Subcortex visualization: A toolbox for custom data visualization in the subcortex and cerebellum

Annie G. Bryant [1,2,✉]

[1]School of Physics, The University of Sydney, Camperdown, NSW, Australia; [2]Centre for Complex Systems, The University of Sydney, Camperdown, NSW, Australia

Correspondence: ***annie.bryant@sydney.edu.au***

Abstract

Recent years have seen an expanding repertoire of code-based tools for visualizing neuroimaging data, promoting reproducibility and interpretability of brain-mapping findings. However, most open-source visualization packages for the human brain are geared toward the cerebral cortex, and the comparatively few options for the subcortex and cerebellum are limited in scope (i.e., atlas support) and flexibility. We address this critical gap by introducing `subcortex_visualization`, an open-source package offered in both Python and R, that provides a unified and accessible framework for programmatically visualizing non-cortical data across many popular subcortical and cerebellar parcellation atlases. These visuals are inspired by the `ggseg` R package for cortical data, which implements standardized rendering conventions to facilitate comparison across atlases in a vectorized two-dimensional format. In addition to the vectorized versions of nine subcortical and cerebellar atlases—to our knowledge, the most comprehensive collection of non-cortical atlases in a single visualization toolbox—we also provide a step-by-step tutorial for users to generate custom vector-based visualizations from any given brain segmentation, enabling flexible extension to new atlases and structures. Collectively, `subcortex_visualization` and the accompanying documentation support reproducible and interpretable visualization of neuroimaging data below the cortical mantle.

1 Introduction

The effective visualization of data across brain structures is crucial for interpreting and communicating brain-mapping results, from neuroimaging to transcriptomics to cytoarchitecture. Frequently used options include volumetric slices (e.g., from NIfTI files), surface-mesh rendering, or two-dimensional (2D) vector plots. As reviewed in Chopra et al. [1], many open-access tools now programmatically render reproducible brain figures across these formats, bolstering image quality and enabling interactive exploration. However, most tools designed for the human brain target the cerebral cortex and far fewer support subcortical and cerebellar structures, leaving the field without a consistent, accessible way to depict non-cortical data across parcellation atlases.

Our vision for a comprehensive subcortical and cerebellar data visualization toolbox was heavily inspired by the `ggseg` package in R [2] (and extensions in Python[1]), which provides a user-friendly interface for projecting empirical data onto a variety of cortical atlases in two dimensions. While `ggseg` offers subcortical plotting with the FreeSurfer aseg atlas [3], it is not currently possible to show data from all seven subcortical regions (accumbens, amygdala, caudate, hippocampus, pallidum, putamen, thalamus) in a single 2D figure—as no single coronal or sagittal volumetric slice exists that includes all regions simultaneously. Users can project parcellated subcortical data onto three-dimensional (3D) volumetric renderings in `ggseg` [2] as well as other packages such as the ENIGMA toolbox [4] (in Python/MATLAB), though interpretation of the color-mapped data is challenged by occlusion and lighting inhomogeneities [5]. Another key challenge is the limited atlas support beyond that of aseg, despite the increasing availability of alternative atlases for subcortex [6–8], brainstem [9, 10], and cerebellum [11–13].

This tutorial and accompanying code repository addresses these critical gaps in two ways. First, we introduce a straightforward package in both Python (`subcortex_visualization`) and R (`subcortexVisualizationR`) that collectively supports vectorized visualization of empirical data across nine popular parcellation atlases for the subcortex, thalamus, cerebellum. To our knowledge, this is the most comprehensive compilation of subcortical and cerebellar atlases for data visualization in one unified toolbox with standardized rendering conventions. We also demonstrate how to apply the included segmentations to visualize user-supplied empirical data in volumetric format, enabling flexible data plotting across atlases,

[1]https://github.com/ggseg/python-ggseg/

using recently released neurotransmitter receptor maps from Hansen and colleagues [14–16]. Finally, we show how this package can easily integrate with the `ggseg` package [2] to combine cortical, subcortical, and cerebellar visualizations in one figure f or comprehensive brain mapping.

Second, we offer a step-by-step guide for creating custom visualizations from any brain segmentation that can readily interface with functionality of the Python and R packages. This approach is applicable across both cortical and non-cortical structures; however, given the wide variety of existing approaches tailored to the cortex [1], we focus here on the subcortex and cerebellum to demonstrate the potential for visualizing data in these brain systems. Altogether, in providing both ready-to-use atlases and instructions with which users can generate vectorized images for new atlases, it is our hope that this provides a crucial resource for data visualization options below the cortical mantle.

2 Introducing the subcortex_visualization package

In the first half of this tutorial, we introduce the '`subcortex_visualization`' package, demonstrating its functionality for code-based data visualization using nine pre-defined subcortical and cerebellar atlases. We first discuss installation and then review the nine atlases that are pre-loaded as vectorized images in the package before introducing simple code snippets that can be used to visualize empirical data in a given parcellation atlas. Additional documentation for the toolbox is provided on the website hosted by the GitHub repository[2]. Finally, we demonstrate how to easily extract and plot statistics using provided atlases, demonstrated with neurotransmitter receptor maps curated by Hansen and colleagues [14–16] as an example.

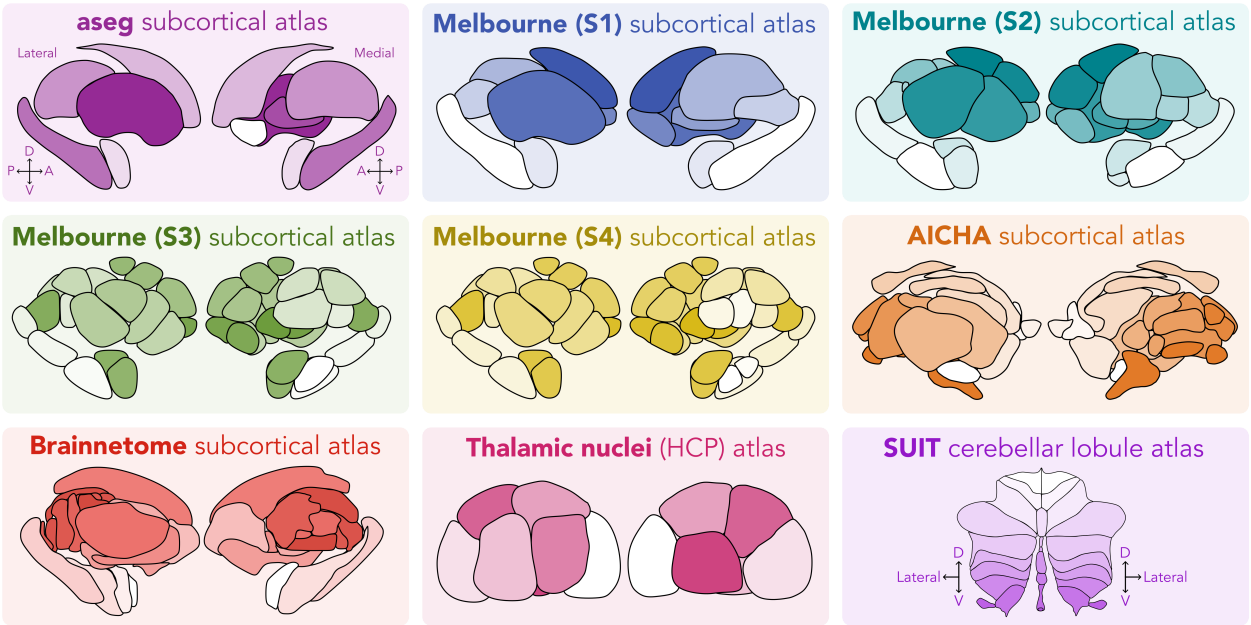

Figure 1. Nine atlases are included in this package for ready-to-go visualization. The nine atlases included with this package are depicted for reference. These include the aseg subcortical atlas [3]; Melbourne subcortical atlases, from resolutions S1 through S4 [8]; AICHA subcortical atlas [6]; Brainnetome subcortical atlas [7]; thalamic nuclei atlas [17]; and the SUIT cerebellar lobule atlas [11, 18]. Atlases are shown for the right hemisphere, with orientation markers indicated in the aseg subcortical atlas panel (top left) and, separately, the SUIT cerebellar lobule atlas (bottom right). D: dorsal; A: anterior; V: ventral; P: posterior.

[2]https://anniegbryant.github.io/subcortex_visualization

2.1 Installation and troubleshooting

Both the Python and R versions of our package are fully open-source, and users are encouraged to share any issues or questions via the GitHub repository and contribute via pull requests.

The Python version of our package (`subcortex_visualization`) can be installed via the PyPI repository [19] or cloned from GitHub [20], with more detailed installation instructions provided in the package website[3]. The most straightforward installation via PyPI can be accomplished as below:

```
pip install subcortex-visualization
```

The R version of our package (`subcortexVisualizationR`) can be installed from the GitHub repository using the `remotes` package, as shown below:

```
# if not already installed
install.packages("remotes")
# then install subcortexVisualizationR
remotes::install_github(
"anniegbryant/subcortex_visualization",
subdir = "subcortexVisualizationR")
```

2.2 Viewing the atlases provided with the package

We have incorporated nine widely used subcortical and cerebellar atlases as a starting point for data visualization—each with its own unique parcellation scheme and resolution (i.e., number of regions). These atlases are summarized in Table 1 and visualized in Fig. 1. Briefly, the nine atlases for which we provide pre-vectorized images in this toolbox include the aseg subcortical atlas [3]; Melbourne Subcortex Atlas, from resolutions S1 through S4 [8]; AICHA subcortical atlas [6]; Brainnetome subcortical atlas [7]; thalamic nuclei atlas [17]; and surface-based probabilistic cerebellar lobule atlas [21], adapted from the volumetric SUIT lobule atlas [11, 18].

The central workhorse function of this package is `plot_subcortical_data`, which takes a variety of parameters designed to maximize flexibility in data visualization. Users can supply their own empirical data in the form of a `pandas DataFrame` [22] (with region names exactly matching those defined in the atlas) to be color-mapped onto the regions of interest. Users can also specify a given atlas (any of the pre-constructed nine will work, along with any custom atlas that a user may incorporate), line color and thickness, and color map (for example, 'viridis' or 'turbo', or a custom Matplotlib [23] color map). To maintain consistency across other visualizations, users can also set a fixed upper bound, lower bound, and/or midpoint for the colormap values. By default, the function will plot the index value for each region for the aseg atlas in the left hemisphere with black outlines and the *viridis* colormap. An example of using this function with default parameters is included below in Code snippet 1 (n.b., all snippets are included in Appendix I), with the resulting output depicted in Fig. 2.

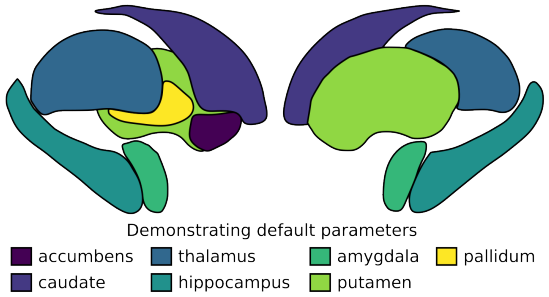

Figure 2. Example plot constructed using the default parameters of `plot_subcortical_data`.

Table 1. The nine subcortical and cerebellar parcellation atlases provided in `subcortex_visualization` are summarized.

