Appendix I. Code supplement

```python
from subcortex_visualization.plotting import plot_subcortical_data
plot_subcortical_data(fill_title = "Demonstrating default parameters", atlas='Melbourne_S1')
```

Code snippet 1. Demonstrating default parameters of `plot_subcortical_data` [Python].

```r
library(subcortexVisualizationR)
plot_subcortical_data(fill_title = "Demonstrating default parameters", atlas = "Melbourne_S1")
```

Code snippet 2. Demonstrating default parameters of `plot_subcortical_data` [R].

```python
from subcortex_visualization.plotting import plot_subcortical_data

# A. AICHA subcortex atlas example, plasma colormap
plot_subcortical_data(atlas = "AICHA_subcortex", cmap="plasma", hemisphere='both',
                      fill_title = "AICHA with plasma colormap")

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

Code snippet 3. Visualizing the AICHA and Brainnetome subcortex atlases with one color per subregion [Python].

```r
library(subcortexVisualizationR)

# A. AICHA subcortex atlas example, plasma colormap
plot_subcortical_data(atlas = "AICHA_subcortex", cmap="plasma", hemisphere='both',
                      fill_title = "AICHA with plasma colormap")

# B. Brainnetome subcortex atlas example, rainbow colormap
rainbow_cmap <- colorRampPalette(RColorBrewer::brewer.pal(11, 'Spectral'))
plot_subcortical_data(atlas = "Brainnetome_subcortex", cmap=rainbow_cmap, hemisphere='both',
                      fill_title = "Brainnetome subcortex with rainbow colormap")
```

Code snippet 4. Visualizing the AICHA and Brainnetome subcortex atlases with one color per subregion [R].

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

# A. THOMAS thalamic nuclei atlas
THOMAS_cmap = mcolors.LinearSegmentedColormap.from_list("", ["white", "#d14662"])
plot_subcortical_data(atlas = "Thalamus_THOMAS", cmap=THOMAS_cmap, hemisphere='both',
                      views = ['lateral', 'medial', 'superior', 'inferior'],
                      show_legend=False, fill_title = "THOMAS thalamic nuclei")

# B. Brainstem Navigator atlas example, rainbow colormap
Brainstem_cmap = mcolors.LinearSegmentedColormap.from_list("", ["white", "#c8499b"])
plot_subcortical_data(atlas = "Brainstem_Navigator", cmap=Brainstem_cmap, hemisphere='both',
                      views = ['lateral', 'medial', 'superior', 'inferior'],
                      show_legend=False,
                      fill_title = "Brainstem Navigator with rainbow colormap")
```

Code snippet 5. Demonstrating the four available view angles with the THOMAS thalamic nuclei atlas and Brainstem Navigator atlas [Python].

```r
library(subcortexVisualizationR)

# A. THOMAS thalamic nuclei atlas
THOMAS_cmap <- colorRampPalette(c("white", "#d14662"))
plot_subcortical_data(atlas = "Thalamus_THOMAS", cmap=THOMAS_cmap, hemisphere='both',
                      views = c('lateral', 'medial', 'superior', 'inferior'),
                      show_legend=FALSE, fill_title = "THOMAS thalamic nuclei")

# B. Brainstem Navigator atlas example, rainbow colormap
Brainstem_cmap <- colorRampPalette(c("white", "#c8499b"))
plot_subcortical_data(atlas = "Brainstem_Navigator", cmap=Brainstem_cmap, hemisphere='both',
                      views = c('lateral', 'medial', 'superior', 'inferior'),
                      show_legend=FALSE,
                      fill_title = "Brainstem Navigator with rainbow colormap")
```

Code snippet 6. Demonstrating the four available view angles with the THOMAS thalamic nuclei atlas and Brainstem Navigator atlas [R].

```python
import numpy as np
import pandas as pd
from subcortex_visualization.utils import get_atlas_regions

np.random.seed(127)
aseg_subcortex_regions = get_atlas_regions("aseg_subcortex")

# Sample eight random values from a normal distribution for each hemisphere
example_continuous_data_L = (pd.DataFrame({"region": aseg_subcortex_regions,
                                           "value": np.random.normal(0, 1,
                                                                     len(aseg_subcortex_regions))})
                                          .assign(Hemisphere = "L"))
example_continuous_data_R = (pd.DataFrame({"region": aseg_subcortex_regions,
                                           "value": np.random.normal(0, 1,
                                                                     len(aseg_subcortex_regions))})
                                          .assign(Hemisphere = "R"))

# Combine left and right hemisphere data into a single DataFrame
example_continuous_data = pd.concat([example_continuous_data_L, example_continuous_data_R],
                                    axis=0)[["region", "Hemisphere", "value"]]
```

Code snippet 7. Simulating empirical data for the left hemisphere of the aseg subcortex atlas [Python].

```r
library(subcortexVisualizationR)
set.seed(127)
aseg_subcortex_regions <- get_atlas_regions("aseg_subcortex")

# Sample eight random values from a normal distribution for each hemisphere
example_continuous_data_L <- data.frame(region = aseg_subcortex_regions,
                                        Hemisphere = "L",
                                        value = rnorm(length(aseg_subcortex_regions),
                                        mean=0, sd=1))
example_continuous_data_R <- data.frame(region = aseg_subcortex_regions,
                                        Hemisphere = "R",
                                        value = rnorm(length(aseg_subcortex_regions),
                                        mean=0, sd=1))

# Combine left and right hemisphere data into a single DataFrame
example_continuous_data <- rbind(example_continuous_data_L, example_continuous_data_R)
```

Code snippet 8. Simulating empirical data for the left hemisphere of the aseg subcortex atlas [R].

```python
from subcortex_visualization.plotting import plot_subcortical_data

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

Code snippet 9. Plotting simulated data for the left hemisphere of the aseg subcortex atlas [Python].

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

Code snippet 10. Plotting simulated data for the left hemisphere of the aseg subcortex atlas [R].

```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_subcortex',
                      hemisphere='both',
                      fill_title = "Normal distribution sample, aseg subcortex atlas",
                      cmap=white_blue_red_cmap, midpoint=0)
```

Code snippet 11. Plotting simulated data from Code snippet 7 with a custom colormap for the aseg subcortex atlas, left hemisphere [Python].

