many small units of space ($>$100.000). Nonetheless, this data is often grouped and summarized into a limited number of regions using predefined brain parcellation atlases. Brain parcellations segment the brain into a finite set of meaningful neurobiological components, which reflect one or more brain features either based on local/structural or connectivity properties (Eickhoff et al. [2018]). The use of brain atlases is widespread as these facilitate interpretation and minimize the amount of data, hence reducing problems with multiple comparisons and facilitating replicability and data sharing in otherwise computationally expensive analyses. Complex analyses and post-processing is often performed in specialized software environments such as R (R Core Team [2019]).

MRI data provides good spatial resolution and thus an optimal representation has to respect spatial relationships across regions. Results from brain atlas analyses are most meaningfully visualized when projected onto a representation of the brain, rather than on other types of graphs (e.g. bar charts). The projection of data into brain representations provides clear points of reference - especially when the reader is unfamiliar with the atlas - eases readability, guides interpretation, and informs on the spatial patterns of the data. Adopting the grammar of graphics implemented in ggplot2 (Wickham [2016]), one can plot neuroimaging data directly in R with several tools such as ggBrain (Fisher [2019]) and ggneuro (Muschelli [2017]; see neuroconductor [2018] for curated neuroimaging packages for R). Yet these tools display whole-brain image files and are not suited for representing brain atlas data.

In this tutorial, we introduce the ggseg-package for visualizing brain atlas data in R. The ggseg – and the complimentary ggsegExtra – package include pre-compiled data sets for different brain atlases that allow for 2D and 3D visualization. Two-dimensional functionality is based on polygons and ggplot-based grammar of graphics (Wickham [2016]), while the 3D functionality is based on tri-surface mesh plots and plotly (Sievert [2018]).

The ggseg-package presents both compiled data sets, tailored scripts to allow brain data integration and plotting and, other minor features such as custom color palettes. The data featured are derived from two well-known parcellations: the Desikan-Killany cortical atlas (DKT; Desikan et al. [2006]), which covers the cortical surface of the brain, and the Automatic Segmentation of Subcortical Structures (aseg; Fischl et al. [2002]), which covers the subcortical structures. Both atlases are implemented in several neuroimaging softwares, such as FreeSurfer (Fischl et al. [1999], Dale et al. [1999], Fischl and Dale [2000]), and are commonly used in relation to developmental changes, disease biomarkers, genomic data, and cognition (Amlien et al. [2019], Walhovd et al. [2005], Pizzagalli et al. [2009]). The ggsegExtra-package currently includes $>$13 additional precompiled atlases and it is continuously expanding. See

Table 1. Table of currently available atlases in either ggseg or ggsegExtra R-package. Most atlases have both polygon and mesh atlases, but the mesh atlases are somewhat easier to create and are thus more plentiful.

Package	Title	Item	Mesh	Polygon
ggseg	Desikan-Killiany Cortical Atlas	dkt	Yes	Yes
	Freesurfer automatic subcortical segmentation of a brain volume	aseg	Yes	Yes
ggsegExtra	Desterieux cortical parcellations	desterieux	Yes	No
	Genetic topography of brain area morphology	chenAr	No	Yes
	Genetic topography of brain thickness morphology	chenTh	No	Yes
	Harvard-Oxford Cortical atlas	hoCort	No	Yes
	Parcellation from a midsagittal slice	midsagittal	No	Yes
	Parcellation from JHU	jhu	Yes	Yes
	Parcellation from of white matter	icbm	Yes	No
	Parcellation from the Human Connectome Project	glasser	Yes	Yes
	Schaefer 17 Resting-state Cortical Parcellations	schaefer17	Yes	No
	Schaefer 7 Resting-state Cortical Parcellations	schaefer7	Yes	No
	White matter tract parcellations	tracula	Yes	Yes
	Yeo 17 Resting-state Cortical Parcellations	yeo17	Yes	Yes
	Yeo 7 Resting-state Cortical Parcellations	yeo7	Yes	Yes

Table 1 for a summary of the available atlases.

We encourage users to contribute to the ggsegExtra brain atlas repository. In SI text and github wiki, we offer a pipeline to create and supply atlases for 2D plotting. See SI text and SI scripts for instruction to include atlases for 3D plotting.

2 TUTORIAL

This tutorial will introduce the `ggseg` and `ggsegExtra` packages and familiarize the reader with the main functions and the general use of the packages. The tutorial will cover the following functions: `ggseg()` for plotting 2D polygons and `ggseg3d()` for plotting 3D brains based on tri-surface mesh plots.

2.1 Plotting polygon data (ggplot2)

`ggseg` is the main function for plotting 2D data. By default, the function automatically plots the DKT atlas (see Figure 1). The `ggseg` function is a wrapper for `geom_polygon` from `ggplot2`, and it can be built upon and combined like any ggplot object. The image plot consists of a simple brain representation containing no extra information. Hence, `ggseg` plots can be easily complemented with any of the available ggplot features and options. We recommend users to get familiarized with ggplot2 grammar (Wickham [2016]).

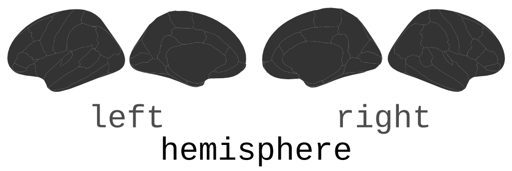


Figure 1. By default `ggseg` will plot the dkt atlas in shaded polygons.

```r
library(ggseg)
library(tidyverse)
ggseg()
```

`ggseg` is implemented as a fully-functional part of ggplot. Thus one should be able to use the function together with any other ggplot function such as themes or scales. You can stack the hemispheres, view only the medial or lateral side, choose either or both of the hemispheres, or a combination of any of these. One has several options for plotting the main brain representations. These options are atlas-specific. For cortical atlases, such as the 'dkt', one can stack the hemispheres, plot one or both hemisphere, or view only the medial or lateral side (or any combination of these; see Figure 2). For subcortical atlases, such as the 'aseg', the options are more limited but one can often choose between axial, sagital, and coronal views.

```r
# dkt dark theme
p1 <- ggseg(position = "stacked") +
  theme_dark() +
  labs(title=" ")

# dkt classic theme
p2 <- ggseg(position = "stacked") +
  theme_classic() +
  labs(title = " ")

# dkt medial view
med <- ggseg(view = "medial") +
  labs(title = " ")

# dkt left hemisphere
left <- ggseg(hemisphere = "left") +
  labs(title = " ")


# aseg default theme
p3 <- ggseg(atlas=aseg) +
  labs(title = " ")


# dkt left medial alone
combo <- ggseg(view = "medial",
      hemisphere = "left") +
  labs(title=" ")

# Combine plots
cowplot::plot_grid(p1, med, combo,  p2, left, p3,
                   labels = c("A: dkt - dark", "B: dkt - medial",
                              "C: dkt - combo", "D: dkt - classic",
                              "E: dkt - left", "F: aseg"),
```