Atlas name	# regions per hemisphere	Source publication	More detail shown in
aseg	7	Fischl et al. [3]	Fig. S1
Melbourne (S1)	8	Tian et al. [8]	Fig. S2
Melbourne (S2)	16	Tian et al. [8]	Fig. S3
Melbourne (S3)	25	Tian et al. [8]	Fig. S4
Melbourne (S4)	27	Tian et al. [8]	Fig. S5
AICHA	20	Joliot et al. [6]	Fig. S6
Brainnetome	18	Fan et al. [7]	Fig. S7
Thalamus (HCP)	7	Najdenovska et al. [17]	Fig. S8
SUIT	17	Diedrichsen and Zotow [21]	Fig. S9

[3] https://anniegbryant.github.io/subcortex_visualization/installation

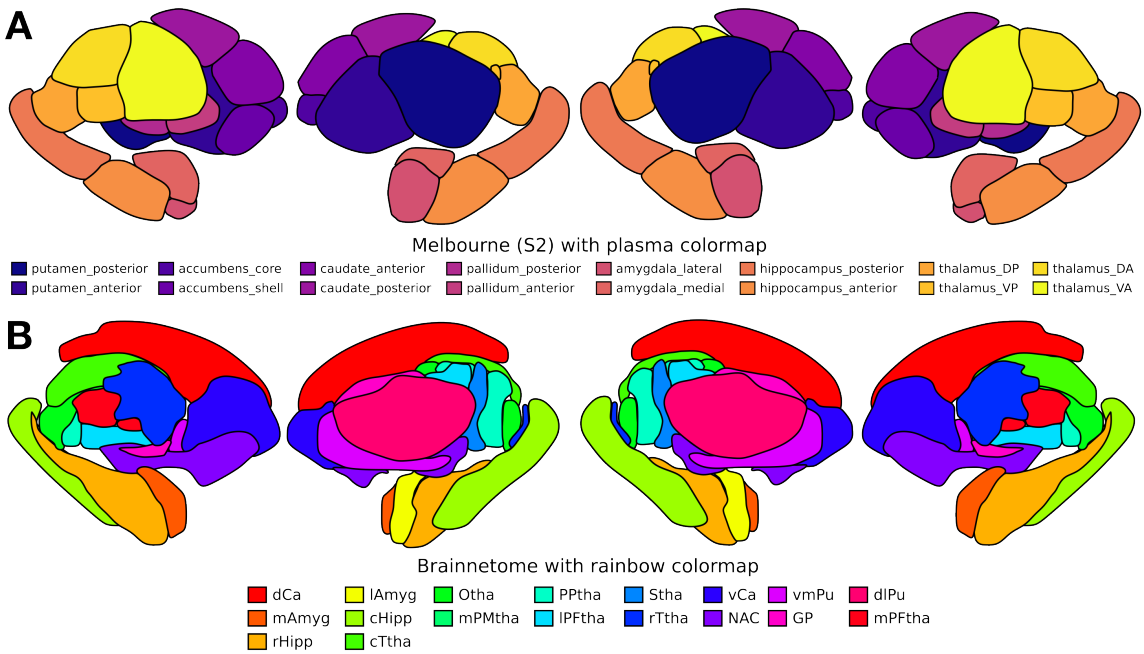

Figure 3. Example plot demonstrating the Melbourne (S2) and Brainnetome subcortical atlas in both hemispheres using different colormaps. A. Melbourne (S2) subcortical atlas with plasma colormap. **B.** Brainnetome subcortical atlas with rainbow colormap.

A similar figure can be easily generated for any of the other atlases by specifying the `atlas` keyword argument, as demonstrated in Code snippet 2 with the Melbourne Subcortex Atlas (S2 resolution; plasma colormap) and Brainnetome subcortical atlas (rainbow colormap) in both hemispheres. The resulting brain maps are depicted in Fig. 3.

2.3 Working with your own data

We now turn to functionality for plotting empirical data in one or both hemispheres using the pre-vectorized atlases provided with this package. Note that the region names used in the `DataFrame` must match those defined in the atlas exactly (including capitalization and spacing). The region names for a given atlas can be shown directly with the following code, using the Melbourne Subcortex Atlas with S1 resolution as an example (same syntax in R and Python):

```
get_atlas_regions('Melbourne_S1')
```

The region names for each atlas can also be found on the package website[4] and are provided in tabular format for each atlas in the Supplementary Material (Table S1).

[4]https://anniegbryant.github.io/subcortex_visualization/atlas_info

To demonstrate this functionality, we simulate random continuous data to plot across the regions of the aseg atlas in Code snippet 3. This simulated data can be directly mapped onto the aseg atlas using the `plot_subcortical_data` function, as shown below and depicted in Fig. 4—with the corresponding code included in Code snippet 4.

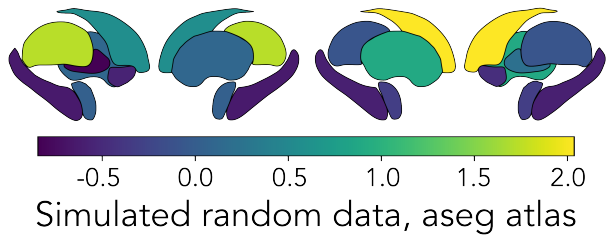

Figure 4. Simulated continuous data plotted in the *aseg* atlas in both hemispheres.

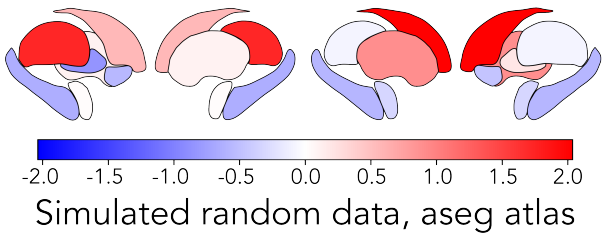

Figure 5. Simulated continuous data plotted in the *aseg* atlas in both hemispheres, with a custom diverging color palette.

One can also pass in a custom colormap to the `cmap` argument, as demonstrated in Code snippet 5—using a blue-white-red colormap to highlight both positive and negative values in the data (Fig. 5). `plot_subcortical_data` also takes an argument, `midpoint`, which specifies the center point for the color palette, as demonstrated in Code snippet 5. Setting this to 0 here enforces the center value to be white. Without explicitly setting the upper and lower bounds (`vmax` and `vmin`, respectively), the color range will be defined symmetrically around `midpoint` to capture the full range of the data.

2.4 Extracting regional statistics from volumetric data

In addition to visualizing empirical data, the `subcortex_visualization` package also provides functionality for extracting summary statistics from volumetric data in any of the nine included atlases. This is accomplished with the `apply_atlas_to_data` function, which takes as input a volumetric dataset (in NIfTI format) and an atlas name (any of the nine included atlases) and applies the `Parcellator` object

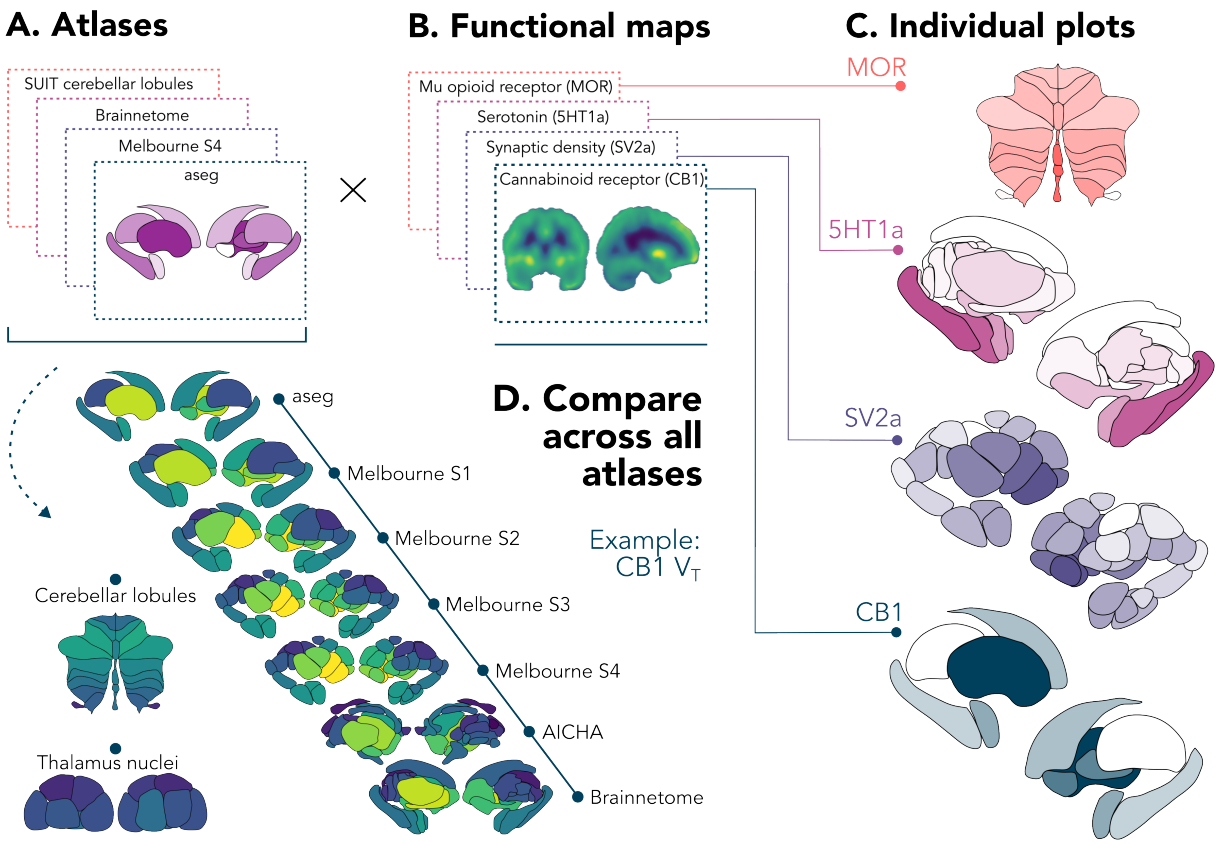

Figure 6. Overview of package functionality for computing and visualizing parcellated statistics from a new map of interest. A. Users can select one or more of the atlases included in this package to parcellate their input volumetric data. **B.** Any 3D NIfTI image can be used as input data, so long as it is registered to MNI152NLin6Asym space. Here, we show four example PET maps of receptor or transporter density from Hansen and colleagues [14–16]: Mu opioid receptor (MOR) [24–26], serotonin (5HT1a) [27], pre-synaptic terminal density (SV2a) [28], and cannabinoid receptor (CB1) [29]. **C.** Results can be plotted individually for each given functional map × atlas combination, with examples here including the MOR map parcellated with the SUIT cerebellar lobule atlas, 5HT1a parcellated with the Brainnetome atlas, SV2a parcellated with the Melbourne Subcortex Atlas (S4 resolution), and CB1 parcellated with the aseg atlas. **D.** For meta-analytic purposes, users can apply multiple parcellation atlases to the same functional map and visualize the results together, as shown here for the CB1 PET absolute tracer volume (V_T) map parcellated with all atlases included in this package.

class from the `neuromaps` package to compute region-averaged statistics [15], analogous to the `fslmeants` function from FSL [30]; we implement the default functionality for the `Parcellator` object to include zero-valued voxels when computing each parcel average. Of note, the aseg, Melbourne Subcortex, AICHA, and SUIT cerebellar atlases were originally shared in MNI152NLin6Asym space at 1mm isotopic resolution, while the Brainnetome and thalamic nuclei atlases were originally shared in MNI152NLin2009cSym space. To unify the space of all atlases, we applied pre-computed transformations from TemplateFlow [31] to resample the Brainnetome and thalamic nuclei atlases to MNI152NLin6Asym space using the ANTs toolbox [32]. We provide the transformed atlases (in MNI152NLin6Asym space) in the `subcortex_visualization` repository for ease of use; an example of the code used to perform atlas alignment is given in Code snippet 6. As a result, users should ensure that their volumetric data is also in MNI152NLin6Asym space prior to calling the `apply_atlas_to_data` function; for those wanting to learn more about these spaces and volumetric mapping, we refer the interested reader to Evans et al. [33].