```r
library(subcortexVisualizationR)
library(ggplot2)

# Create a blue-white-red colormap
white_blue_red_cmap <- colorRampPalette(c("blue", "white", "red"))

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

Code snippet 12. Plotting simulated data from Code snippet 8 with a custom colormap for the aseg subcortex atlas, left hemisphere [R].

```python
import numpy as np
from subcortex_visualization.plotting import plot_subcortical_data

for fill_alpha in np.arange(0.2, 1.1, 0.2):
    this_alpha_plot = plot_subcortical_data(atlas='Melbourne_S2', fill_alpha=fill_alpha,
                                            cmap='plasma', show_legend=False)
```

Code snippet 13. Demonstrating the `fill_alpha` transparency parameter [Python].

```r
library(subcortexVisualizationR)

for (fill_alpha in seq(0.2, 1, by=0.2)) {
    this_alpha_plot <- plot_subcortical_data(atlas='Melbourne_S2', fill_alpha=fill_alpha,
                                             cmap='plasma', show_legend=FALSE)}
```

Code snippet 14. Demonstrating the `fill_alpha` transparency parameter [R].

```python
from subcortex_visualization.plotting import plot_subcortical_data
from subcortex_visualization.utils import get_atlas_regions

# Simulate some continuous data for plotting
np.random.seed(128)
this_atlas = 'SUIT_cerebellar_lobule'
this_atlas_hemisphere_regions, this_atlas_vermis_regions = get_atlas_regions(this_atlas)

this_atlas_simdata = pd.DataFrame({
    'region': np.concatenate([this_atlas_hemisphere_regions,
                              this_atlas_hemisphere_regions, this_atlas_vermis_regions]),
    'Hemisphere': ['L']*len(this_atlas_hemisphere_regions) +
                  ['R']*len(this_atlas_hemisphere_regions) +
                  ['V']*len(this_atlas_vermis_regions)})
this_atlas_simdata['value'] = np.random.normal(loc=0, scale=1, size=len(this_atlas_simdata))
this_atlas_simdata['p_value'] = np.random.uniform(low=0.05, high=1.0,
                                                  size=len(this_atlas_simdata))

# Set the four largest-magnitude values to be significant (p<0.05)
largest_indices = np.argsort(np.abs(this_atlas_simdata['value']))[-4:]
this_atlas_simdata.loc[largest_indices, 'p_value'] = 0.01

this_atlas_simdata.head()

plot_subcortical_data(atlas=this_atlas, subcortex_data=this_atlas_simdata, hemisphere='both',
                      midpoint=0, cmap='PiYG', show_legend=True, line_thickness=3,
                      fill_title="Simulated statistical test results",
                      fill_alpha=1.0, fill_by_significance=True, nonsig_fill_alpha=0.5)
```

Code snippet 15. Simulating statistical significance data to visualize in the SUIT cerebellar lobule atlas [Python].

```r
library(subcortexVisualizationR)

# Note that R and Python have different random number generators
set.seed(128)

this_atlas <- 'SUIT_cerebellar_lobule'
this_atlas_regions <- get_atlas_regions(this_atlas)
this_atlas_hemisphere_regions <- this_atlas_regions$hemisphere_regions
this_atlas_vermis_regions <- this_atlas_regions$vermis_regions

# Simulate some continuous data for plotting
this_atlas_simdata <- data.frame(
    region = c(this_atlas_hemisphere_regions, this_atlas_hemisphere_regions,
               this_atlas_vermis_regions),
    Hemisphere = c(rep('L', length(this_atlas_hemisphere_regions)),
                   rep('R', length(this_atlas_hemisphere_regions)),
                   rep('V', length(this_atlas_vermis_regions)))
)
this_atlas_simdata$value <- rnorm(nrow(this_atlas_simdata), mean=0, sd=1)
this_atlas_simdata$p_value <- runif(nrow(this_atlas_simdata), min=0.05, max=1.0)

# Set the four largest-magnitude values to be significant (p<0.05)
largest_indices <- order(abs(this_atlas_simdata$value), decreasing=TRUE)[1:4]
this_atlas_simdata$p_value[largest_indices] <- 0.01

# PiYG-inspired colormap
PiYG_cmap <- colorRampPalette(c("#C51B7D", "#E9A3C9", "#FDE0EF", "#F7F7F7",
                                "#E6F5D0", "#A1D76A", "#4D9221"))
plot_subcortical_data(atlas=this_atlas, subcortex_data=this_atlas_simdata, hemisphere='both',
                      midpoint=0, cmap=PiYG_cmap, show_legend=TRUE, line_thickness=1.5,
                      fill_title="Simulated statistical test results",
                      fill_alpha=1.0, fill_by_significance=TRUE, nonsig_fill_alpha=0.5)
```

Code snippet 16. Simulating statistical significance data to visualize in the SUIT cerebellar lobule atlas [R].

```
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_subcortex"
func_name = "CB1_density"

# The PET maps are shared in MNI152NLin6Asym space so we specify
# this space for the atlas to ensure proper alignment
atlas_space='MNI152NLin6Asym'

# Apply the aseg subcortex atlas to the functional map to extract
# the median value per region, specifying interpolation = 'nearest'
# to preserve discrete labels in the atlas
this_func_atlas_data = segmentation.parcel_segstats(input_vol = functional_map_path,
                                                    atlas_space = atlas_space,
                                                    atlas = atlas_name,
                                                    func_name=func_name,
                                                    parc_stat = np.median,
                                                    interpolation = 'nearest'
                                                    )

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

plotting.plot_subcortical_data(subcortex_data=this_func_atlas_data,
            atlas=atlas_name, value_column='value',
            fill_title='Median CB1 density - aseg subcortex atlas',
            hemisphere='both', show_legend=True, show_figure=True, cmap=custom_cmap
)
```

Code snippet 17. Computing and plotting region-averaged CB1 densities in the aseg subcortical atlas [Python].

```r
library(subcortexVisualizationR)
library(ggplot2)

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

# The PET maps are shared in MNI152NLin6Asym space so we specify
# this space for the atlas to ensure proper alignment
atlas_space='MNI152NLin6Asym'

# Apply the aseg subcortex atlas to the functional map to extract
# the median value per region, specifying interpolation = 'nearest'
# to preserve discrete labels in the atlas
this_func_atlas_data <- parcel_segstats(input_vol = functional_map_path,
                                        atlas_space = atlas_space,
                                        atlas = atlas_name,
                                        func_name=func_name,
                                        parc_stat = median,
                                        interpolation = 'nearest'
                                        )

# Custom color map
custom_cmap <- colorRampPalette(c("white", "#00405c"))

this_plot <- plot_subcortical_data(subcortex_data=this_func_atlas_data,
                atlas=atlas_name, value_column='value',
                fill_title='Median CB1 density - aseg subcortex atlas',
                hemisphere='both', show_legend=TRUE, cmap=custom_cmap
                )

# Add an overall title for the figure
this_plot
```