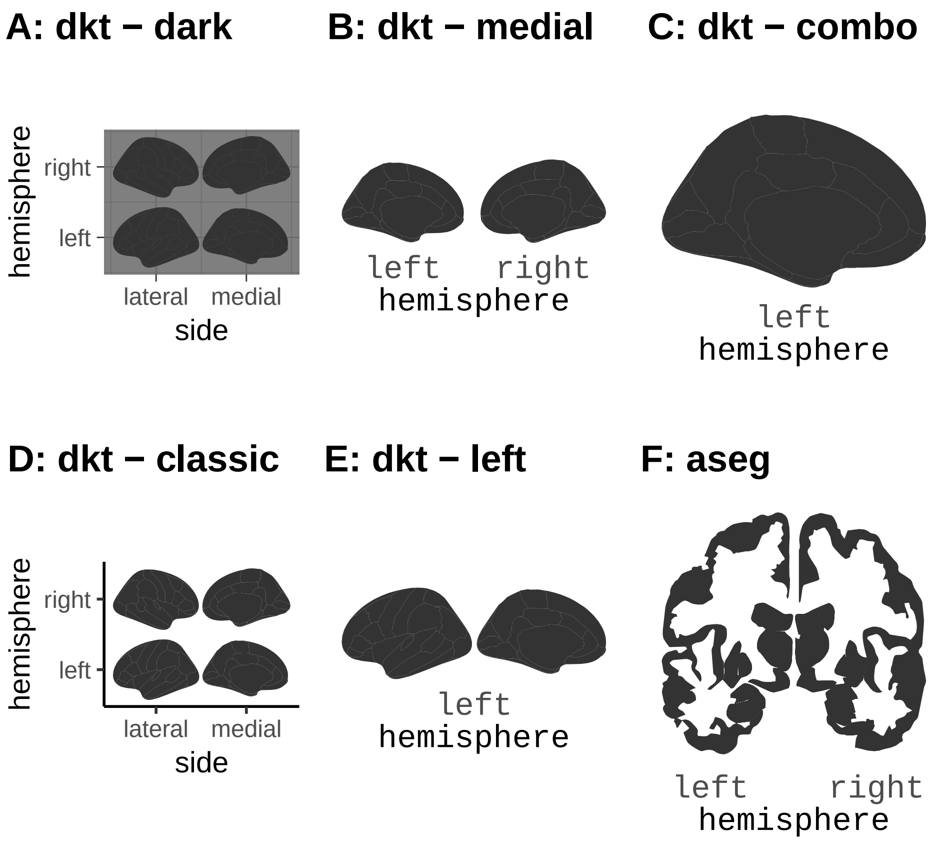


Figure 2. ggseg plots can be used with most standard scales, themes and such that work with ggplot. The special ggseg options for hemispheres, view etc. depend on the atlas used, and some options are only available for certain atlases. There is no 'lateral' or 'medial' views of subcortical atlases. There are several ggseg-special options that may be supplied to control how the plots looks and is organised. **A:** dkt atlas, stacked with dark theme ; **B:** dkt with medial view only; **C:** dkt atlas with only left medial display; **D:** dkt atlas, stacked, with classic theme; **E:** dkt atlas with left hemisphere only; **F:** complete aseg atlas

```
                   hjust = -.05)
```

2.1.1 Using own data with fill and colour

`ggseg` accepts any argument you can supply to `geom_polygon` and therefore is easy to work with for those familiar with ggplot functionality. Standard arguments like `fill` to flood the segments with a colour, or `colour` to colour the edges around the segments are typical arguments to provide to the function either as a single value or within the ggplot mapping option `aes`. To use color palettes corresponding to those used in the original neuroimaging softwares one can use atlas-specific 'brain' palette scales. See an example in Figure 3.

```
ggseg(mapping=aes(fill = area), colour="black") +
  scale_fill_brain("dkt") +
  theme(legend.justification=c(1,0),
        legend.position="bottom",
        legend.text = element_text(size = 5)) +
  guides(fill = guide_legend(ncol = 3))
```

Most users will use ggseg to display - using a color scale - some descriptive or inferential statistics, such as mean thickness or cognition relationships etc., across the different brain regions. The bulk of the package originates not in its functions, but in the data sets that accompany the package. There is one