If a user has their own volumetric data in NIfTI format that they wish to analyze, they can select one or more of the atlases provided in this package (Fig. 6A) to extract regional statistics with the `apply_atlas_to_data` function. As an example, we demonstrate how to use this function to extract regional statistics from group-averaged positron emission tomography (PET) neurotransmitter receptor/transporter maps curated by Hansen and colleagues [14–16]. These maps are openly available from GitHub (https://github.com/netneurolab/hansen_receptorvar) as well as the `neuromaps` Python package [15]. There are currently twenty distinct group-averaged PET maps in this dataset, spanning seven distinct neurotransmitter systems and pre-synaptic terminal density. Some example maps are shown in Fig. 6B, including ones for the Mu opioid receptor (MOR) [24–26], 5HT1a serotonergic receptor [27], SV2a synaptic vesicle protein [28], and CB1 cannabinoid receptor [29]. All volumes were obtained in MNI152NLin6Asym space at 1mm isotropic resolution (from the MNI152 ICBM atlas [34]).

We include a worked example in Code snippet 7, showing how to compute the region-averaged CB1 receptor density across the aseg atlas. This offers flexibility in choosing the most appropriate parcellation scheme for statistical analysis and data visualization in one unified framework. For example, as depicted in Fig. 6C, a researcher could visualize the mean MOR tracer binding in the SUIT cerebellar lobule atlas; the mean 5HT1a receptor density in the Brainnetome subcortical atlas; the mean SV2a synaptic density in the Melbourne Subcortex Atlas at S4 resolution; or the mean CB1 receptor density in the aseg atlas. Of note, many receptor maps are quantified using either the standardized uptake value (SUV) or binding potential (BP_{ND}), which generally involve normalizing the raw PET signal to a reference region that is typically the cerebellum (either whole cerebellum or cerebellar gray matter) [16]—with the exception of the MOR tracer, which was normalized to the occipital cortex. This is an important consideration when selecting which atlas to use for visualization of a given PET volume, since resulting values in the cerebellar lobules are more likely to reflect noise rather than true receptor density.

Users can also supply multiple atlases simultaneously to the `apply_atlas_to_data` function, such that the output `DataFrame` contains regional statistics across all specified atlases. This is particularly useful for comparing empirical results across multiple parcellation schemes, as demonstrated in Fig. 6D, which shows the mean CB1 receptor binding across all nine atlases provided in this package. We include a worked example for this functionality in Code snippet 8. As long as the functional volume is in MNI152NLin6Asym space, users can flexibly choose any combination of atlases to extract and visualize region-averaged results.

2.5 Combining visualizations across cortical and non-cortical structures

Since our visualizations were designed in the style of the `ggseg` cortical plots in R [2], we finally demonstrate how to combine cortical and non-cortical visualizations in one figure for comprehensive brain mapping. Importantly, the R functionality with `subcortexVisualizationR` allows the user to easily combine cortical plots from `ggseg` with subcortical and cerebellar plots from `subcortexVisualizationR` in one figure, all within the `ggplot2` [36] framework. As an example, we visualize the mean $GABA_A$-$\alpha 1$ receptor subunit density [37] (from the same PET data repository as in our previous empirical data example, Hansen et al. [16]) across the cortex, subcortex, and cerebellum together. The $GABA_A$-$\alpha 1$ receptor subunit was measured using the [^{11}C]Ro15-4513 PET tracer and quantified as the total volume of tracer distribution (V_T), derived using kinetic modeling with an arterial input function [37]—and is not normalized to a reference region, enabling visualization in the cerebellar lobules. We selected the 1000-parcel resolution from Schaefer et al. [35] to parcellate the cerebral cortex, which we applied to the PET data mapped to *fsaverage* surface space using the `neuromaps` package [15]. For the subcortex, we selected the Melbourne Subcortex Atlas [8] at the S4 resolution to match the visualizations in Hansen et al. [16] and for the cerebellum, we used the SUIT cerebellar lobule atlas [21].

In this workflow (included in Code snippets 9 and 10), we construct a separate `ggplot2` plot for the cortex (using `ggseg` [2]) and for the subcortex and cerebellum (using `subcortexVisualizationR`), which we then combine into one figure using the `patchwork` package [38]. The resulting composite $GABA_A$-$\alpha 1$ subunit density figure is shown in Fig. 7. Note that the color range is consistent across all three plots, allowing for direct comparison of receptor subunit density across the cortical and non-cortical structures. This presents a clear and interpretable way to visualize brain-wide data in one unified figure with consistent rendering conventions, eliminating the need to manually combine cortical surface plots with volumetric subcortical slices or 3D mesh renderings.

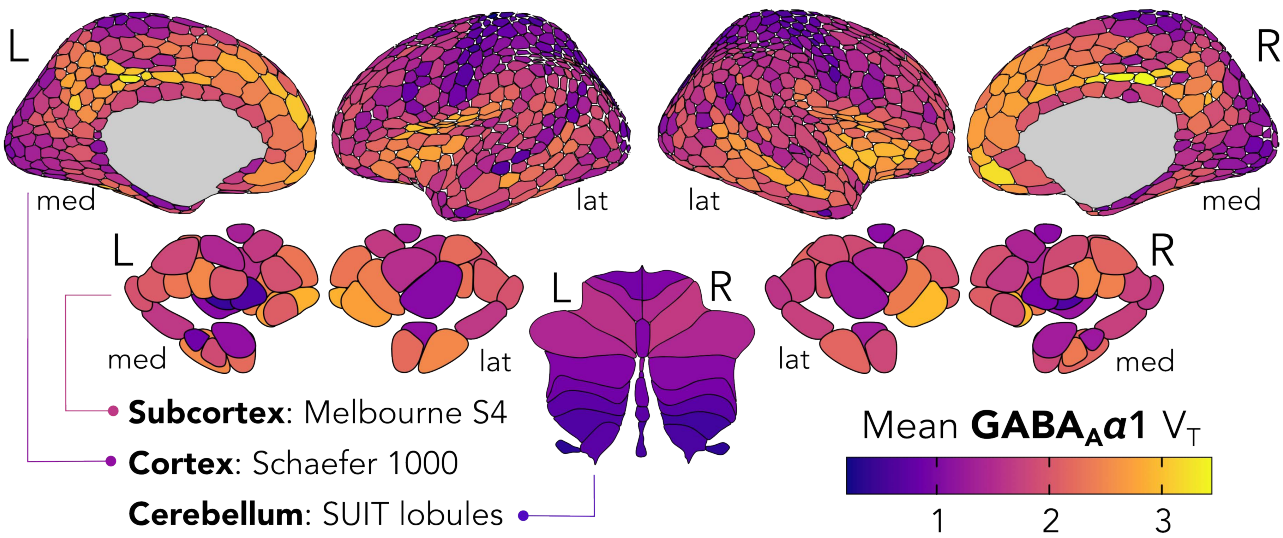

Figure 7. With subcortexVisualizationR, users can plot data across the cortex, subcortex, and cerebellum all together. Here, we plot region-averaged receptor density maps for $GABA_A\alpha 1$, parcellated in the cortex (Schaefer-1000 atlas [35]), subcortex (S4 resolution from the Melbourne Subcortex Atlas [8]), and cerebellum (SUIT lobule atlas [18, 21]). The color bar indicates total volume of tracer distribution (V_T) values that map to cortex, subcortex, and cerebellum.

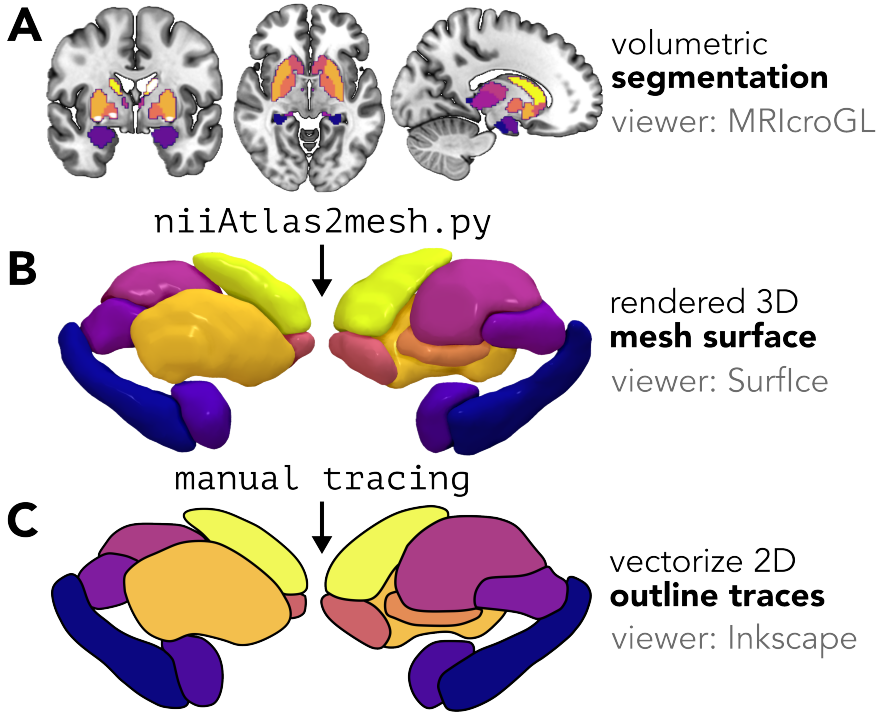

Figure 8. Pipeline from volumetric segmentation through to vectorized two-dimensional image. A. Most non-cortical atlases originate in the form of a volumetric segmentation, typically in the form of a NIfTI image. Such a volume can be viewed as an overlay atop three-dimensional structural images using software like MRIcroGL [39]. **B.** The segmentation volume is converted to a 3D mesh surface using the `niiAtlas2mesh.py` script, which yields a mesh representation of the atlas in the form of an .mz3 file. This mesh file is then loaded into Surfice [40] for interactive visualization and customization of the surface, from which still images of the medial and lateral surfaces are captured. **C.** These captured images are loaded using the vector graphic software Inkscape, where the outlines of the structures are manually traced to yield a contiguous shape for each region.

3 Creating custom two-dimensional brain visualizations

The nine subcortical and cerebellar atlases included in this package represent an extensible starting point for visualizing data in the subcortex and cerebellum. We now walk through the process of constructing 2D vector visualizations for any given brain parcellation atlas, which could also include cortical structures as well as the brainstem [9, 10]. The overall workflow is depicted schematically in Fig. 8, using the Melbourne Subcortex Atlas [8] with the S1 resolution as an example, comprising three key steps: going from the volumetric segmentation (Fig. 8A) to rendered 3D mesh (Fig. 8B) to manually-traced 2D vector graphic (Fig. 8C). Each conversion and visualization step is implemented with free open-source tools to promote accessibility and reproducibility, which we outline in detail below.