Code snippet 18. Computing and plotting region-averaged CB1 densities in the aseg subcortical atlas [R].

```python
import numpy as np
import matplotlib.pyplot as plt
from subcortex_visualization import segmentation, plotting

# For cortical data visualization
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

# Add connectome workbench to path; note you will need to change for your system
os.environ['PATH'] = os.environ['PATH'] + ':/Applications/workbench/bin_macosx64'

# Define path to functional maps
functional_map_path = "/path/to/CB1_density_mean.nii.gz"

# Transform functional map to fsaverage surface
func_map_in_surf = transforms.mni152_to_fsaverage(functional_map_path, method='nearest')

# Download and read in the Schaefer 1000-parcel lookup table
s1000_url = (
    "https://raw.githubusercontent.com/ThomasYeoLab/CBIG/master/stable_projects/"
    "brain_parcellation/Schaefer2018_LocalGlobal/Parcellations/MNI/freeview_lut/"
    "Schaefer2018_1000Parcels_17Networks_order.txt"
)
schaefer1000_lookup = pd.read_csv(s1000_url, header=None, sep='\t')[[0,1]]
schaefer1000_lookup.columns = ['index', 'label']
schaefer1000_labels = schaefer1000_lookup.label.tolist()

# 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['1000Parcels17Networks'])
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',
                                           ignore_background_data=True)

# Create a dataframe with the parcellated cortical data
func_map_cortical_parc_df = (pd.DataFrame({
    'region': schaefer1000_labels,
    'Atlas': 'Schaefer1000_17Networks',
    'value': func_map_in_surf_parc})
    .assign(Hemisphere = lambda x: np.where(x['region'].str.contains('LH'), 'L', 'R'))
)
```

Code snippet 19. Computing the region-averaged $GABA_A$-$\alpha 1$ densities for the Schaefer-1000 cortical parcellation [Python].

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from subcortex_visualization import segmentation, plotting

# Define subcortical atlases
all_atlases = ['aseg_subcortex', 'Melbourne_S1', 'Melbourne_S2',
               'Melbourne_S3', 'Melbourne_S4', 'AICHA_subcortex',
               'Brainnetome_subcortex', 'CIT168_subcortex', 'Thalamus_HCP',
               'Thalamus_THOMAS', 'Brainstem_Navigator', 'SUIT_cerebellar_lobule']

# Use the mean (excluding NaNs) as the summary statistic for parcellation
parc_stat = np.nanmean
atlas_space='MNI152NLin6Asym'

func_map_path = "/path/to/GABAa1_density_mean.nii.gz" # PET map path
func_name = "GABA_A_a1_density"

# Extract mean values for all atlases from the GABA_A_a1 density map
# Use nearest neighbor interpolation to preserve discrete labels in the atlas
func_map_subcortical_parc_df = segmentation.parcel_segstats(input_vol=func_map_path,
                                                atlas=all_atlases,
                                                atlas_space='MNI152NLin6Asym',
                                                parc_stat = parc_stat,
                                                ignore_background=True,
                                                func_name=func_name,
                                                interpolation='nearest')

# Combine with cortical data from previous code chunk
func_map_all_parc_df = pd.concat([func_map_cortical_parc_df, func_map_subcortical_parc_df],
                                 axis=0)

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

# Plot each atlas with consistent color scale
for atlas in all_atlases:
    this_atlas_data = func_map_all_parc_df.query("Atlas == @atlas")
    this_fill_title = f"{func_name} - {atlas}"

    this_atlas_plot = plotting.plot_subcortical_data(
        subcortex_data=this_atlas_data,
        atlas=atlas,
        value_column='value',
        fill_title=this_fill_title,
        hemisphere='L',
        vmin=this_vmin,
        vmax=this_vmax,
        cmap='plasma',
        show_legend=False,
        show_figure=False
)
```

Code snippet 20. Computing and plotting the region-averaged $GABA_A$-$\alpha 1$ densities in all twelve provided subcortical/cerebellar atlases [Python].

```r
library(subcortexVisualizationR)
library(ggplot2)

# Define subcortical atlases
all_atlases <- c('aseg_subcortex', 'Melbourne_S1', 'Melbourne_S2',
                 'Melbourne_S3', 'Melbourne_S4', 'AICHA_subcortex',
                 'Brainnetome_subcortex', 'CIT168_subcortex', 'Thalamus_HCP',
                 'Thalamus_THOMAS', 'Brainstem_Navigator', 'SUIT_cerebellar_lobule')

atlas_space='MNI152NLin6Asym'
functional_map_path = "/path/to/GABAa1_density_mean.nii.gz"
func_name = "GABA_A_a1_density"

# Extract mean values for all atlases from the GABA_A_a1 density map
# Use nearest neighbor interpolation to preserve discrete labels in the atlas
func_map_subcortical_parc_df <- parcel_segstats(input_vol=functional_map_path,
                                                atlas=all_atlases,
                                                atlas_space='MNI152NLin6Asym',
                                                parc_stat = median,
                                                ignore_background=TRUE,
                                                func_name=func_name,
                                                interpolation='nearest')

# Combine with cortical data from previous code chunk (need to import with rpy2)
func_map_all_parc_df <- plyr::rbind.fill(func_map_cortical_parc_df,
                                         func_map_subcortical_parc_df)

# Get min and max values across all atlases for consistent color scaling
min_func_parc_value <- min(func_map_all_parc_df$value)
max_func_parc_value <- max(func_map_all_parc_df$value)

# Plot each atlas with consistent color scale
for (atlas in all_atlases) {
    this_atlas_data <- subset(func_map_all_parc_df, Atlas == atlas)
    this_fill_title <- paste(func_name, "-", atlas, sep = " ")
    this_atlas_plot <- plot_subcortical_data(
        subcortex_data=this_atlas_data,
        atlas=atlas,
        value_column='value',
        fill_title=this_fill_title,
        hemisphere='both',
        vmin=min_func_parc_value,
        vmax=max_func_parc_value,
        cmap='plasma',
        show_legend=FALSE
    )
    print(this_atlas_plot)
}
```

Code snippet 21. Computing and plotting the region-averaged $GABA_A$-$\alpha 1$ densities in all twelve provided subcortical/cerebellar atlases [R].

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

# Get min and max values across all atlases for consistent color scaling
min_func_parc_value <- min(func_map_all_parc_df$value)
max_func_parc_value <- max(func_map_all_parc_df$value)

# Cortical plot with ggseg
cortex_p <- func_map_all_parc_df %>%
    filter(Atlas == 'Schaefer1000_17Networks') %>%
    rename(Region_Label = region) %>%
    mutate(label = ifelse(str_detect(Region_Label, "_LH_"),
                  paste0("lh_", Region_Label), paste0("rh_", Region_Label))) %>%
    ggplot() +
      geom_brain(atlas = schaefer17_1000(), mapping = aes(fill = value),
                 colour = "black", linewidth=0.1) +
    ggtitle("Schaefer-1000 Cortex") +
    labs(fill='Mean GABAa1 Signal (SUVR)') +
    scale_fill_viridis_c(limits=c(min_func_parc_value, max_func_parc_value),
                                na.value='gray80', option='plasma') +
    theme_void() +
    # 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 =
                        subset(func_map_all_parc_df, Atlas == "Melbourne_S4"),
                    atlas = "Melbourne_S4", hemisphere = "both",
                    value_column='value', vmin=min_func_parc_value,
                    vmax=max_func_parc_value, cmap='plasma', show_legend=F) +
                    ggtitle("Melbourne S4 Subcortex") +
                    theme(plot.title = element_text(hjust=0.5))

# SUIT cerebellum plot
cerebellum_p <- plot_subcortical_data(subcortex_data =
                        subset(func_map_all_parc_df, Atlas == "SUIT_cerebellar_lobule"),
                    atlas = "SUIT_cerebellar_lobule", hemisphere = "both",
                    value_column='value', vmin=min_func_parc_value,
                    vmax=max_func_parc_value, cmap='plasma', show_legend=F) +
                    ggtitle("SUIT Cerebellum") +
                    theme(plot.title = element_text(hjust=0.5))

# Use patchwork to combine the plots; titles need to be manually
# adjusted in Inkscape after saving the figure
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 22. Plotting GABA$_A$-$\alpha 1$ receptor density in the cortex (from Code snippet 19), subcortex, and cerebellum all together [R].