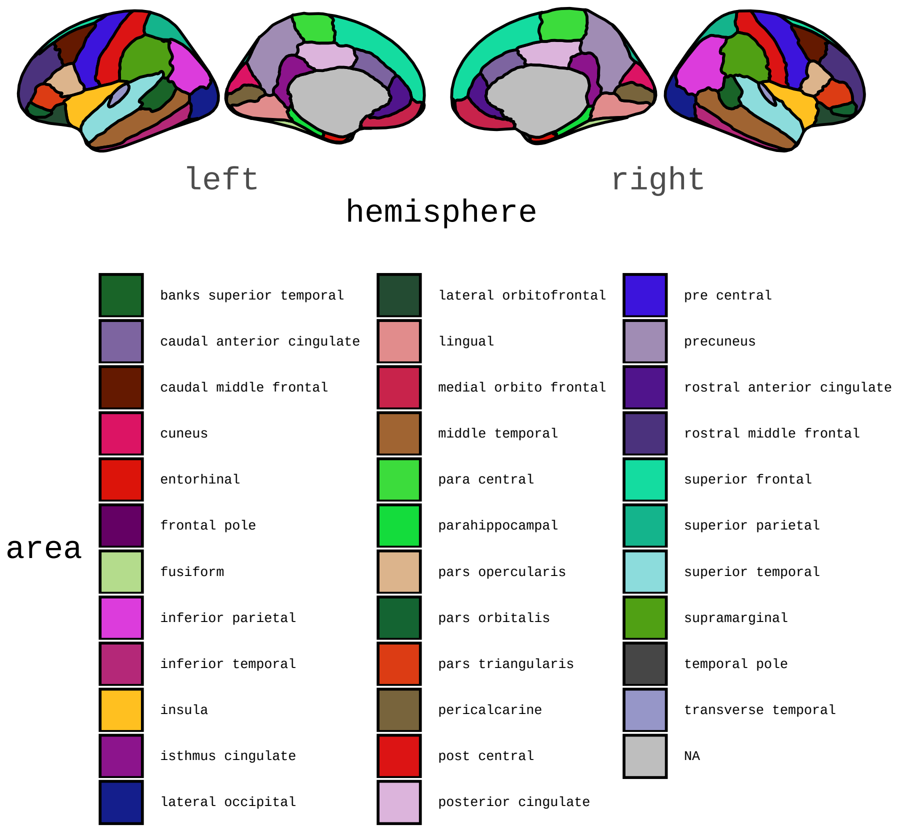


Figure 3. Supplying ‘area’ to the fill option in ggseg, will use the column ‘area’ from the accompanying dataset to create a discrete colour palette over the segments in the atlas.

data set per atlas, which contains key information regarding the atlas, and coordinates for the segment polygons of the atlas. Before setting up the data from the statistics to project onto the segments, looking at the atlas data sets might help understand what the data needs to look like, or avoid looking like.

```
dkt
```

```
## # A tibble: 80 x 6
##    atlas area                hemi  side    label              ggseg
##    <chr> <chr>               <chr> <chr>   <chr>              <list>
##  1 dkt   superior temporal   left  lateral lh_superiortempo~ <tibble [1,494~
##  2 dkt   pre central         left  lateral lh_precentral      <tibble [1,314~
##  3 dkt   post central        left  lateral lh_postcentral     <tibble [1,164~
##  4 dkt   rostral middle fr~ left  lateral lh_rostralmiddle~ <tibble [1,194~
##  5 dkt   insula              left  lateral lh_insula          <tibble [870 x~
##  6 dkt   superior parietal   left  lateral lh_superiorparie~ <tibble [480 x~
##  7 dkt   inferior temporal   left  lateral lh_inferiortempo~ <tibble [906 x~
##  8 dkt   lateral occipital   left  lateral lh_lateraloccipi~ <tibble [738 x~
##  9 dkt   lateral orbitofro~ left  lateral lh_lateralorbito~ <tibble [528 x~
## 10 dkt   superior frontal    left  lateral lh_superiorfront~ <tibble [420 x~
## # ... with 70 more rows
```

In any atlas, the column ‘label’ is particularly useful for combining the data of interest with the `ggseg`-polygons. The column ‘label’ contains the label (region) names as in the original neuroimaging software. For example, the DKT atlas label column matches the region names from Freesurfer statistics tables. Yet the data in `ggseg` is in a long format - that is each region has its own row - and data of interest needs to be in this same format. Most data sets are organized in wide format, in which subjects are represented by rows and each different data variable is represented in a separate column, and thus need to be rearranged in order to work with ggseg. See below an example of wide-to-long conversion.

```
freesurfer_stats <- data.frame(
  id = c(10:12),
  lh_superiortemporal = c(3.32, 4.1, 3.5),
  lh_precentral = c(2.3, 2.5, 2.1),
  lh_rostralmiddlefrontal = c(3.3, 3.2, 3.1)
  )
freesurfer_stats
```

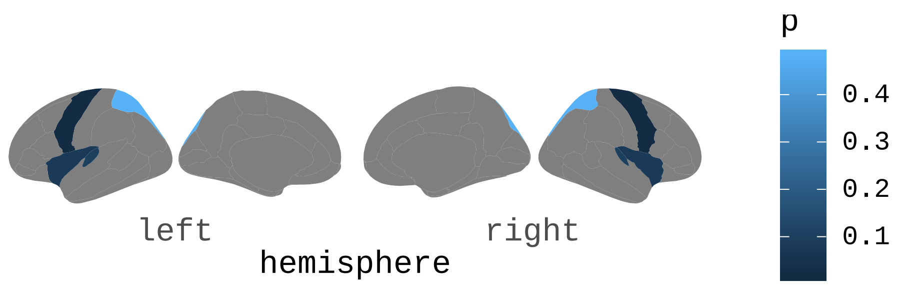
```
##   id lh_superiortemporal lh_precentral lh_rostralmiddlefrontal
## 1 10                3.32           2.3                     3.3
## 2 11                4.10           2.5                     3.2
## 3 12                3.50           2.1                     3.1
```

```
freesurfer_long <- freesurfer_stats %>%
  gather(label, thickness, -id)
freesurfer_long
```

```
##   id                   label thickness
## 1 10     lh_superiortemporal      3.32
## 2 11     lh_superiortemporal      4.10
## 3 12     lh_superiortemporal      3.50
## 4 10           lh_precentral      2.30
## 5 11           lh_precentral      2.50
## 6 12           lh_precentral      2.10
## 7 10 lh_rostralmiddlefrontal      3.30
## 8 11 lh_rostralmiddlefrontal      3.20
## 9 12 lh_rostralmiddlefrontal      3.10
```

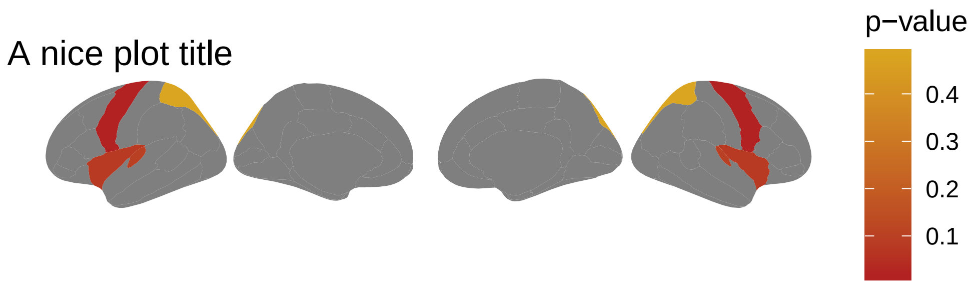
This data - in long format - can now be used directly with the `ggseg` function, as the `label` column corresponds in name and content with the `label` column in the atlas data of dkt. The data **must** include a column that has the same name and at least *some* data matching the values in the corresponding column in the atlas data. In the next example we create some data with 4 rows, and an 'area' and 'p' column, representing the results of an analysis. The `ggseg` function will recognise the matching column 'area', and merge the supplied data into the atlas using `dplyr` joins. We use the `p` column as the column flooding the segment with colour.

```
someData = data.frame(
  area = c("transverse temporal", "insula",
           "pre central","superior parietal"),
  p = sample(seq(0,.5,.001), 4),
  stringsAsFactors = FALSE)

p <- ggseg(.data=someData, mapping=aes(fill=p))
p
```