3.1 Rendering a three-dimensional surface mesh

Given that segmentations below the cortical mantle are generally volume-based, rather than surface-based, most atlases are stored in a 3D volumetric format (e.g., NIfTI). Fig. 8A shows the Melbourne Subcortex Atlas (S1 resolution) segmentation overlaid in color atop the MNI152 brain template using the MRIcroGL visualization software [39]. To view just these subcortical structures in three dimensions, we rendered the atlas regions as a set of mesh surfaces forming one cohesive 3D object, as depicted in Fig. 8B.

Here, we implement a Python-based solution developed by Professor Chris Rorden at the University of South Carolina, which uses an enhanced marching cubes implementation [41]: niiAtlas2mesh.py (which can be found among other useful Python-

based neuroimaging scripts at https://github.com/rordenlab/pythonScripts). This script utilizes the niimath library [42]—developed by the same group—which incorporates command-line tool functionality from FSL [30] into Python. Specifically, niiAtlas2mesh.py calls niimath (which emulates fslmaths) with a series of steps that binarize and smooth each region in the segmentation volume before extracting a mesh surface with an enhanced marching cubes algorithm [41, 43], and finally (slightly) inflating and smoothing the mesh to mitigate small irregularities such as holes or sharp edges. For more discussion about the details of this workflow, we refer the interested reader to Rorden et al. [44]. Together with combinemz3.py, these two scripts first generate individual mesh surfaces for each region in the segmentation volume and then combine them into one cohesive mesh file in .mz3 format. While .mz3 mesh files can be opened in various neuroimaging applications, it is primarily designed for Surfice [40], which we use here; more information about this file format can be found at https://github.com/neurolabusc/surf-ice/tree/master/mz3. As shown in Fig. 9, the resulting mesh file (Subcortex_S1_3T_1mm.mz3, in this example) is opened in Surfice, which renders the entire mesh object with color coding for each region. We recommend rotating the mesh to capture screenshots of the medial and lateral views for tracing, as discussed in the next section.

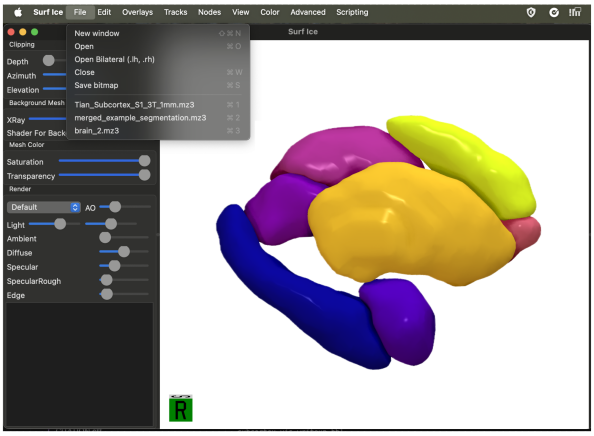

Figure 9. Screenshot from the Surfice visualization software. This screen capture depicts the Surfice software rendering the generated .mz3 mesh surface file for the Melbourne Subcortex Atlas at S1 resolution (right hemisphere only) using the plasma color map. Users can export a still image of their desired angles (we recommend lateral, as depicted, and medial) using the 'File > Save bitmap' action and exporting as a .png image.

The Python scripts are adapted in our workflow to allow the user to specify a custom color map that assigns an RGBA value to each vertex, corresponding to one color per unique region index. One can construct a new color map with the desired number of colors (e.g., the number of regions to be visualized in the mesh) in Python as demonstrated in Code snippet 11, which selects eight evenly spaced shades across the 'plasma' colormap from Matplotlib [23] as an example. Note that any colormap can be used (including a custom-generated one), and the number of discrete colors to sample should be set to match the number of subregions in the given atlas and number of hemispheres—for example, for the left subcortex in the Melbourne Subcortex Atlas (S2 resolution), 16 colors should be sampled. The script generate_color_map.py yields a .txt file with RGBA-formatted colors that can then be read with the -colors command-line input with the volume_to_mesh.py script, along with the segmentation volume and desired output filename, as shown in Code snippet 12. Generally speaking, we recommend separating out the two hemispheres for maximum visibility along the medial face (i.e., so medial areas are not obstructed by the opposite hemisphere). If there are substantial structural differences between the left and right hemispheres, this process should be repeated separately for each hemisphere; otherwise, the traced regions can simply be duplicated and then mirrored along the horizontal axis, though one should still use the appropriate region name suffixes as discussed below. The example walked through in this guide works with the right hemisphere of the Melbourne Subcortex Atlas at S1 resolution.

We note that alternative graphical user interface (GUI)-based and code-based options are also available to convert NIfTI files into 3D surface meshes. Madan [45] provides a comprehensive tutorial for constructing 3D surface meshes from a volumetric atlas using a combination of the ITK-SNAP [46] and ParaView [47] programs, which we recommend for users who prefer a GUI-based approach. There are also excellent resources for web-based mesh surface rendering through the NiiVue package (https://niivue.com/demos/features/meshes.html); for example, the SUIT probabilistic cerebellar lobule atlas is rendered as a mesh with interactive viewing options at https://niivue.com/demos/features/mesh.atlas.suit.html. Alternatively, other programmatic approaches exist for constructing these triangulated meshes from voxel-based volumes (including a Python-based version of

NiiVue with Jupyter notebooks, at https://github.com/niivue/ipyniivue). Any of these approaches can be used to generate the 3D mesh surface as an intermediate step toward creating the final 2D vector graphic.

3.2 Tracing the outline of each region

The next step is to trace the outline of each region in the segmentation mesh to create a vector-based graphic. While there are edge detection tools that can semi-automate this process (such as the binary silhouette mesh shader in the NiiVue package; https://github.com/niivue/niivue), we opt to manually trace each region to ensure consistent quality across atlases. We use the freely available `Inkscape` software for this process (https://inkscape.org/), which allows users to import bitmap images (e.g., screenshots of the mesh renderings from Surfice) and trace around each region to create vectorized polygons—though other vector graphic editing software can also be used for this purpose, such as Adobe Illustrator or Blender. One should create a traced object for each region (e.g., putamen) and face (e.g., lateral and medial) separately, as discussed below. The final traced images can be stacked along the z-axis and combined with fill colors match-ing those in the original mesh rendering as shown in Fig. 8C.

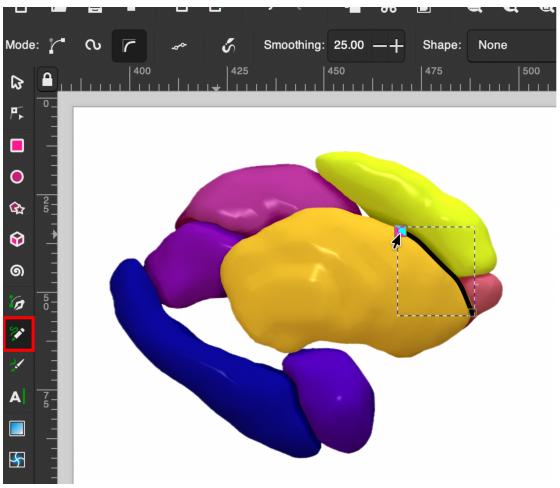

Figure 10. Example of manual tracing of structure outlines using Inkscape. Here, the 'Draw freehand lines' tool is selected (along the toolbar on the left, highlighted in red) and the outline of the right putamen is being traced.

After opening the bitmap image (named, e.g., 'My_Atlas_lateral.png') in Inkscape, one should select the 'Draw freehand lines' option from the toolbar on the left-hand side of the window (denoted with a pencil and drawing icon), as shown in a screenshot in Fig. 10. In this example, we start with the right putamen by simply tracing loosely around the border of this region. The line can be enclosed into a polygon encapsulating this region by connecting the line back to the original trace starting point. This process should be repeated to construct one traced polygon for each region visible in the corresponding view angle (i.e., not all regions will be visible in both the medial and lateral planes).

Once all regions have been traced for the corresponding segmentation and view angle(s), the final step in Inkscape is to assign a title to each region in the Object properties panel. This step enables the `subcortex_visualization` toolbox to programmatically access each region and map them to user-supplied data for flexible downstream data visualization, using the `svgpath2mpl` package (https://github.com/nvictus/svgpath2mpl). As depicted schematically in Fig. 11, one should select each traced region one by one, and in the 'Object properties' panel on the righthand side (denoted with a wrench icon), assign an appropriate title to the region. Both our Python package (`subcortex_visualization`) and R package (`subcortexVisualizationR`) rely on a consistent naming convention to identify each region, face, and hemisphere, as color-coded in Fig. 11: `[region]_[face]_[hemisphere]`, with the example of `putamen_lateral_R` shown here. The `face` can be either 'medial' or 'lateral' and the `hemisphere` can be either 'right' or 'left' (or 'vermis', for the case of the surface-based cerebellar atlas [21]).

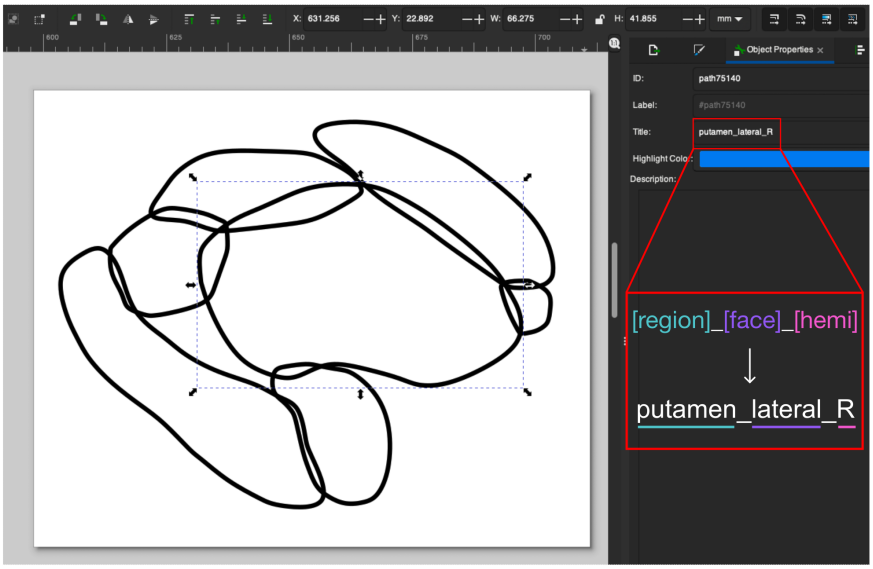

Figure 11. Naming convention for traced objects in Inkscape. Each traced structure is given a name in the Inkscape 'Object Properties' panel (shown on the right side of the screenshot). The naming convention used here is [region]_[face]_[hemisphere], where [region] is the name of the brain region in the corresponding atlas (e.g., 'putamen'), [face] indicates whether the traced object corresponds to the 'lateral' or face of the structure, and [hemisphere] indicates the hemisphere ('L' for left, 'R' for right).