```bash
# This script calls the create_meshes_and_SVGs.py script for each atlas
repository_path=/path/to/cloned/subcortex_visualization/
cd $repository_path/adding_new_atlases/semi_automated_pipeline_code

# Set the name of the new atlas you want to process
atlas_name=name_of_new_atlas

# Copy the atlas in its original space to its own directory in atlas_info
atlas_data_path=${repository_path}/atlas_info/original_space/${atlas_name}
atlas_SVG_output_path=$repository_path/subcortex_visualization/data/${atlas_name}

# Set smoothing parameters for mesh creation
smooth_i=20 # Number of iterations for smoothing the meshe
smooth_f=0.9 # Smoothing factor for the meshes
cmap=viridis # Colormap to use for the meshes and SVGs

# Define paths to atlas data
atlas_nii=$atlas_data_path/$atlas_name/${atlas_name}.nii.gz
atlas_lookup_txt=$atlas_data_path/$atlas_name/${atlas_name}_lookup.txt

# Call python script
python automate_create_meshes_and_SVGs.py --atlas_name ${atlas_name} \
    --atlas_nii ${atlas_nii} \
    --atlas_lookup_txt ${atlas_lookup_txt} \
    --atlas_output_data_path ${atlas_SVG_output_path} \
    --smooth_i ${smooth_i} \
    --smooth_f ${smooth_f} \
    --cmap ${cmap}
```

Code snippet 23. The semi-automated mesh creation pipeline for a new atlas [bash].

```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 24. Generating a custom RGB colormap for the 3D mesh object [Python].

```bash
python volume_to_mesh_mz3.py \
--input_volume /path/to/input_volume.nii.gz \
--output_path /path/to/output/mz3s/ --out_file name_of_atlas.mz3 \
--index_max 8 --colors plasma_8_colors.txt --delete_mz3
```

Code snippet 25. Rendering the segmentation volume as an .mz3 mesh object for use in Surf Ice [bash].

Appendix II. Supplementary figures

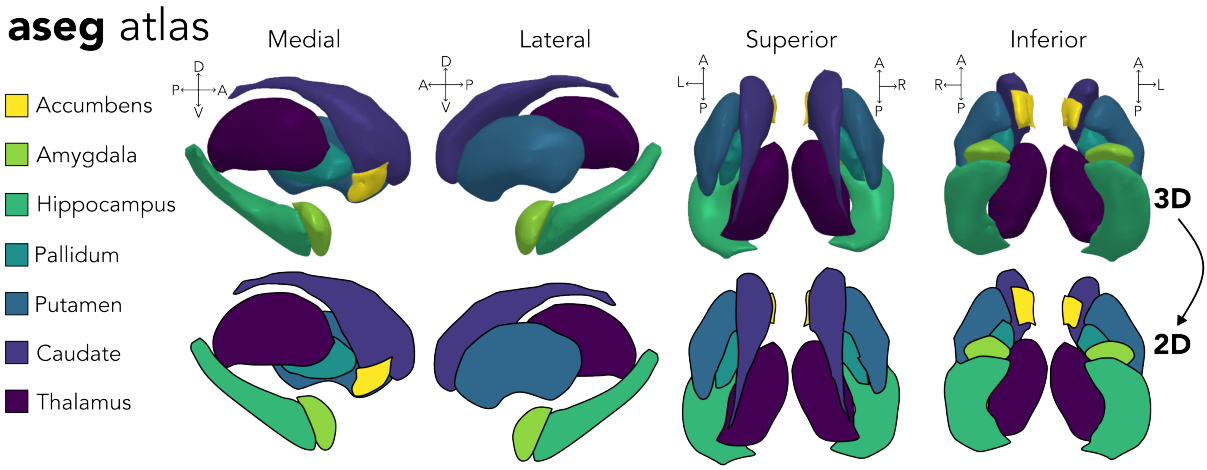

Figure S1. aseg subcortex atlas.

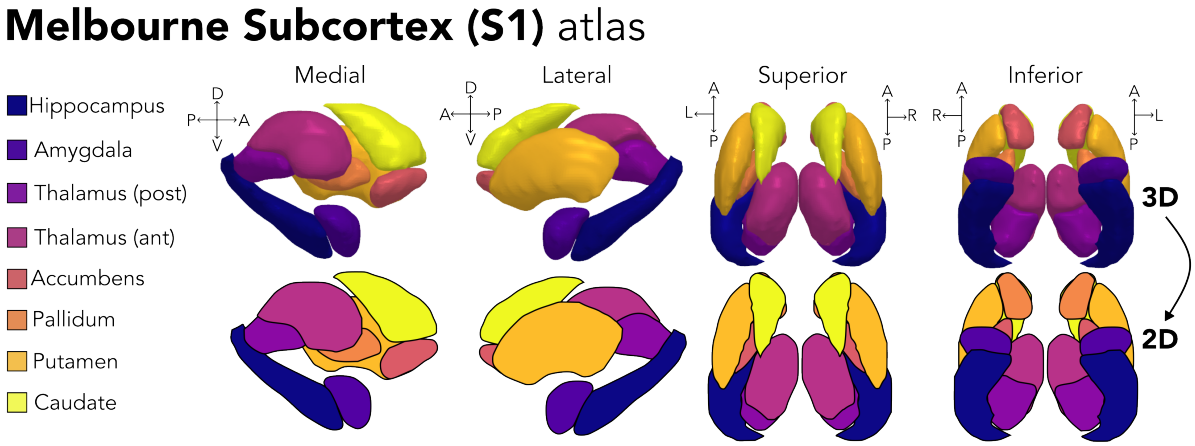

Figure S2. Melbourne (S1) subcortical atlas.