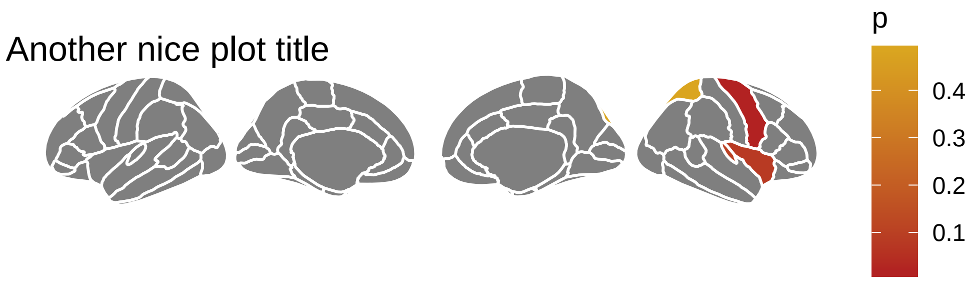
The appearance of this plot can then be modified similarly to any other ggplot graph using functions such as scales, labs, themes, etc.

```
p +
  theme_void() +
  scale_fill_gradient(low="firebrick",high="goldenrod") +
  labs(title="A nice plot title", fill="p-value")
```



If the results are only in one hemisphere, but you still want to plot both of them, make sure your data.fame includes the column `hemi` with either 'right' or 'left' for this to happen.

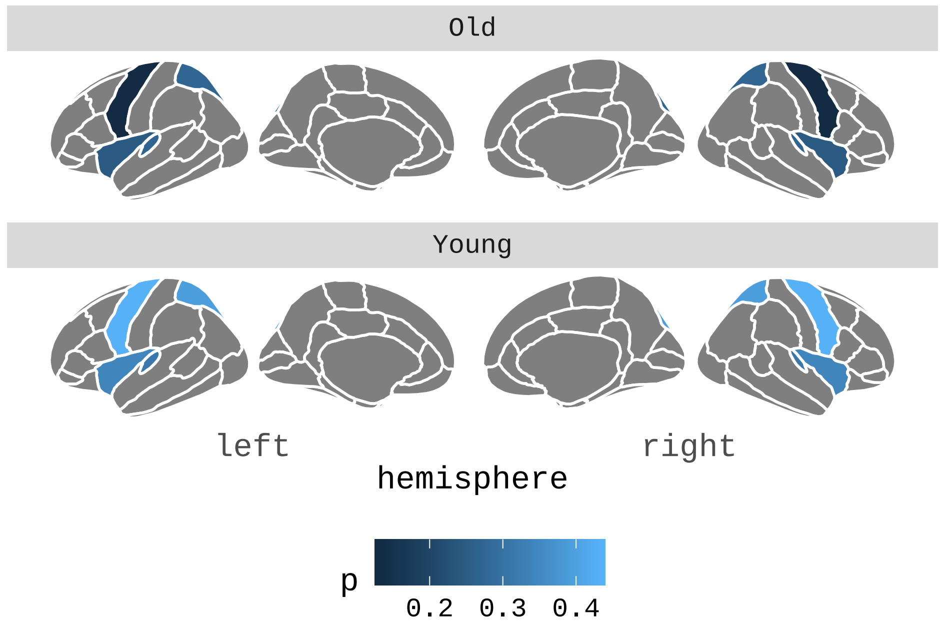
```
someData$hemi = "right"

ggseg(.data=someData, colour="white",mapping=aes(fill=p)) +
  theme_void() +
  scale_fill_gradient(low="firebrick",high="goldenrod") +
  labs(title="Another nice plot title")
```



2.1.2 Creating subplots

There is often the need to plot a statistic of interest in different groups (i.e. thickness or significance in young or older adults). This may be obtained also with ggseg, using ggplot's `facet_wrap` or `facet_grid`, using two guiding rules: **1)** as before, data needs to be in long format (group data should appear in seperate rows, not in separate columns). **2)** The data needs to be grouped using `dplyr`'s `group_by` function *before* providing the data to the `ggseg` function. The ggseg function will detect grouped data, and adapt it to `facet`'s requirements.

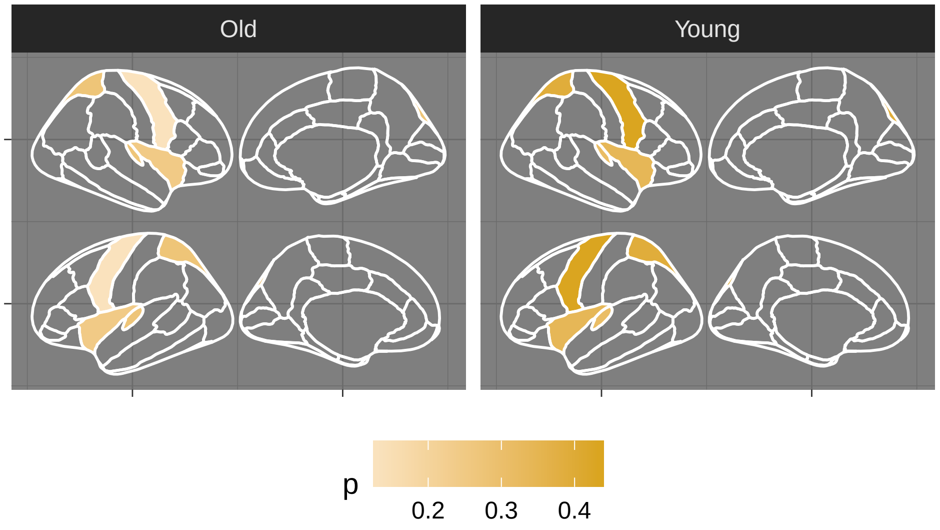
```
someData = data.frame(
  area = rep(c("transverse temporal", "insula",
               "pre central","superior parietal"),2),
  p = sample(seq(0,.5,.001), 8),
  AgeG = c(rep("Young",4), rep("Old",4)),
  stringsAsFactors = FALSE) %>%
  group_by(AgeG)

ggseg(.data=someData, colour="white", mapping=aes(fill=p)) +
  facet_wrap(~AgeG, ncol=1) +
  theme(legend.position = "bottom")
```