3.3 Incorporating custom visualizations into the subcortex_visualization package

To align with the file structure of the `subcortex_visualization` toolbox (both Python and R versions), users should save one set of labeled region tracings each for the left hemisphere, right hemisphere, and both hemispheres together. These files should be organized into the `subcortex_visualization/data/` folder with the following general format, using an atlas named 'My_Atlas' as an example:

```
# Python package file structure
subcortex_visualization/
  data/
    - My_Atlas_both.svg
    - My_Atlas_L.svg
    - My_Atlas_R.svg

# R package file structure
subcortexVisualizationR/
  inst/
    extdata/
      - My_Atlas_both.svg
      - My_Atlas_L.svg
      - My_Atlas_R.svg
```

For the traced regions to be properly superimposed along the z-axis in the code-based rendering, the region layering order should be explicitly defined in a tabular data file for each hemisphere (including one .CSV file for the two hemispheres together). The first few rows of an appropriate region ordering table are shown in Table 2, which includes the name of the region, the face, the order in which that region should be drawn (with 1 being lowest in the z-axis; in other words, placed at the bottom of the stack), the hemisphere, and the index mapping to that region/hemisphere in the original segmentation volume. Constructing this table often requires some trial and error to refine the correct ordering for z-stacking each region in layers.

Together with the .svg images containing the vectors themselves, the file structure including these layer ordering tabular files should now look like:

```
# Python package file structure
subcortex_visualization/
  data/
    - My_Atlas_both_ordering.csv
    - My_Atlas_both.svg
    - My_Atlas_L_ordering.csv
    - My_Atlas_L.svg
    - My_Atlas_R_ordering.csv
    - My_Atlas_R.svg
```

```
# R package file structure
subcortexVisualizationR/
  inst/
    extdata/
      - My_Atlas_both_ordering.csv
      - My_Atlas_both.svg
      - My_Atlas_L_ordering.csv
      - My_Atlas_L.svg
      - My_Atlas_R_ordering.csv
      - My_Atlas_R.svg
```

In order to integrate the newly vectorized 'My_Atlas' with the existing `subcortex_visualization` toolbox, one simply needs to reinstall the toolbox from the cloned repository where the new atlas has been added. For example, if this was performed in a parent directory called 'code', the commands to re-install the Python package are:

```
cd code/subcortex_visualization
pip install .
```

Similarly, for the R package, one would run the following commands from the parent directory (in R):

```
setwd("code/subcortexVisualizationR")
remotes::install_local(".")
```

Now, all data visualization functionality introduction in Sec. 2.3 can be implemented with atlas 'My_Atlas'. Users are also encouraged to share any added custom atlas vector graphics to the project GitHub repository (https://github.com/anniegbryant/subcortex_visualization) through a pull request, which will allow other researchers to benefit from the expanded library of subcortical and cerebellar atlases.

Table 2. Format of the user-generated region layer ordering in the SVG file.

region	face	plot_order	Hemisphere	seg_index
putamen	medial	1	R	7
pallidum	medial	2	R	5
accumbens	medial	3	R	6
caudate	medial	4	R	8
thalamus_anterior	medial	5	R	4

4 Discussion

In this work, we introduce the `subcortex_visualization` toolbox (in both Python and R), with the aim of broadening the availability of non-cortical data visualization tools for the human brain in a streamlined and accessible manner. This presents a unified subcortical and cerebellar visualization tool that complements existing software for 2D cortical rendering (e.g., `ggseg` [2], along with `pycortex` [48] for cortical flatmap projections) and projection of empirical data onto a 3D surface for cortical and non-cortical structures alike (e.g., ggseg [2], ENIGMA toolbox [4], fsbrain [49], and brainrender [50]). We curated a set of nine existing atlases that are frequently used in human brain-mapping studies, including seven for the broader subcortex, one for thalamic nuclei, and one for the cerebellum. Additionally, we developed an integrated tutorial pipeline for creating new vector-based visualizations for segmentations beyond those included here, paving the way for flexible incorporation of new atlases across both cortical and non-cortical structures.

To our knowledge, this is the most comprehensive library for subcortical and cerebellar atlases in one data visualization toolbox. The limited availability of code-based data visualization tools for the subcortex and cerebellum reflects a historical 'cortico-centric' bias in cognitive and computational neuroscience [51], arising in part from physical difficulties in obtaining reliable signal from more deeply embedded structures relative to superficial cortical areas [52–54]. However, in an era of increasing focus on the subcortex and cerebellum in health and disease, imaging resolution continues to improve, such that more studies can start to depict analytical results from the subcortex and cerebellum alongside those of the cortex [8, 10, 55]. This package thus aims to facilitate accessible and interpretable data visualization across non-cortical parcellation atlases, which is particularly timely with the recent release of multiple receptor density PET maps across cortex, subcortex, and cerebellum by Hansen et al. [16].

We hope that this serves as an important step towards increasing the availability of open-source programmatic data visualization tools for non-cortical structures, which can be readily integrated with other R- and Python-based imaging analysis workflows. Recognizing the importance of open science and community engagement, we have made this package available on GitHub at https://github.com/anniegbryant/subcortex_visualization. Additionally, documentation (including installation instructions and tutorial Jupyter notebooks) can be found at the project website at https://anniegbryant.github.io/subcortex_visualization/. As an open-source tool, we warmly welcome user contributions for new atlases to include in the library for the benefit of other researchers. For example, future work might also incorporate brainstem atlases [9, 10] and other thalamic nuclei parcellations [56–58]. Altogether, by providing both out-of-the-box atlases and instructions with which users can generate such atlases, this toolbox represents a crucial resource for data visualization options below the cortical mantle.

5 Acknowledgments

A.G.B. was supported by an Australian Government Research Training Program (RTP) Scholarship and the Paulette Isabel Jones Career Award, both granted by The University of Sydney. We also wish to thank Sidhant Chopra, Chris Rorden, Justine Hansen, and Ye Tian for their suggestions and continued development of open tools for neuroimaging visualization that enabled the development of this project.

References

[1] Sidhant Chopra et al. A practical guide for generating reproducible and programmatic neuroimaging visualizations. *Aperture Neuro* 3 (2023), pp. 1–20.

[2] Athanasia M Mowinckel and Didac Vidal-Piñeiro. Visualization of brain statistics with R packages ggseg and ggseg3d. *Advances in Methods and Practices in Psychological Science* 3.4 (2020), pp. 466–483.

[3] Bruce Fischl et al. Whole brain segmentation: automated labeling of neuroanatomical structures in the human brain. *Neuron* 33.3 (2002), pp. 341–355.

[4] Sara Larivière et al. The ENIGMA Toolbox: multiscale neural contextualization of multisite neuroimaging datasets. *Nature methods* 18.7 (2021), pp. 698–700.

[5] Danielle Albers Szafir et al. Lightness constancy in surface visualization. *IEEE transactions on visualization and computer graphics* 22.9 (2015), pp. 2107–2121.

[6] Marc Joliot et al. AICHA: An atlas of intrinsic connectivity of homotopic areas. *Journal of neuroscience methods* 254 (2015), pp. 46–59.

[7] Lingzhong Fan et al. The human Brainnetome atlas: A new brain atlas based on connectional architecture. *Cerebral cortex* 26.8 (2016), pp. 3508–3526.

[8] Ye Tian et al. Topographic organization of the human subcortex unveiled with functional connectivity gradients. *Nature neuroscience* 23.11 (2020), pp. 1421–1432.

[9] Marta Bianciardi et al. Toward an in vivo neuroimaging template of human brainstem nuclei of the ascending arousal, autonomic, and motor systems. *Brain connectivity* 5.10 (2015), pp. 597–607.

[10] Justine Y Hansen et al. Integrating brainstem and cortical functional architectures. *Nature Neuroscience* 27.12 (2024), pp. 2500–2511.

[11] Jörn Diedrichsen. A spatially unbiased atlas template of the human cerebellum. *Neuroimage* 33.1 (2006), pp. 127–138.

[12] Min Tae M Park et al. Derivation of high-resolution MRI atlases of the human cerebellum at 3 T and segmentation using multiple automatically generated templates. *Neuroimage* 95 (2014), pp. 217–231.

[13] Jose E Romero et al. CERES: a new cerebellum lobule segmentation method. *Neuroimage* 147 (2017), pp. 916–924.

[14] Justine Y Hansen et al. Mapping neurotransmitter systems to the structural and functional organization of the human neocortex. *Nature neuroscience* 25.11 (2022), pp. 1569–1581.

[15] Ross D Markello et al. Neuromaps: structural and functional interpretation of brain maps. *Nature Methods* 19.11 (2022), pp. 1472–1479.

[16] Justine Y Hansen et al. Inter-individual variability of neurotransmitter receptor and transporter density in the human brain. *bioRxiv* (2025), pp. 2025–09.

[17] Elena Najdenovska et al. In-vivo probabilistic atlas of human thalamic nuclei based on diffusion-weighted magnetic resonance imaging. *Scientific data* 5.1 (2018), pp. 1–11.

[18] Jörn Diedrichsen et al. Imaging the deep cerebellar nuclei: a probabilistic atlas and normalization procedure. *Neuroimage* 54.3 (2011), pp. 1786–1794.

[19] Annie G. Bryant. *subcortex-visualization*. Version 1.2.0. 2026. URL: https://pypi.org/project/subcortex-visualization/.

[20] Annie G. Bryant. *anniegbryant/subcortex-visualization*. Version 1.2.0. 2026. DOI: 10.5281/zenodo.15385315. URL: https://doi.org/10.5281/zenodo.15385316.

[21] Jörn Diedrichsen and Ewa Zotow. Surface-based display of volume-averaged cerebellar imaging data. *PloS one* 10.7 (2015), e0133402.

[22] The pandas development team. *pandas-dev/pandas: Pandas*. Version latest. Feb. 2020. DOI: 10.5281/zenodo.3509134. URL: https://doi.org/10.5281/zenodo.3509134.

[23] J. D. Hunter. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering* 9.3 (2007), pp. 90–95. DOI: 10.1109/MCSE.2007.55.

[24] Jarkko Johansson et al. Intranasal naloxone rapidly occupies brain mu-opioid receptors in human subjects. *Neuropsychopharmacology* 44.9 (2019), pp. 1667–1673.

[25] S Lamusuo et al. Neurotransmitters behind pain relief with transcranial magnetic stimulation–positron emission tomography evidence for release of endogenous opioids. *European Journal of Pain* 21.9 (2017), pp. 1505–1515.

[26] Lauri Tuominen et al. Mapping neurotransmitter networks with PET: an example on serotonin and opioid systems. *Human brain mapping* 35.5 (2014), pp. 1875–1884.

[27] Vincent Beliveau et al. A high-resolution in vivo atlas of the human brain's serotonin system. *Journal of Neuroscience* 37.1 (2017), pp. 120–128.

[28] Sjoerd J Finnema et al. Kinetic evaluation and test–retest reproducibility of [11C] UCB-J, a novel radioligand for positron emission tomography imaging of synaptic vesicle glycoprotein 2A in humans. *Journal of Cerebral Blood Flow & Metabolism* 38.11 (2018), pp. 2041–2052.

[29] Laura Pekkarinen et al. Lower abdominal adipose tissue cannabinoid type 1 receptor availability in young men with overweight. *Obesity* 31.7 (2023), pp. 1844–1858.

[30] Mark Jenkinson et al. FSL. *Neuroimage* 62.2 (2012), pp. 782–790.

[31] Rastko Ciric et al. TemplateFlow: FAIR-sharing of multi-scale, multi-species brain models. *Nature methods* 19.12 (2022), pp. 1568–1571.

[32] Nicholas J. Tustison et al. The ANTsX ecosystem for quantitative biological and medical imaging. *Scientific Reports* 11.1 (Apr. 2021), p. 9068. ISSN:

2045-2322. DOI: 10.1038/s41598-021-87564-6. URL: https://doi.org/10.1038/s41598-021-87564-6.