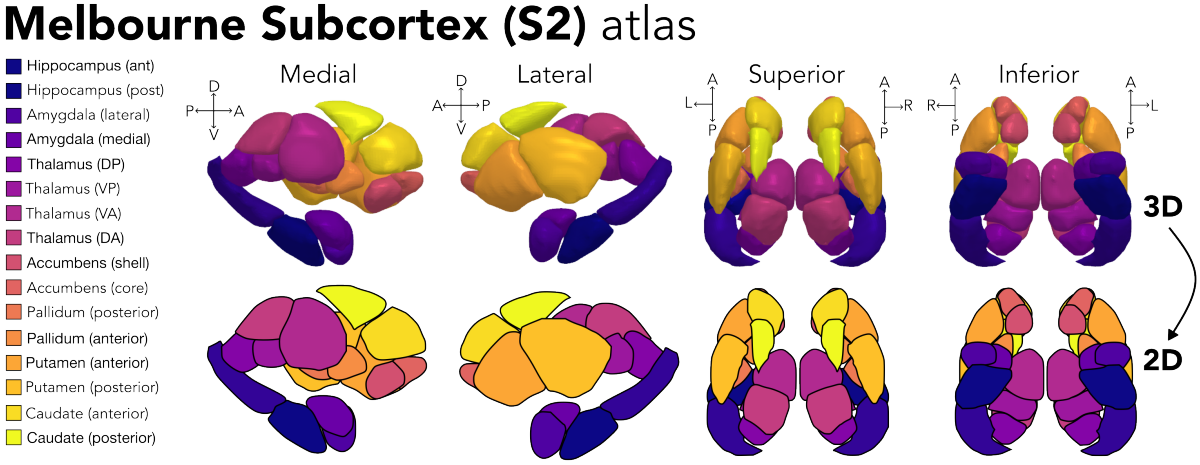

Figure S3. Melbourne (S2) subcortical atlas.

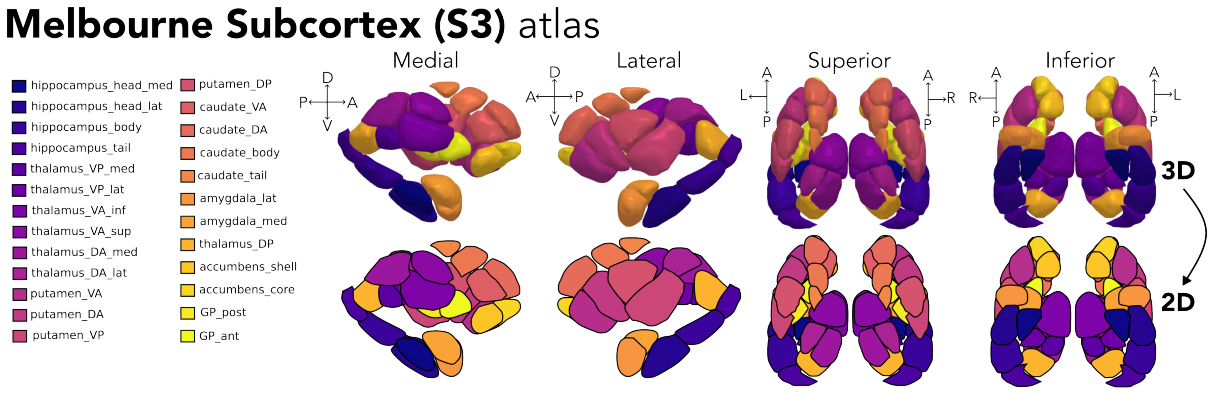

Figure S4. Melbourne (S3) subcortical atlas.

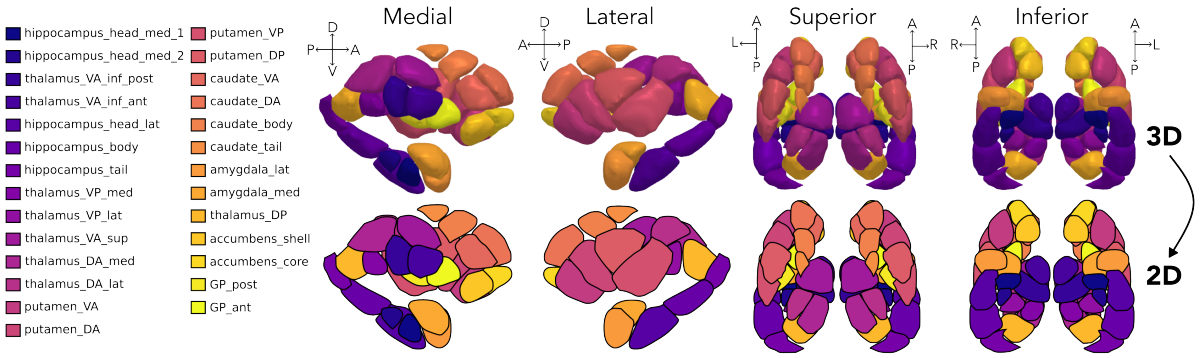

Figure S5. Melbourne (S4) subcortical atlas.

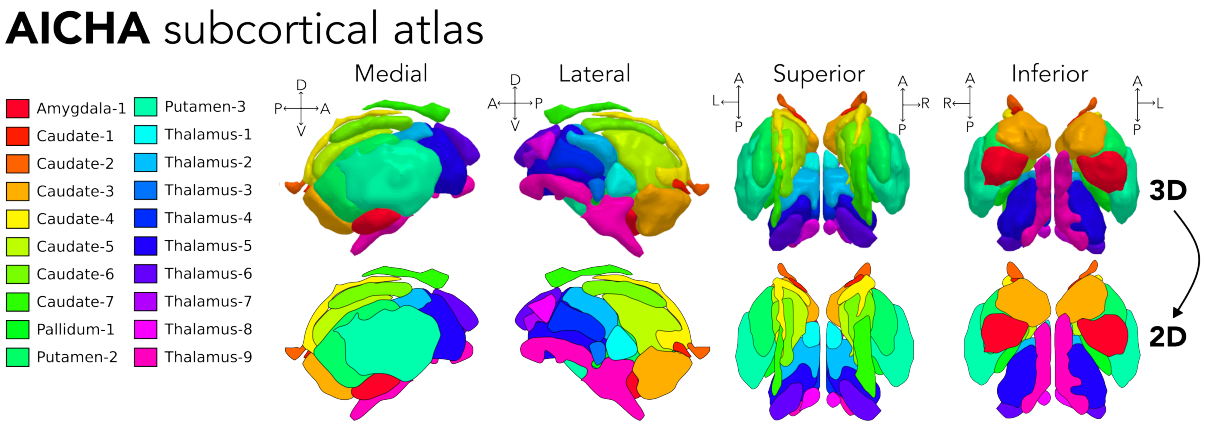

Figure S6. AICHA subcortical atlas.