As before, one can apply the different ggplot and facet options, such as `scales` to modify the plot's appearance.

```
ggseg(.data = someData, atlas=dkt,
      colour="white", position="stacked",
      mapping=aes(fill=p)) +
  facet_wrap(~AgeG, ncol=2) +
```

```r
  theme_dark() +
  theme(legend.position = "bottom",
        axis.text = element_blank(),
        axis.title = element_blank()
        ) +
  scale_fill_gradient2(high="goldenrod",
                       low="firebrick",
                       na.value="transparent")
```



All the concepts described above also work with the ‘aseg’ atlas for subcortical structures, except hemisphere and view arguments that do not apply in the same way in subcortical atlases such as aseg. Again, inspecting the atlas data directly will be of aid when preparing data to use with the atlas.

```r
aseg
```

```
## # A tibble: 34 x 6
##    atlas area            hemi  side  label                ggseg
##    <chr> <chr>           <chr> <chr> <fct>                <list>
##  1 aseg  <NA>            right axial <NA>                 <tibble [808 x 5~
##  2 aseg  <NA>            left  axial <NA>                 <tibble [1,014 x~
##  3 aseg  thalamus proper left  axial Left-Thalamus-Proper <tibble [56 x 5]>
##  4 aseg  thalamus proper left  axial Right-Thalamus-Prop~ <tibble [56 x 5]>
##  5 aseg  thalamus proper right axial Left-Thalamus-Proper <tibble [61 x 5]>
##  6 aseg  thalamus proper right axial Right-Thalamus-Prop~ <tibble [61 x 5]>
##  7 aseg  lateral ventri~ right axial Left-Lateral-Ventri~ <tibble [66 x 5]>
##  8 aseg  lateral ventri~ right axial Right-Lateral-Ventr~ <tibble [66 x 5]>
##  9 aseg  hippocampus     left  axial Left-Hippocampus     <tibble [47 x 5]>
## 10 aseg  hippocampus     left  axial Right-Hippocampus    <tibble [47 x 5]>
## # ... with 24 more rows
```

```r
ggseg(atlas="aseg", mapping=aes(fill=area))
```

2.2 Plotting 3D mesh data

Representing brains as 2D polygons is a good solution for fast, efficient, and flexible plotting, and can be easily combined with interactive apps such as Shiny (Chang et al. [2019]). Yet, brains are intrinsically 3-dimensional and it can be challenging to recognize the location of a region as a flattened image. This problem is exacerbated in atlases that represent subcortical features – after all, cortical surfaces are 2-dimensional – such as grey matter structures or white matter tracts. Hence, here we also provide the `ggseg3d` function to plot, view, and print 3D-atlases in R. `ggseg3d` is based on tri-surface mesh plots

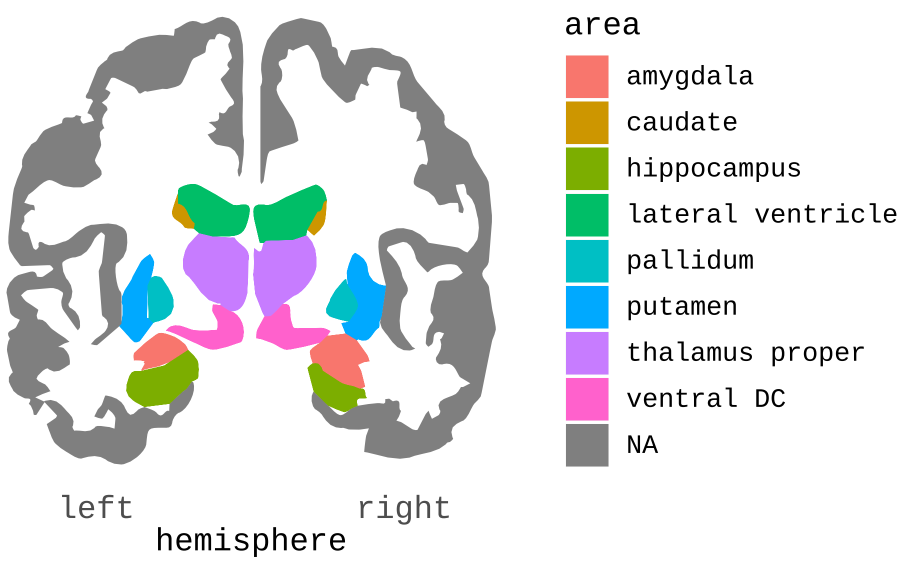


Figure 4. The first 10 rows of the `aseg` dataset, that has some specific differences from the dkt, like missing the option to not view both hemispheres, as the plot would be hard to understand that way.

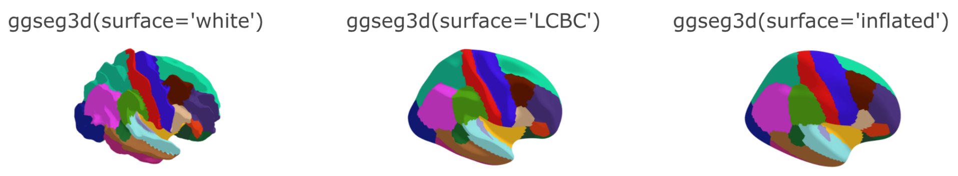


Figure 5. The three surface options provided in ggseg 3d atlases. From left: the 'white' surface is the white matter surface, 'LCBC' surface is the white matter surface inflated in 10 steps, and the 'inflated' surface is a fully inflated sortical surface.

using plotly (Sievert [2018]). The data structure is somewhat more complex than the ggplot polygons, and includes additional options for brain inflation, glass brains, camera locations, etc. As `ggseg3d` is based on plotly, the resulting brain atlases are interactive, which guides interpretation, and is useful for public dissemination. We recommend users to familiarize themselves with plotly (Sievert [2018]) when using this function.

Out-of-the-box, `ggseg3d()` plots the `dkt_3d` atlas in 'LCBC' surface. The 'LCBC' surface consists on a semi-inflated white matter surface based on the *fsaverage* template subject. See *SI SCRIPT* for reproducibility. All `[...]_3d` atlases have a built in `colour` column for default colour plotting of the regions, based on the colorlut used in the neuroimaging software.