[33] Alan C Evans et al. Brain templates and atlases. *Neuroimage* 62.2 (2012), pp. 911–922.

[34] John C Mazziotta et al. A probabilistic atlas of the human brain: theory and rationale for its development. *Neuroimage* 2.2 (1995), pp. 89–101.

[35] Alexander Schaefer et al. Local-global parcellation of the human cerebral cortex from intrinsic functional connectivity MRI. *Cerebral cortex* 28.9 (2018), pp. 3095–3114.

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

[37] Colm J McGinnity et al. Alpha 5 subunit-containing GABAA receptors in temporal lobe epilepsy with normal MRI. *Brain communications* 3.1 (2021), fcaa190.

[38] Thomas Lin Pedersen. *patchwork: The Composer of Plots*. R package version 1.3.2.9000. 2025. URL: https://patchwork.data-imaginist.com.

[39] Christopher Rorden. MRIcroGL: Voxel-based visualization for neuroimaging. *Nature methods* (2025), pp. 1–2.

[40] Christopher Rorden. Surfice: Visualizing neuroimaging meshes, tractography streamlines and connectomes. *Nature Methods* (2025), pp. 1–2.

[41] Thomas Lewiner et al. Efficient implementation of marching cubes' cases with topological guarantees. *Journal of graphics tools* 8.2 (2003), pp. 1–15.

[42] Christopher Rorden et al. niimath and fslmaths: Replication as a method to enhance popular neuroimaging tools. *Aperture neuro* 4 (2024), pp. 10–52294.

[43] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3D surface construction algorithm. In: *Seminal graphics: pioneering efforts that shaped the field*. 1998, pp. 347–353.

[44] Christopher Rorden et al. brain2print AI powered web tool for creating 3D printable brain models. *Scientific Reports* 15.1 (2025), p. 15664.

[45] Christopher R Madan. Creating 3D visualizations of MRI data: A brief guide. *F1000Research* 4 (2015), p. 466.

[46] Paul A Yushkevich et al. User-guided 3D active contour segmentation of anatomical structures: significantly improved efficiency and reliability. *Neuroimage* 31.3 (2006), pp. 1116–1128.

[47] Utkarsh Ayachit. *The ParaView guide: A parallel visualization application*. Kitware, Inc., 2015.

[48] James S Gao et al. Pycortex: an interactive surface visualizer for fMRI. *Frontiers in neuroinformatics* 9 (2015), p. 23.

[49] Tim Schäfer and Christine Ecker. fsbrain: an R package for the visualization of structural neuroimaging data. *Biorxiv* (2020), pp. 2020–09.

[50] Federico Claudi et al. Visualizing anatomically registered data with brainrender. *Elife* 10 (2021), e65751.

[51] Josef Parvizi. Corticocentric myopia: old bias in new cognitive sciences. *Trends in cognitive sciences* 13.8 (2009), pp. 354–359.

[52] Marine Thomasson and Julie Péron. Principles of brain and emotion: beyond the cortico-centric bias. In: *The Emotional Cerebellum*. Springer, 2022, pp. 13–24.

[53] William Saban and Shai Gabay. Contributions of lower structures to higher cognition: towards a dynamic network model. *Journal of Intelligence* 11.6 (2023), p. 121.

[54] Karolina Janacsek et al. Subcortical cognition: the fruit below the rind. *Annual review of neuroscience* 45 (2022), pp. 361–386.

[55] Wenjiao Lyu et al. A multimodal submillimeter MRI atlas of the human cerebellum. *Scientific reports* 14.1 (2024), p. 5622.

[56] Juan Eugenio Iglesias et al. A probabilistic atlas of the human thalamic nuclei combining ex vivo MRI and histology. *Neuroimage* 183 (2018), pp. 314–326.

[57] Manojkumar Saranathan et al. In vivo high-resolution structural MRI-based atlas of human thalamic nuclei. *Scientific data* 8.1 (2021), p. 275.

[58] Jason H Su et al. Thalamus Optimized Multi Atlas Segmentation (THOMAS): fast, fully automated segmentation of thalamic nuclei from structural MRI. *Neuroimage* 194 (2019), pp. 272–282.

Appendix I. Code supplement

```
from subcortex_visualization.plotting import plot_subcortical_data
plot_subcortical_data(fill_title = "Demonstrating default parameters")
```

Code snippet 1. Demonstrating default parameters of `plot_subcortical_data`.

```
from subcortex_visualization.plotting import plot_subcortical_data

# A. Melbourne (S2) atlas example, plasma colormap
plot_subcortical_data(atlas = "Melbourne_S2", cmap="plasma", hemisphere='both',
                      fill_title = "Melbourne (S2) with plasma colormap")

# B. Brainnetome atlas example, rainbow colormap
plot_subcortical_data(atlas = "Brainnetome", cmap="hsv", hemisphere='both',
                      fill_title = "Brainnetome with rainbow colormap")
```

Code snippet 2. Visualizing the Melbourne (S2) and Brainnetome atlases with one color per subregion.

```
import numpy as np
import pandas as pd
# Set random seed for reproducibility
np.random.seed(127)

# Sample seven random values from a normal distribution for each hemisphere
example_continuous_data_L = (pd.DataFrame({"region": ["accumbens", "amygdala",
                                                      "caudate", "hippocampus",
                                                      "pallidum", "putamen",
                                                      "thalamus"],
                                           "value": np.random.normal(0, 1, 7)})
                                           .assign(Hemisphere = "L"))
example_continuous_data_R = (pd.DataFrame({"region": ["accumbens", "amygdala",
                                                      "caudate", "hippocampus",
                                                      "pallidum", "putamen",
                                                      "thalamus"],
                                             "value": np.random.normal(0, 1, 7)})
                                             .assign(Hemisphere = "R"))

# Combine left and right hemisphere data into a single DataFrame
# for plotting both hemispheres simultaneously
example_continuous_data = pd.concat([example_continuous_data_L,
                                     example_continuous_data_R], axis=0)
```

Code snippet 3. Simulating empirical data for the left hemisphere and aseg atlas.

```
from subcortex_visualization.plotting import plot_subcortical_data

plot_subcortical_data(subcortex_data=example_continuous_data,
                      atlas = 'aseg',
                      hemisphere='both',
                      fill_title = "Normal distribution sample, aseg atlas",
                      cmap='viridis')
```

Code snippet 4. Plotting simulated data with the left aseg atlas.

```python
from subcortex_visualization.plotting import plot_subcortical_data
import matplotlib.colors as mcolors

# Create a blue-white-red colormap
white_blue_red_cmap = mcolors.LinearSegmentedColormap.from_list("BlueWhiteRed",
                                                                ["blue", "white",
                                                                "red"])

plot_subcortical_data(subcortex_data=example_continuous_data,
                      atlas = 'aseg',
                      hemisphere='both',
                      fill_title = "Normal distribution sample, aseg atlas",
                      cmap=white_blue_red_cmap, midpoint=0)
```

Code snippet 5. Plotting simulated data with a custom colormap for the aseg atlas, left hemisphere.

```python
import templateflow
import templateflow.api as tflow
import ants

# Get the .xfm file for MNI152NLin2009cAsym to MNI152NLin6Asym
tflow.get(
        "MNI152NLin6Asym",
        suffix="xfm",
        desc=None,
        **{"from": "MNI152NLin2009cAsym"}
)

# Define paths for reference image and transform
ref_img = os.fspath(tflow.get(
    "MNI152NLin6Asym",
    resolution=1,
    desc=None,
    suffix="T1w"
))

# Define xfm
xfm = (
    "path/to/templateflow/" # Change to your local templateflow path
    "tpl-MNI152NLin6Asym/"
    "tpl-MNI152NLin6Asym_from-MNI152NLin2009cAsym_mode-image_xfm.h5"
)

# Load in image using ANTs
atlas_volume_native = ants.image_read(atlas_volume)
reference = ants.image_read(ref_img)

# Apply nearest neighbor interpolation to preserve labels
atlas_volume_warped = ants.apply_transforms(
    fixed=reference,
    moving=atlas_volume_native,
    transformlist=[xfm],
    interpolator="nearestNeighbor"
)
```

Code snippet 6. Example of how to align a segmentation atlas to MNI152NLin6Asym space.

```python
import numpy as np
import matplotlib
from nilearn.maskers import NiftiLabelsMasker
from subcortex_visualization import segmentation, plotting

# Define file paths
functional_map_path = "/path/to/CB1_density_mean.nii.gz"
atlas_name = "aseg"
func_name = "CB1_density"

# Extract segstats for this atlas and functional map
this_func_atlas_data = segmentation.apply_atlas_to_data(functional_map_path,
                                                        atlas_name,
                                                        func_name=func_name)

# Custom color map
custom_cmap = mcolors.LinearSegmentedColormap.from_list("this_atlas_cmap",
                                                        ["white", "#00405c"])

this_plot = plotting.plot_subcortical_data(subcortex_data=this_func_atlas_data,
  atlas=atlas_name, value_column='Mean_Signal', fill_title=this_fill_title,
  hemisphere='R', show_legend=False, show_figure=False, cmap=custom_cmap
)

# Add an overall title for the figure
this_plot.suptitle(func_name, fontsize=16)
plt.tight_layout()
```

Code snippet 7. Computing and plotting region-averaged CB1 densities in the aseg subcortical atlas.

```python
import numpy as np
import matplotlib
from nilearn.maskers import NiftiLabelsMasker
from subcortex_visualization import segmentation, plotting

all_atlases = ['aseg', 'Melbourne_S1', 'Melbourne_S2', 'Melbourne_S3',
'Melbourne_S4', 'AICHA', 'Brainnetome',
'Thalamus_Nuclei_HCP', 'SUIT_cerebellar_lobule']

functional_map_path = "/path/to/CB1_density_mean.nii.gz"
func_name = "CB1_density"

# Extract segstats for all atlases and this functional map
all_parc_data = segmentation.apply_atlas_to_data(functional_map_path,
                                                 all_atlases,
                                                 func_name=func_name)

# Get min and max values across all atlases for consistent color scaling
this_vmin = all_parc_data['Mean_Signal'].min()
this_vmax = all_parc_data['Mean_Signal'].max()

# Plot each atlas with consistent color scale
fig, axes = plt.subplots(3, 3, figsize=(15, 10))
for ax, atlas in zip(axes.flat, all_atlases):
    this_atlas_data = all_parc_data.query("Atlas == @atlas")
    this_fill_title = f"{this_atlas_data.Functional_Map.unique()[0]}, {atlas}"

    plotting.plot_subcortical_data(
        subcortex_data=this_atlas_data, atlas=atlas,
        value_column='Mean_Signal', fill_title=this_fill_title,
        hemisphere='R', vmin=this_vmin, vmax=this_vmax,
        ax=ax, show_legend=False, show_figure=False
    )

# Add an overall title for the figure
fig.suptitle(f"Functional Map: {func_name}", fontsize=16)

plt.tight_layout()
```