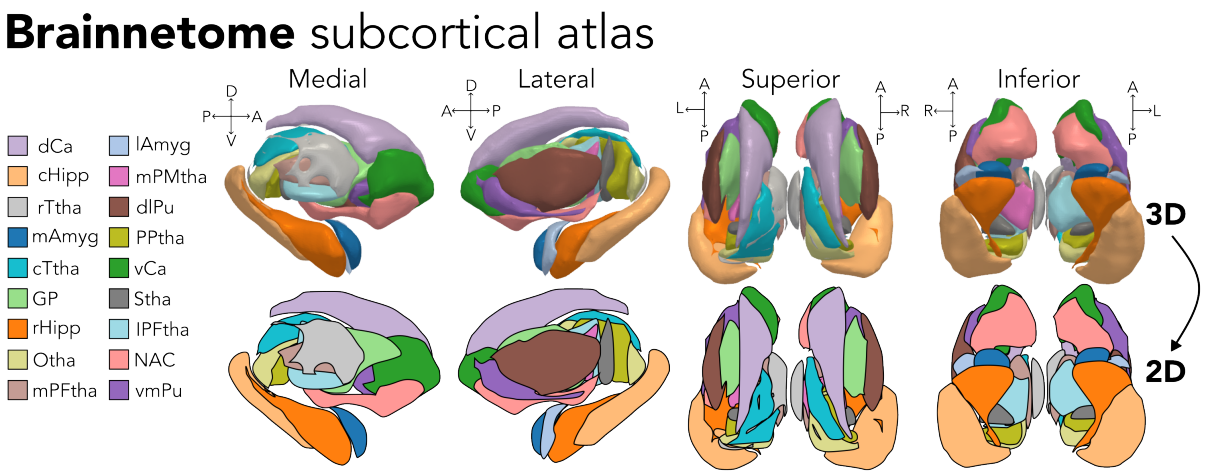

Figure S7. Brainnetome subcortical atlas.

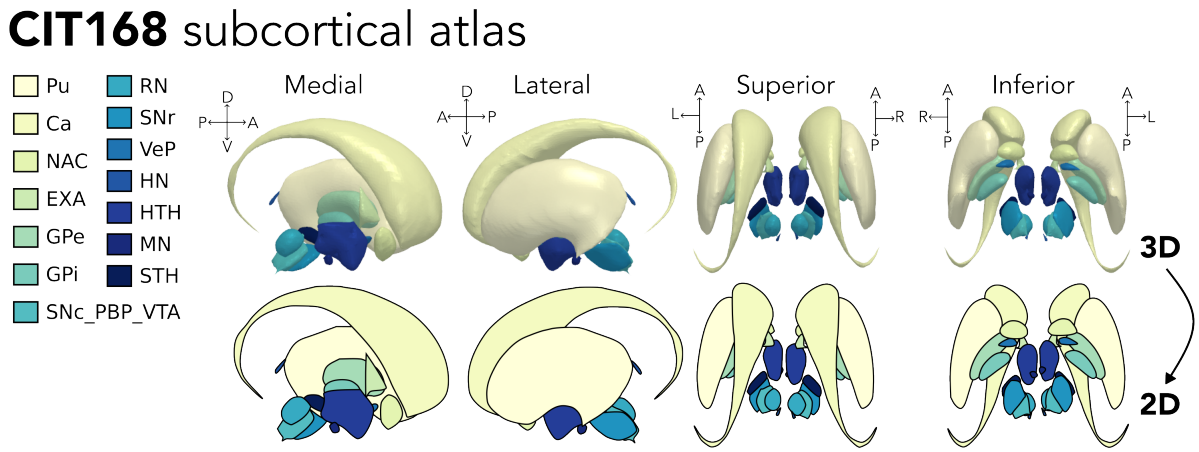

Figure S8. CIT168 subcortical atlas.

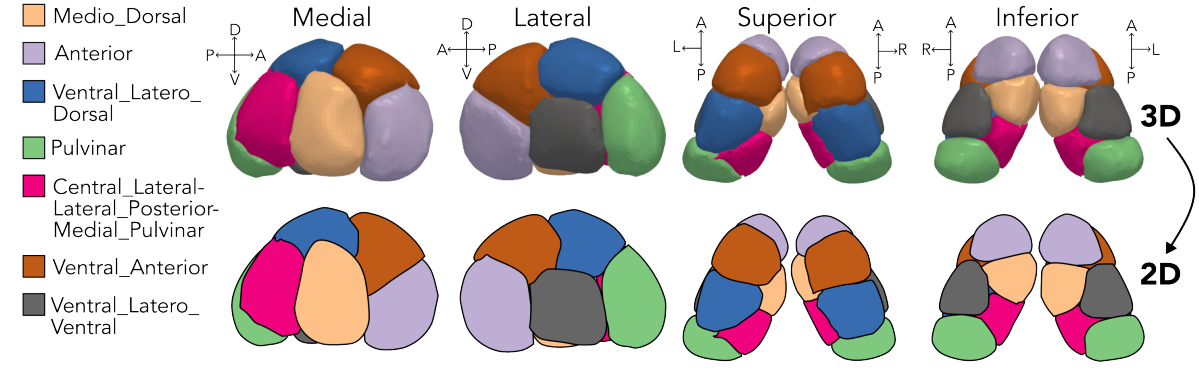

Figure S9. Thalamic nuclei (HCP) atlas.

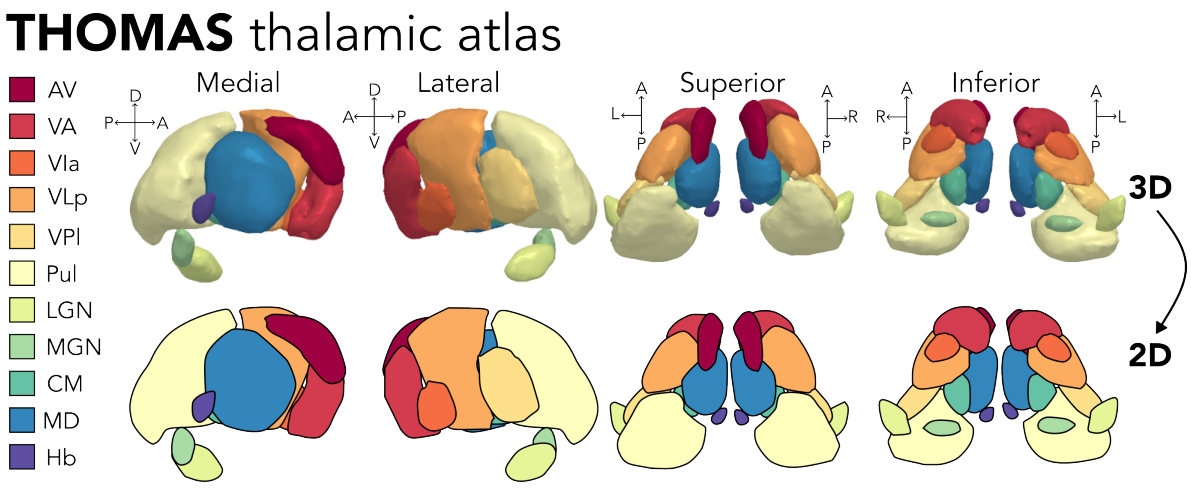

Figure S10. Thalamic nuclei (THOMAS) atlas.