The 3D-atlas data is stored in nested tibbles. Each cortical atlas has data sets for three different surfaces (see Figure @ref(fig:ggseg3d_1_out)) and the two hemispheres. As there are no meaningful other surfaces for subcortical atlases, these only have a single surface. The 'ggseg_3d' column includes all necessary information to `ggseg3d()` to create a mesh-plot. The additional 3D-atlases in ggsegExtra have the same data structure.

```
dkt_3d
```

```
## # A tibble: 6 x 4
##   atlas  surf     hemi  ggseg_3d
##   <chr>  <chr>    <chr> <list>
## 1 dkt_3d inflated left  <tibble [36 x 8]>
## 2 dkt_3d inflated right <tibble [36 x 8]>
## 3 dkt_3d LCBC      left  <tibble [36 x 8]>
## 4 dkt_3d LCBC      right <tibble [36 x 8]>
## 5 dkt_3d white     left  <tibble [36 x 8]>
## 6 dkt_3d white     right <tibble [36 x 8]>
```

2.2.1 External data supply

Similarly as in the 2D-atlas, the user will use `ggseg3d` to display through a color scale some descriptive or inferential statistics. If the data is not already in the correct long format, or uses similar naming as the atlas, one needs to grab the data for a specific surface (and hemisphere, if desired) and then `unnest(ggseg_3d)` it to see what the atlas is expecting.

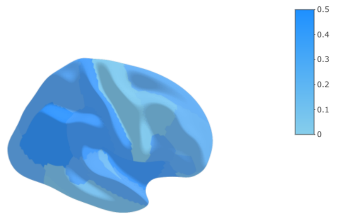
```
dkt_3d %>%
  filter(surf == "inflated" & hemi == "right") %>%
  unnest(ggseg_3d) %>%
  select(-lobe, -acronym)
```

```
## # A tibble: 36 x 9
## atlas surf hemi area colour mesh label roi annot
## <chr> <chr> <chr> <chr> <chr> <list> <chr> <chr> <chr>
## 1 dkt_3d infla~ right <NA> <NA> <list~ rh_medial~ 0001 medialwa~
## 2 dkt_3d infla~ right banks supe~ #1964~ <list~ rh_bankss~ 0002
bankssts
## 3 dkt_3d infla~ right caudal ant~ #7D64~ <list~ rh_caudal~ 0003
caudalan~
## 4 dkt_3d infla~ right caudal mid~ #6419~ <list~ rh_caudal~ 0004
caudalmi~
## 5 dkt_3d infla~ right corpus cal~ <NA> <list~ rh_corpus~ 0005
corpusca~
## 6 dkt_3d infla~ right cuneus #DC14~ <list~ rh_cuneus 0006 cuneus
## 7 dkt_3d infla~ right entorhinal #DC14~ <list~ rh_entorh~ 0007
entorhin~
## 8 dkt_3d infla~ right fusiform #B4DC~ <list~ rh_fusifo~ 0008
fusiform
## 9 dkt_3d infla~ right inferior p~ #DC3C~ <list~ rh_inferi~ 0009
inferior~
## 10 dkt_3d infla~ right inferior t~ #B428~ <list~ rh_inferi~ 0010
inferior~
## # ... with 26 more rows
```

Note the `mesh` column, which contains lists. Each list corresponds to a region and contains 6 vectors required to create the mesh of the tri-surface plot. It should also be noted that the ‘label’, ‘annot’ and ‘area’ columns could provide matching values for your own data. Similarly to the `ggseg`-function, the ‘label’ column should match the region names used in the original neuroimaging software while ‘area’ and ‘annot’ provide alternative/secondary names. It is thus important to match your regional identifiers with those used in the atlas. A warning will be issued in case of any mismatch. The column you want to use for colour, needs to be supplied to the `colour` option, and you’ll likely want to supply it to the `text` option, as this will add another line to the plotly hover information.

```
someData = dkt_3d %>%
  filter(surf == "inflated" &
         hemi == "right") %>%
  unnest(ggseg_3d) %>%
  select(area) %>%
  na.omit() %>%
  mutate(p = sample(seq(0,.5, length.out = 100 ), nrow(.)) %>%
           round(2))
```

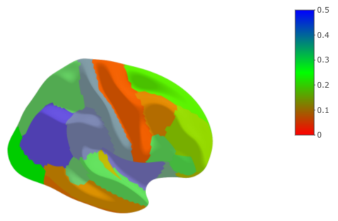
```r
ggseg3d(.data = someData, atlas = dkt_3d, colour = "p", text = "p")
```



2.2.2 Colours

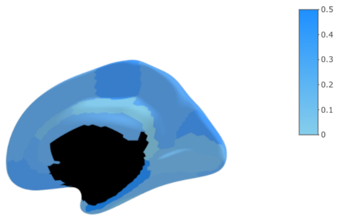
You can provide custom colour palettes either in hex or R-names. Colours will be evenly spaced when creating the colour-scale, and you may provide as many colours as you wish.

```r
ggseg3d(.data = someData, atlas = dkt_3d,
        colour = "p", text = "p",
        palette = c("#ff0000", "#00ff00", "#0000ff"))
```



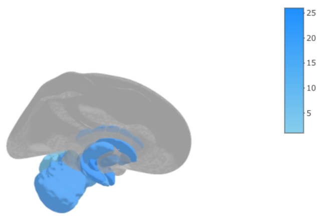
If you want to alter the colour of `NA` regions, supply `na.colour`, either as HEX colour or colour name. This option only takes a single colour.

```r
ggseg3d(.data = someData, atlas = dkt_3d,
        colour = "p", text = "p",
        na.colour = "black", camera="medial")
```



Subcortical atlases include cortical surfaces and other landmark structures for visualization purposes only. One can control the opacity of the these `NA` structures, to improve visualization. Additionally, one can add the `glassbrain` option, which provides a frame of reference for the subcortical structures. Glassbrains can be controlled with three different arguments: opacity (0-1), hemisphere (left, right), and colour.

```
somData_aseg = aseg_3d %>%
  unnest() %>%
  select(label) %>%
  filter(!grepl("Ventricle|Putamen", label)) %>%
  mutate(p = seq(1, nrow(.)))

  ggseg3d(.data = somData_aseg, atlas = aseg_3d,
          colour = "p", text = "p",
          na.alpha= .5, glassbrain = .5,
          glassbrain_hemisphere = "left")
```



ggsed3d() is based on plotly and thus additional plotly functionalities can be used to modify and improve the 3D atlas representations. In addition to Carson Sievert's book on plotly in R ([2018]), we recommend resources for modifying axes in 3D plots (unknown [unknowna]), the basic introduction to tri-surface plots (unknown [unknownb]), and this tutorial on tri-surface plots with plotly in R (Riddihiman [2016]). Finally, we recommend orca command line tool to save ggseg3d atlas snapshots.