Code snippet 8. Computing and plotting region-averaged CB1 densities across all nine provided atlases.

```python
# Packages for parcellating cortical+subcortical data
from subcortex_visualization import plotting, segmentation
from netneurotools import datasets as nntdata
import neuromaps
from neuromaps import transforms, images
from neuromaps.parcellate import Parcellater
from neuromaps.images import dlabel_to_gifti
import nilearn

# Add connectome workbench to path
os.environ['PATH'] = os.environ['PATH'] + '/path/to/connectome/workbench/bin'

# Transform functional map to fsaverage surface
functional_map_path = "/path/to/GABAa1_density_mean.nii.gz"
GABAa1_surf = transforms.mni152_to_fsaverage(functional_map_path, method='linear')

# Load Schaefer1000 (17-network) region names
schaefer1000_labels = list(np.loadtxt('1000Parcels_17Networks.txt',
                                      dtype=str, usecols=0))
schaefer1000_labels = (lambda x: x.startswith('17Networks'), schaefer1000_labels)

# Download Schaefer parcellations in fsaverage space
schaefer_fsaverage = nntdata.fetch_schaefer2018(version='fsaverage')
schaefer_fsaverage_1000 = schaefer_fsaverage['1000Parcels17Networks']

# Convert 1000-parcellation into gifti space
schaefer_res_gifti = neuromaps.images.annot_to_gifti(schaefer_fsaverage_1000)
schaefer_res_gifti = neuromaps.images.relabel_gifti(schaefer_res_gifti)

# Fit a Parcellator object
parc = Parcellater(schaefer_res_gifti, 'fsaverage', resampling_target='parcellation')
func_map_in_surf_parc = parc.fit_transform(func_map_in_surf, 'fsaverage')

# Create DataFrame for plotting
func_map_cortex_parc_df = (pd.DataFrame({
    'Region_Label': schaefer1000_labels,
    'Atlas': 'Schaefer1000_17Networks',
    'Mean_Signal': func_map_in_surf_parc})
    .assign(Hemisphere = lambda x: np.where(x['Region_Label'].str.contains('LH'),
                                            'L', 'R'))
)

# Define functional map and subcortical parcellations
subctx_cbm_atlases = ['Melbourne_S4', 'SUIT_cerebellar_lobule']
subctx_CBM_GABAa1 = segmentation.apply_atlas_to_data(functional_map_path,
                                                     subctx_cbm_atlases,
                                                     func_name='GABAa1_density')

# FIlter to align with cortical data
func_map_subcortex_CBM_parc_df = func_map_subcortex_CBM_parc_df[['region',
    'Mean_Signal', 'Hemisphere', 'Atlas']]
```

Code snippet 9. Parcellate GABAa1 density map with cortex, subcortex, and cerebellum atlases.

```
%%R -i func_map_cortex_parc_df,func_map_subcortex_CBM_parc_df

library(subcortexVisualizationR)
library(ggseg)
library(ggsegSchaefer)
library(patchwork)
library(tidyverse)

functional_map_path = "/path/to/GABAa1_density_mean.nii.gz" # PET map path

# Find the min/max values across all GABAa1 data for consistent color scaling
func_map_min <- min(c(func_map_cortex_parc_df$Mean_Signal,
                      func_map_subcortex_CBM_parc_df$Mean_Signal), na.rm=TRUE)
func_map_max <- max(c(func_map_cortex_parc_df$Mean_Signal,
                       func_map_subcortex_CBM_parc_df$Mean_Signal), na.rm=TRUE)

# Cortical plot with ggseg
cortex_p <- func_map_cortex_parc_df %>%
    filter(Atlas == "Schaefer1000_17Networks") %>%
    mutate(label = ifelse(str_detect(Region_Label, "_LH_"),
                   paste0("lh_", Region_Label), paste0("rh_", Region_Label))) %>%
    ggseg(atlas = schaefer17_1000, mapping = aes(fill = Mean_Signal),
          colour = "black", linewidth=0.1) +
    ggtitle("Schaefer-1000 Cortex") +
    labs(fill='Mean GABAa1 Signal (SUVR)') +
    scale_fill_viridis_c(limits=c(func_map_min, func_map_max),
                                 na.value='gray80', option='plasma') +
    # Set colorbar as 5cm wide and 1cm tall, with legend title on top
    guides(fill=guide_colorbar(barwidth = unit(5, "cm"),
                               barheight = unit(0.5, "cm"),
                               title.position="top", title.hjust = 0.5)) +
    theme(legend.position = 'bottom', plot.title = element_text(hjust=0.5))

# Melbourne S4 subcortex plot
subcortex_p <- plot_subcortical_data(subcortex_data = all_func_parc_data,
                      atlas = "Melbourne_S4", hemisphere = "L",
                      value_column='Mean_Signal', vmin=func_map_min,
                      vmax=func_map_max, cmap=custom_cmap, show_legend=F) +
                      ggtitle("Melbourne S4 Subcortex") +
                      theme(plot.title = element_text(hjust=0.5)) +
  guides(fill=guide_colorbar(barwidth = unit(5, "cm"), barheight = unit(0.5, "cm"),
                             title.position="top", title.hjust = 0.5))
# SUIT cerebellum plot
cerebellum_p <- plot_subcortical_data(subcortex_data = all_func_parc_data,
                      atlas = "SUIT_cerebellar_lobule", hemisphere = "L",
                      value_column='Mean_Signal', vmin=func_map_min,
                      vmax=func_map_max, cmap=custom_cmap, show_legend=F) +
                      ggtitle("SUIT Cerebellum") +
                      theme(plot.title = element_text(hjust=0.5)) +
  guides(fill=guide_colorbar(barwidth = unit(5, "cm"), barheight = unit(0.5, "cm"),
                             title.position="top", title.hjust = 0.5))

# Use patchwork to combine the plots
plot_design <- "AAA
                BBC"
wrap_plots(A=cortex_p, B=subcortex_p, C=cerebellum_p,
           design=plot_design, heights=c(0.6, 0.4))
```

Code snippet 10. Plotting GABAa1 receptor density in the cortex, subcortex, and cerebellum all together.

```python
import numpy as np
import matplotlib

cmap = matplotlib.colormaps.get_cmap('plasma') # Use plasma colormap
colors = cmap(np.linspace(0, 1, 8)) # Sample 8 colors
rgb_colors = colors[:, :3] # Extract RGB (no alpha)
rgb_256 = (rgb_colors * 256).astype(int) # 0-256 scaling
np.savetxt('my_colors.txt', rgb_256, fmt='%d') # Save colors to .txt file
```

Code snippet 11. Generating a custom RGB colormap for the 3D mesh object.

```bash
python volume_to_mesh_mz3.py \
--input_volume ../example_segmentation/Tian_Subcortex_S1_3T_1mm.nii.gz \
--output_path ../mesh_outputs/ --out_file Tian_Subcortex_S1_3T_1mm.mz3 \
--index_max 8 --colors plasma_8_colors.txt --delete_mz3
```

Code snippet 12. Rendering the segmentation volume as an .mz3 mesh object for use in Surf Ice.

Appendix II. Supplementary Figures

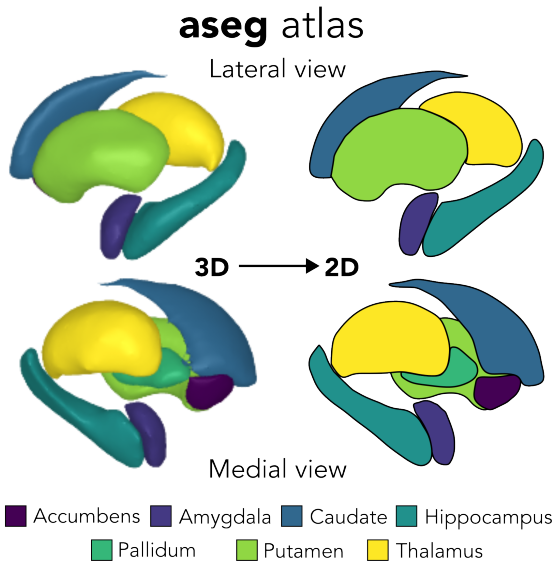

Figure S1. aseg atlas.

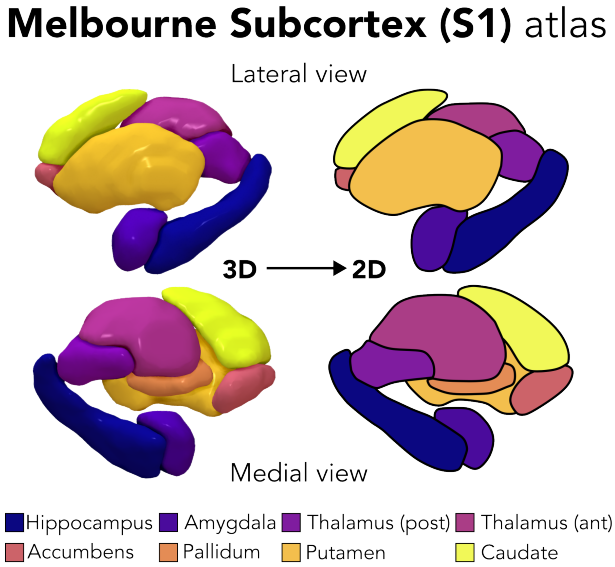

Figure S2. Melbourne (S1) subcortical atlas .

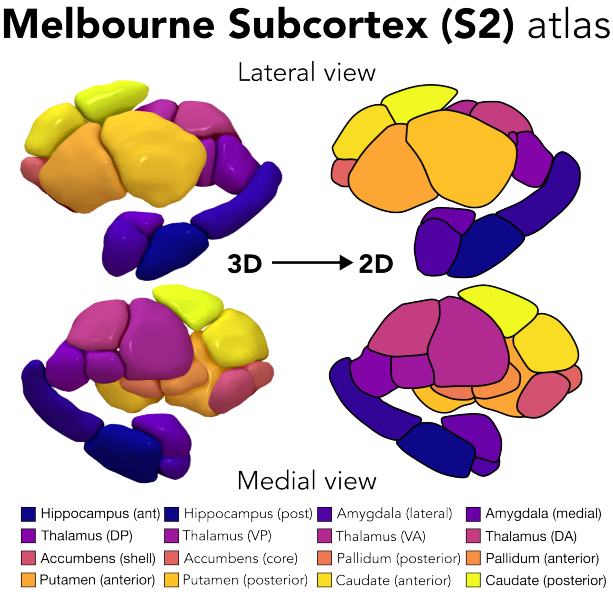

Figure S3. Melbourne (S2) subcortical atlas.

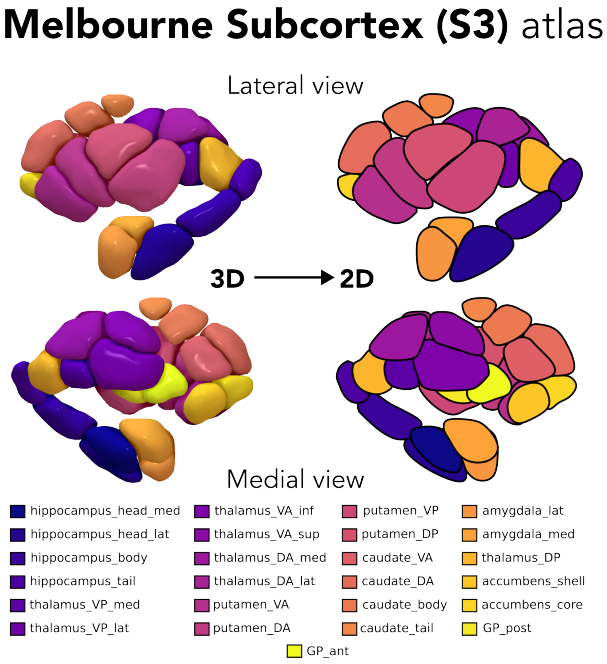

Figure S4. Melbourne (S3) subcortical atlas.