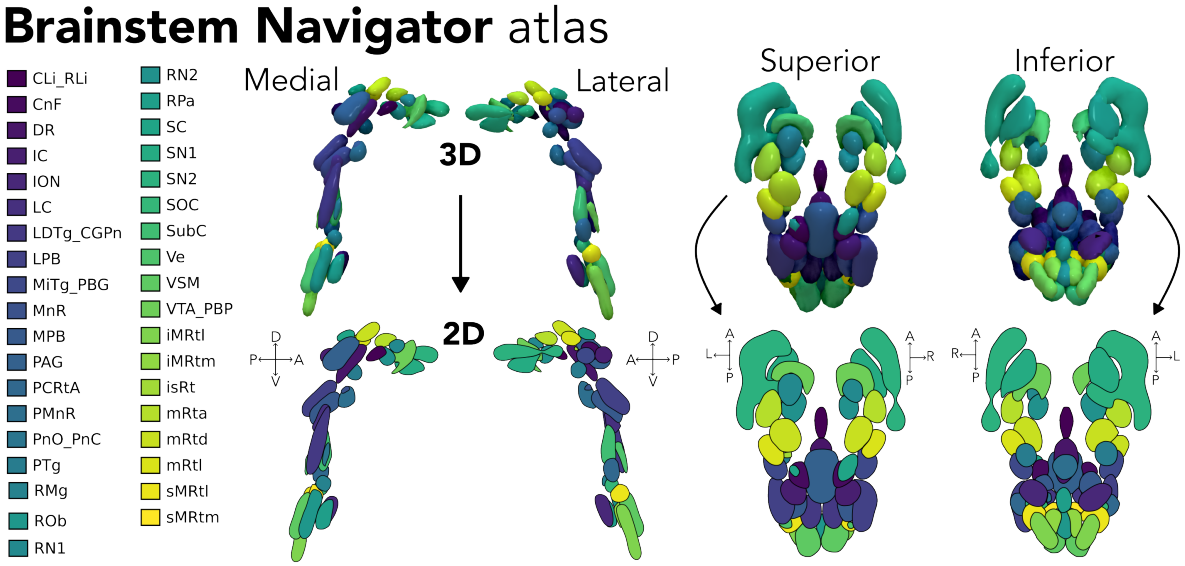

Figure S11. Brainstem Navigator atlas.

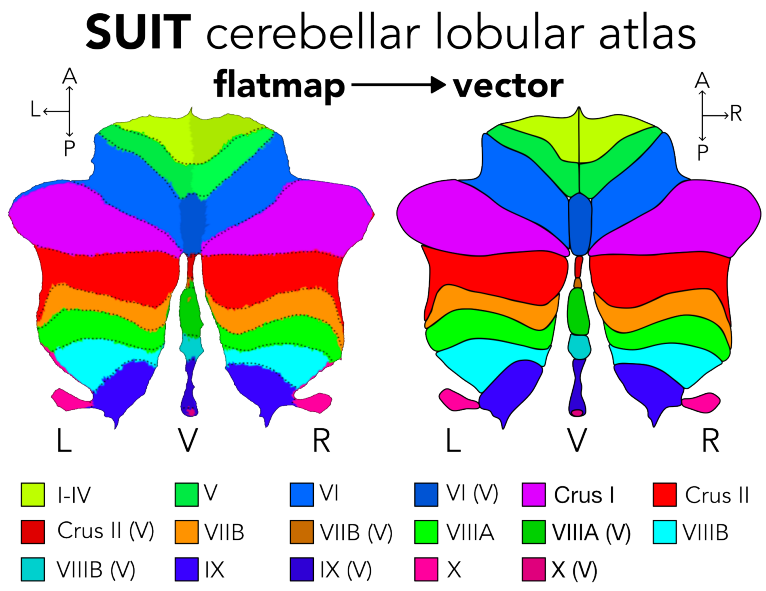

Figure S12. SUIT cerebellar lobular atlas.

Appendix III. Supplementary tables

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

Atlas	Regions
aseg subcortex	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 subcortex	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 subcortex	GP, NAC, Otha, PPtha, Stha, cHipp, cTtha, dCa, dlPu, lAmyg, lPFtha, mAmyg, mPFtha, mPMtha, rHipp, rTtha, vCa, vmPu
CIT168 subcortex	Pu, Ca, NAC, EXA, GPe, GPi, SNc_PBP_VTA, RN, SNr, VeP, HN, HTH, MN, STH
Thalamus Nuclei (HCP)	Anterior, Central_Lateral-Lateral_Posterior-Medial_Pulvinar, Medio_Dorsal, Pulvinar, Ventral_Anterior, Ventral_Latero_Dorsal, Ventral_Latero_Ventral
Thalamus (THOMAS)	AV, VA, VLa, VLp, VPl, Pul, LGN, MGN, CM, MD, Hb
Brainstem Navigator	CLi_RLi, CnF, DR, IC, ION, LC, LDTg_CGPn, LPB, MiTg_PBG, MnR, MPB, PAG, PCRtA, PMnR, PnO_PnC, PTg, RMg, RN1, RN2, ROb, RPa, SC, SN1, SN2, SOC, SubC, Ve, VSM, VTA_PBP, iMRtl, iMRtm, isRt, mRta, mRtd, mRtl, sMRtl, sMRtm
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

Table S2. Information about original MNI space and download access for each atlas in our toolbox. Links were last accessed on 1 May 2026.

Atlas	Template space	Download link
aseg_subcortex	MNI152NLin6Asym	https://github.com/ThomasYeoLab/CBIG/tree/master/data/templates/volume/FSL_MNI152_FS4.5.0/
Melbourne S1	MNI152NLin6Asym	https://github.com/yetianmed/subcortex/blob/master/Group-Parcellation/3T/Subcortex-Only/Tian_Subcortex_S1_3T_1mm.nii.gz
Melbourne S2	MNI152NLin6Asym	https://github.com/yetianmed/subcortex/blob/master/Group-Parcellation/3T/Subcortex-Only/Tian_Subcortex_S2_3T_1mm.nii.gz
Melbourne S3	MNI152NLin6Asym	https://github.com/yetianmed/subcortex/blob/master/Group-Parcellation/3T/Subcortex-Only/Tian_Subcortex_S3_3T_1mm.nii.gz
Melbourne S4	MNI152NLin6Asym	https://github.com/yetianmed/subcortex/blob/master/Group-Parcellation/3T/Subcortex-Only/Tian_Subcortex_S4_3T_1mm.nii.gz
AICHA	MNI152NLin6Asym	https://search.kg.ebrains.eu/instances/Dataset/b820fbed-d920-43c4-9b85-2e6be3d04cd1
Brainnetome	MNI152NLin6Asym	http://www.brainnetome.org/resource/201910/t20191030_522618.html
CIT168	MNI152NLin2009cAsym	https://osf.io/r2hvk/files/osfstorage
Thalamus (HCP)	MNI152NLin2009cSym	https://doi.org/10.5281/zenodo.1405484
Thalamus (THOMAS)	MNI152NLin2009bAsym	https://zenodo.org/records/5045684
Brainstem Navigator	MNI152NLin6Asym	https://www.nitrc.org/projects/brainstemnavig
SUIT cerebellar lobule	MNI152NLin2009cSym	https://github.com/DiedrichsenLab/cerebellar_atlases/blob/master/Diedrichsen_2009/atl-Anatom_space-MNISym_probseg.nii