3 DISCUSSION

The main aim of the ggseg – and ggsegExtra - package is to ease and streamline visualization of brain atlas data in R, by gathering a collection of atlases from several scientific sources and providing plotting functions adapted to these atlases. In this tutorial, we introduce the package to the readers by presenting some use examples and highlighting the main functions and options that are available. As a visualization tool, this package adds up to manifold functionalities such as ggBrain (Fisher [2019]) and ggneuro (Muschelli [2017]) in R, and software-specific image viewers such as FSLeyes (McCarthy [2019]) and Freeview (Dale et al. [1999]). In this regard, we do not aim to compete with software-specific visualizations or advocate for the superiority of ggseg as a visualization tool. After all, flattened 2D polygons do not rely on a meaningful brain coordinate system and the units of information in 3D meshes are limited to the number of parcellations. On the contrary, we believe the ggseg niche among visualization tools resides in its simplicity and its ability to be combined with statistical analysis pipelines. The possibility to serve as an interactive tool for dissemination and reproducibility when combined with other technologies, such as Binder (et al. [2018]) or Shiny (Chang et al. [2019]), is an added benefit.

The ggseg and ggsegExtra packages contain three main features: **1)** a collection of 2D-polygon and 3D mesh brain parcellation atlases. The atlas data are necessary to create the coordinate system for the brain atlas plotting, but are made in such a way to make i easy to use for ones own purposes. **2)** `ggseg()` and `ggseg3d()` functions for visualization. `ggseg()` as a wrapper for `geom_polygon` from `ggplot2` and it can be built upon like any ggplot object. `ggseg3d()` as a plotly wrapper for tri-surface mesh plots which prints 3D atlases. Both functions are flexible and well-adapted to its environment as can be combined with any additional argument from ggplot and plotly, respectively. **3)** Complimentary features – e.g. color scales - and functions such as `as_ggseg_atlas()` and `as_ggseg3d_atlas()` to convert data in the correct atlas format. These function should provide users with the possibility of adapting their plots to their wishes, and also to make it possible to create and contribute to the atlas repository.

The foundations of the ggseg-package trace back to the necessity of visualizing and exploring the lifespan trajectories of cortical thickness across different brain regions (see supporting information in Vidal-Piñeiro et al. ([2019])) . That is, ggseg appears with the need to inspect and display 4D information - including a spatial dimension - overcoming the constrains of printed journals and classical 2D plots (e.g. bar plots). The current state of science requires researches to share the results of studies in both high detail and in an intuitive manner, as it permits communication to wide audiences and facilitates reproducibility. Hence, we believe this tool conforms to the essence of open science and invite users to improve the code, provide examples, or tutorials, and contribute to the atlas collection according to their own interest and needs via the public ggseg GitHub repository and ggsegExtra GitHub repository. See X and Y for detailed guidelines on how to create new atlases. Finally - while the ggseg-package is circumscribed to brain parcellations - we believe that the structure and functions of the package can be easily applied to any scientific field that benefits from data being displayed across the spatial dimension. We encourage readers to borrow the package functionalities and adapt it to their respective fields and structures of interest, such as has already been done with the gganatogram-package (Maag [2018]).

4 CONCLUSION

Visualization is a fundamental aspect of neuroimaging to explore and understand data, guide interpretation and, communicate with colleagues and general audience. In this tutorial, we have introduced the ggseg-package, a tool for visualizing brain statistics through brain parcellation atlases in R. This visualization tool easily combines with interactive routines as well as with diverse statistical analysis pipelines. We hope this tool and tutorial proves useful to neuroscientists and inspires others to apply the functions in a wide variety of fields and structures.

5 AUTHOR CONTRIBUTIONS

Didac Vidal-Piñeiro generated the idea for the tool, and the initial scripts that made the plots. He has also been responsible for converting images from neuroimaging software into the type of data necessary for plotting polygons and mesh plots in R. Athanasia M. Mowinckel adapted the initial scripts and made the functions into package format and has continued developing the functions with the aim of increasing user-friendliness. She is also responsible for conceiving and adding the mesh-plot functionality through plotly, and developing the pipeline for making that possible. A. M. Mowinckel wrote the first draft of the paper, and both have since critically edited it.

6 CONFLICTS OF INTEREST

The authors declare that there were no conflicts of interest with respect to the authorship or the publication of this article.

7 ACKNOWLEDGEMENTS

We wish to thank John Muschelli for his initial improvements in the package code and preparing it for incorporation in neuroconductor ([2018]), and Richard Beare for submitting the first community created atlas, and providing helpful comments to the base code. A. M. Winkler, who originally introduced us to the procedure of converting 3D brain images from Freesurfer into `.ply` files, also needs to be credited with this, we would not have managed to create the mesh-plots without his scripts. We also thank Anders M. Fjell and Kristine B. Walhovd for allowing us to pursue the creation of this tool and encouraging our progress. We are indebted to those users that actively contributed to the package and all the scientists and engineers that worked on the framework on which the package is sustained (e.g. R-project, neuroimaging software).

8 FUNDING

This work is funded by EU Horizon 2020 Grant ‘Healthy minds 0-100 years: Optimizing the use of European brain imaging cohorts (Lifebrain)’, with grant agreement number 732592.

9 PRIOR VERSIONS

Some of the content presented here also appears in the package vignette of `ggseg`, which may be accessed through R or in the package website (Mowinckel and Piñeiro [2019]). Athanasia Monika Mowinckel also has several tutorials on her blog regarding ggseg creation and functionality (Mowinckel [2018a], Mowinckel [2018b]).

REFERENCES

Inge K. Amlien, Markus H. Sneve, Didac Vidal-Piñeiro, Kristine B. Walhovd, and Anders M. Fjell. Elaboration Benefits Source Memory Encoding Through Centrality Change. *Scientific Reports*, 9(1): 3704, March 2019. ISSN 2045-2322. doi: 10.1038/s41598-019-39999-1. URL `https://doi.org/10.1038/s41598-019-39999-1`.