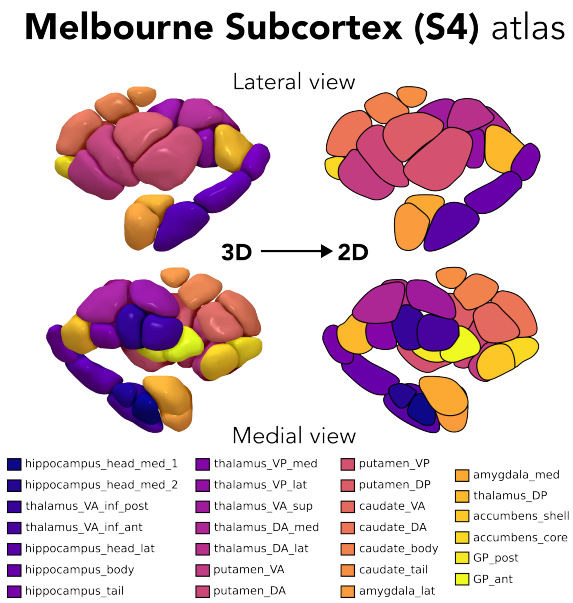

Figure S5. Melbourne (S4) subcortical atlas.

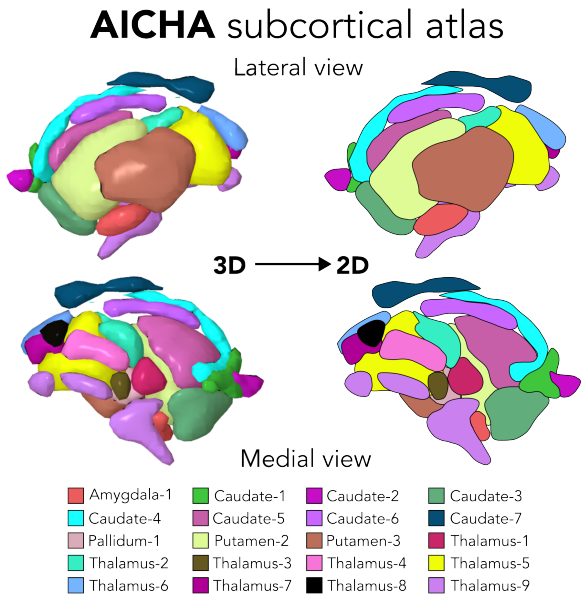

Figure S6. AICHA subcortical atlas.

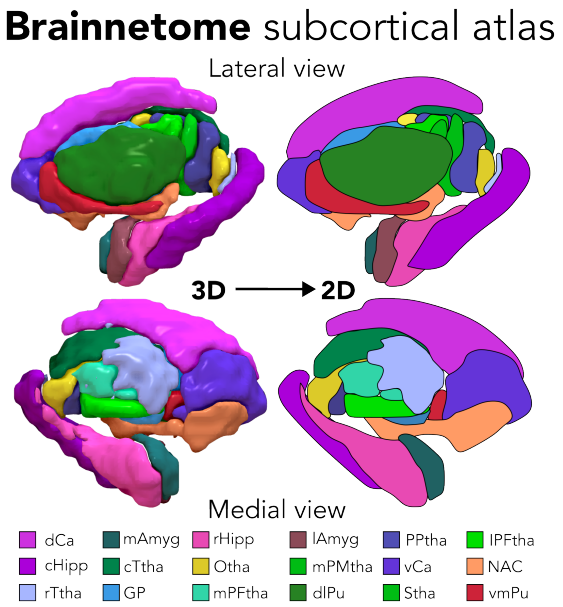

Figure S7. Brainnetome subcortical atlas.

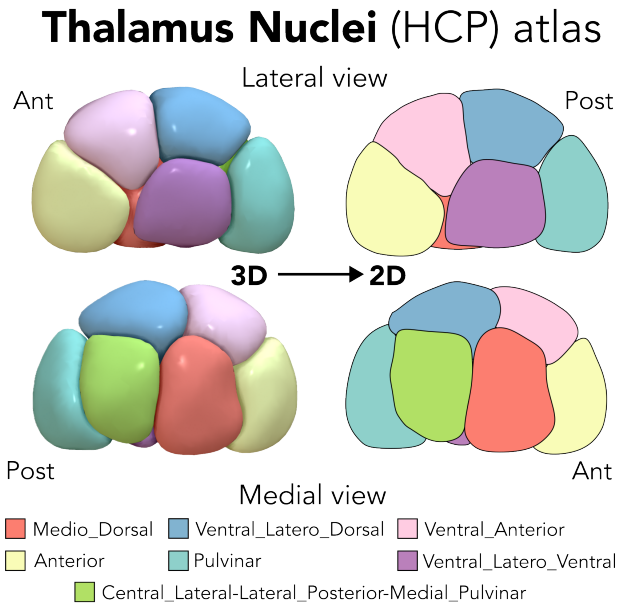

Figure S8. Thalamic nuclei (HCP) atlas.

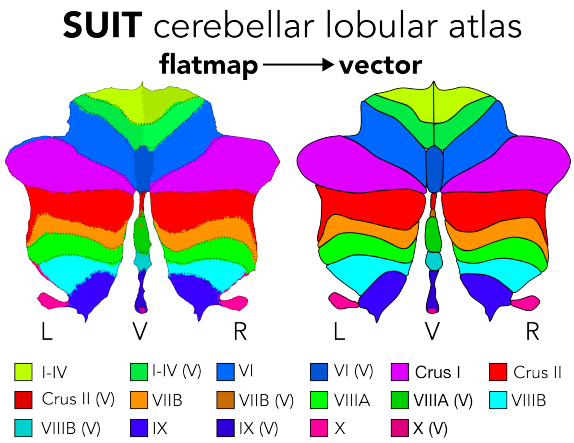

Figure S9. SUIT cerebellar lobular atlas.

Appendix III. Supplementary tables

Table S1. Individual regions within each atlas (per hemisphere).

Atlas	Regions
aseg	accumbens, amygdala, caudate, hippocampus, pallidum, putamen, thalamus
Melbourne (S1)	accumbens, amygdala, caudate, hippocampus, pallidum, putamen, thalamus_anterior, thalamus_posterior
Melbourne (S2)	accumbens_core, accumbens_shell, amygdala_lateral, amygdala_medial, caudate_anterior, caudate_posterior, hippocampus_anterior, hippocampus_posterior, pallidum_anterior, pallidum_posterior, putamen_anterior, putamen_posterior, thalamus_DA, thalamus_DP, thalamus_VA, thalamus_VP
Melbourne (S3)	accumbens_core, accumbens_shell, amygdala_lateral, amygdala_medial, caudate_anterior, caudate_posterior, hippocampus_anterior, hippocampus_posterior, pallidum_anterior, pallidum_posterior, putamen_anterior, putamen_posterior, thalamus_DA, thalamus_DP, thalamus_VA, thalamus_VP
Melbourne (S4)	accumbens_core, accumbens_shell, amygdala_lateral, amygdala_medial, caudate_anterior, caudate_posterior, hippocampus_anterior, hippocampus_posterior, pallidum_anterior, pallidum_posterior, putamen_anterior, putamen_posterior, thalamus_DA, thalamus_DP, thalamus_VA, thalamus_VP
AICHA	Amygdala-1, Caudate-1, Caudate-2, Caudate-3, Caudate-4, Caudate-5, Caudate-6, Caudate-7, Pallidum-1, Putamen-2, Putamen-3, Thalamus-1, Thalamus-2, Thalamus-3, Thalamus-4, Thalamus-5, Thalamus-6, Thalamus-7, Thalamus-8, Thalamus-9
Brainnetome	GP, NAC, Otha, PPtha, Stha, cHipp, cTtha, dCa, dlPu, lAmyg, lPFtha, mAmyg, mPFtha, mPMtha, rHipp, rTtha, vCa, vmPu
Thalamus Nuclei (HCP)	Anterior, Central_Lateral-Lateral_Posterior-Medial_Pulvinar, Medio_Dorsal, Pulvinar, Ventral_Anterior, Ventral_Latero_Dorsal, Ventral_Latero_Ventral
SUIT cerebellar lobules	I_IV, V, VI, VI_vermis, Crus_I, Crus_II, Crus_II_vermis, VIIb, VIIb_vermis, VIIIa, VIIIa_vermis, VIIIb, VIIIb_vermis, IX, IX_vermis, X, X_vermis

Appendix IV. Supplementary Methods

aseg subcortical atlas

While we created the 2D vector image directly from the ENIGMA-generated 3D mesh of the aseg atlas, we also include the volumetric segmentation in MNI152 space (processed by Thomas Yeo's Computational Brain Imaging Group, known as CBIG) so that users can analyze their own data using this atlas. This segmentation atlas is included with FreeSurfer installations (https://surfer.nmr.mgh.harvard.edu/), though we obtained a version from the CBIG repository, in which FreeSurfer's 'recon-all' pipeline (version 4.5.0) was applied to the FSL MNI152 1mm isotropic template. The results of this pipeline are openly available at https://github.com/ThomasYeoLab/CBIG/tree/master/data/templates/volume/FSL_MNI152_FS4.5.0/.

Melbourne Subcortex Atlas

The Melbourne Subcortex Atlas (S1, S2, S3, and S4 resolutions) were obtained from the main repository accompanying the original publication [8]. Specifically, the NIfTI files were downloaded from https://github.com/yetianmed/subcortex/tree/master/Group-Parcellation/3T/Subcortex-Only. For each atlas resolution, we downloaded the 3T volume resampled onto a 1mm isotropic anatomical grid in MNI152 Nonlinear 6th generation (MNI152NLin6Asym) space.

AICHA subcortical atlas

The AICHA (v2) subcortical atlas NIfTI volume was obtained from EBRAINS at https://search.kg.ebrains.eu/instances/Dataset/b820fbed-d920-43c4-9b85-2e6be3d04cd1 (file name: 'AICHA1mm.nii'). The full citation for the atlas dataset is provided below. The subcortical indices range from 345-384, which we used to extract the subcortical regions from the full atlas volume.

Joliot, M., & Naveau, M. (2021). AICHA: An atlas of intrinsic connectivity of homotopic areas (Version 2) [Data set]. EBRAINS. DOI: 10.25493/5NX4-HXR

Brainnetome subcortical atlas

The Brainnetome atlas (including cortical and subcortical regions) was downloaded from the atlas website at http://www.brainnetome.org/resource/201910/t20191030_522618.html. Specifically, the file downloaded was titled 'BN_Atlas_246_1mm.nii.gz'. The subcortical indices range from 211-246, which we used to extract the subcortical regions from the full atlas volume. This atlas is provided in MNI152NLin2009cSym space.

Thalamus nuclei (HCP) atlas

The thalamic nuclei atlas based on HCP data was obtained from Zenodo at https://zenodo.org/record/1405484. The full citation for the atlas dataset is provided below. This atlas is provided in MNI152NLin2009cSym space.

Najdenovska, E., Alemán-Gómez, Y., & Bach Cuadra, M. (2018). Dataset In-vivo probabilistic atlas of human thalamic nuclei based on diffusion weighted magnetic resonance imaging (1.0) [Data set]. Zenodo. https://doi.org/10.5281/zenodo.1405484

SUIT cerebellar lobule atlas

For volumetric segmentation, we downloaded the probabilistic atlas in MNI152NLin6Asym space from the Diedrichsen Lab GitHub repository at the following link: https://github.com/DiedrichsenLab/cerebellar_atlases/blob/master/Diedrichsen_2009/atl-Anatom_space-MNI_probseg.nii.