Winston Chang, Joe Cheng, JJ Allaire, Yihui Xie, and Jonathan McPherson. *shiny: Web Application Framework for R*, 2019. URL `https://CRAN.R-project.org/package=shiny`. R package version 1.3.1.

Anders Dale, Bruce Fischl, and Martin I. Sereno. Cortical surface-based analysis: I. segmentation and surface reconstruction. *NeuroImage*, 9(2):179 – 194, 1999.

Rahul S. Desikan, Florent Ségonne, Bruce Fischl, Brian T. Quinn, Bradford C. Dickerson, Deborah Blacker, Randy L. Buckner, Anders M. Dale, R. Paul Maguire, Bradley T. Hyman, Marilyn S. Albert, and Ronald J. Killiany. An automated labeling system for subdividing the human cerebral cortex on mri scans into gyral based regions of interest. *NeuroImage*, 31(3):968 – 980, 2006. ISSN 1053-8119. doi: https://doi.org/10.1016/j.neuroimage.2006.01.021. URL `http://www.sciencedirect.com/science/article/pii/S1053811906000437`.

Simon B. Eickhoff, B. T. Thomas Yeo, and Sarah Genon. Imaging-based parcellations of the human brain. *Nature Reviews Neuroscience*, 19(11):672–686, November 2018. ISSN 1471-0048. doi: 10.1038/s41583-018-0071-7. URL `https://doi.org/10.1038/s41583-018-0071-7`.

Jupyter et al. *Binder 2.0 - Reproducible, Interactive, Sharable Environments for Science at Scale.*, 2018. Proceedings of the 17th Python in Science Conference.

Bruce Fischl and Anders M. Dale. Measuring the thickness of the human cerebral cortex from magnetic resonance images. *Proceedings of the National Academy of Sciences of the United States of America*, 97(20):11050–11055, 2000.

Bruce Fischl, Martin I. Sereno, and Anders Dale. Cortical surface-based analysis: Ii: Inflation, flattening, and a surface-based coordinate system. *NeuroImage*, 9(2):195 – 207, 1999.

Bruce Fischl, David H. Salat, Evelina Busa, Marilyn Albert, Megan Dieterich, Christian Haselgrove, Andre van der Kouwe, Ron Killiany, David Kennedy, Shuna Klaveness, Albert Montillo, Nikos Makris, Bruce Rosen, and Anders M. Dale. Whole brain segmentation: Automated labeling of neuroanatomical structures in the human brain. *Neuron*, 33(3):341 – 355, 2002. ISSN 0896-6273. doi: https://doi.org/10.1016/S0896-6273(02)00569-X. URL `http://www.sciencedirect.com/science/article/pii/S089662730200569X`.

Aaron Fisher. *ggBrain: ggplot Brain Images*, 2019. R package version 0.1.

Jesper Maag. gganatogram: An r package for modular visualisation of anatograms and tissues based on ggplot2. *f1000research*, 2018. URL `https://f1000research.com/articles/7-1576/v1`. Version 1: Awaiting peer review.

Paul McCarthy. Fsleyes, April 2019. URL `https://doi.org/10.5281/zenodo.2630502`.

A.M. Mowinckel. Get the brain animated! - Dr.Mowinckels blog. `https://drmowinckels.io/blog/get-the-brain-animated/`, 2018a. Accessed: 2019-04-01.

A.M. Mowinckel. Introducing the ggseg r-package for brain segmentations - Dr.Mowinckels blog. `https://drmowinckels.io/blog/introducing-the-ggseg-r-package-for-brain-segmentations/`, 2018b. Accessed: 2019-04-01.

A.M. Mowinckel and D.V. Piñeiro. ggseg package website. `https://lcbc-uio.github.io/ggseg/`, 2019. Accessed: 2019-04-01.

J. Muschelli, A. Gherman, J. P. Fortin, B. Avants, B. Whitcher, J. D. Clayden, B. S. Caffo, and C. M. Crainiceanu. Neuroconductor: an R platform for medical imaging analysis. *Biostatistics*, Jan 2018.

John Muschelli. *ggneuro: Plotting Functions for Neuroimaging Data in 'ggplot2'*, 2017. R package version 0.5.0.

Diego A. Pizzagalli, Avram J. Holmes, Daniel G. Dillon, Elena L. Goetz, Jeffrey L. Birk, Ryan Bogdan, Darin D. Dougherty, Dan V. Iosifescu, Scott L. Rauch, and Maurizio Fava. Reduced caudate and nucleus accumbens response to rewards in unmedicated individuals with major depressive disorder. *American Journal of Psychiatry*, 166(6):702–710, 2009. doi: 10.1176/appi.ajp.2008.08081201. URL `https://doi.org/10.1176/appi.ajp.2008.08081201`. PMID: 19411368.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2019. URL `https://www.R-project.org/`.

Riddihiman. plotly trisurf2. `https://moderndata.plot.ly/trisurf-plots-in-r-using-plotly/`, 2016. Accessed: 2019-04-01.

Carson Sievert. *plotly for R*, 2018. URL `https://plotly-r.com`.

unknown. plotly axes. `https://plot.ly/r/axes/#modifying-axes-for-3D-plots`, unknowna. Accessed: 2019-04-01.

unknown. plotly tri-surf. `https://plot.ly/r/trisurf/`, unknownb. Accessed: 2019-04-01.

D. Vidal-Pineiro, N. Parker, J. Shin, L. French, AP. Jackowski, AM. Mowinckel, Y. Patel, Z. Pausova, G. Salum, Ø. Sørensen, KB Walhovd, T. Paus, and AM Fjell. Cellular correlates of cortical thinning throughout the lifespan. *bioRxiv*, page 585786, January 2019. doi: 10.1101/585786. URL `http://biorxiv.org/content/early/2019/03/23/585786.abstract`.

Kristine B. Walhovd, Anders M. Fjell, Ivar Reinvang, Arvid Lundervold, Anders M. Dale, Dag E. Eilertsen, Brian T. Quinn, David Salat, Nikos Makris, and Bruce Fischl. Effects of age on volumes of cortex, white matter and subcortical structures. *Neurobiology of Aging*, 26(9):1261 – 1270, 2005. ISSN 0197-4580. doi: https://doi.org/10.1016/j.neurobiolaging.2005.05.020. URL `http://www.sciencedirect.com/science/article/pii/S0197458005001673`.

Hadley Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016. ISBN 978-3-319-24277-4. URL `https://ggplot2.tidyverse.org`.