Appendix IV. Supplementary Methods

10 Additional atlas information

10.1 aseg subcortex atlas

The aseg atlas was obtained as a volumetric segmentation in MNI152 space from the Computational Brain Imaging Group (CBIG; led by Professor Thomas Yeo), which is based on FreeSurfer's 'recon-all' pipeline applied to the FSL MNI152 1mm isotropic template and was shared in MNI152NLin6Asym space. The results of this pipeline are openly available at https://github.com/ThomasYeoLab/CBIG/tree/master/data/templates/volume/FSL_MNI152_FS4.5.0/. The aseg atlas originally contains both cortical and subcortical regions, so we filtered the atlas to include only the subcortical regions based on the following indices: 10, 11, 12, 13, 17, 18, 26, 49, 50, 51, 52, 53, 54, 58.

10.2 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 MNI152NLin6Asym space.

10.3 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 AICHA atlas was originally provided in MNI152NLin2009cAsym space. The subcortical indices range from 345-384, which we used to extract the subcortical regions from the full atlas volume.

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

10.4 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 Brainnetome atlas was originally provided in MNI152NLin6Asym space. 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.

10.5 CIT168 subcortex atlas

The CIT168 subcortex atlas was obtained from OSF using a shell script prepared by the Penn Lifespan Informatics and Neuroimaging Center (PennLINC) and shared via their GitHub repository at the following link: https://github.com/PennLINC/AtlasPack/blob/main/CIT168/prepare_atlas.sh. The shell script calls a Python script that splits the hemispheres of the original atlas volume such that each region is assigned a unique index across hemispheres (e.g., the left caudate and right caudate are assigned different indices). The CIT168 atlas was originally provided in MNI152NLin2009cAsym space.

10.6 Thalamus nuclei (HCP) atlas

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

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

10.7 Thalamus nuclei (THOMAS) atlas

The THOMAS thalamic nuclei atlas was obtained from the Zenodo repository that accompanies the original publication at https://zenodo.org/records/5045684. Please note that the Zenodo record is publicly available but the files are restricted and require users to request access to download the files directly. The THOMAS atlas was originally provided in MNI152NLin2009bAsym space.

10.8 Brainstem Navigator atlas

The Brainstem Navigator atlas was obtained as a directory of individual NIfTI files for each brainstem region from NITRC at https://www.nitrc.org/projects/brainstemnavigator. We selected the region of interest (ROI) files contained in the '2a.BrainstemNucleiAtlas_MNI/labels_thresholded_binary_0.35/' directory to use binary segmentation volumes of each brainstem nucleus. The Brainstem Navigator atlas regions were all shared in MNI152NLin6Asym space.

10.9 SUIT cerebellar lobule atlas

Our implementation of the SUIT cerebellar lobule atlas is based on the surface flatmap rendering shared by Diedrichsen and Zotow [54]. However, we also include a volumetric segmentation of the SUIT cerebellar lobules in our toolbox, which was obtained from the Diedrichsen Lab GitHub repository at the following link: https://github.com/DiedrichsenLab/cerebellar_atlases. The SUIT volumetric atlas was originally provided in MNI152NLin2009cSym space.

11 Further technical details about atlas alignment to MNI152NLin6Asym and MNI152NLin2009cAsym spaces

As shown in Table S2, the twelve atlases included in our toolbox are originally provided in either MNI152NLin6Asym, MNI152NLin2009cAsym, MNI152NLin2009cSym, or MNI152NLin2009bAsym space. Since the MNI152NLin6Asym and MNI152NLin2009cAsym spaces are the most commonly used template spaces in our toolbox, we aligned the atlases that were originally provided in other spaces to both of these template spaces. We opted to use the widely adopted `templateflow` framework to perform the atlas alignments, which provides pre-computed transformations between commonly used template spaces. For template pairs that did not already have pre-computed transformations available in TemplateFlow (e.g., MNI152NLin2009bAsym to MNI152NLin2009cAsym), we performed registration using the `registration` function from the ANTS toolbox in Python [81], using the `type_of_transform='SyN'` function to specify symmetric normalization. In either case, we applied the corresponding transform file to each atlas as needed using the `apply_transforms` function from ANTS, specifying nearest-neighbor interpolation with the `interpolator='nearestNeighbor'` to preserve discrete region indices.

First, we provide an example of the Python code we used to transform an atlas from MNI152NLin2009cAsym to MNI152NLin6Asym space, for which TemplateFlow provides a pre-computed transformation. The code snippet is shown in Listing 26.

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

space_from = "MNI152NLin2009cAsym"
space_to = "MNI152NLin6Asym"

# Get the .xfm file for space_from to space_to transformation from TemplateFlow
tflow.get(
        space_to,
        suffix="xfm",
        desc=None,
        **{"from": space_from}
)

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

# Define xfm
xfm = (
    "path/to/templateflow/" # Change to your local templateflow path
    "tpl-" + space_to + "/"
    "tpl-" + space_to + "_from-" + space_from + "_mode-image_xfm.h5"
)

# Load in atlas volume image using ANTs and the reference image for the target space
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 26. Example of how to align a segmentation atlas from MNI152NLin2009cAsym to MNI152NLin6Asym space [Python].

Alternatively, we show an example of how to perform registration using ANTS when a pre-computed transformation is not available in TemplateFlow. In this example, we align an atlas from MNI152NLin2009bAsym to MNI152NLin2009cAsym space. The code snippet is shown in Listing 27.

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

# Step 1. Perform registration using ANTs to get the forward and inverse transform files
input_ants = ants.image_read(os.fspath(tflow.get(input_template, resolution=1,
                                                 suffix='T1w', desc=None)))
output_ants = ants.image_read(os.fspath(tflow.get(output_template, resolution=1,
                                                  suffix='T1w', desc=None)))

# Set an output path
output_path = "path/to/output_directory"  # Change to your desired output path

reg = ants.registration(
    fixed=output_ants,
    moving=input_ants,
    type_of_transform='SyN',
    outprefix=f'{output_path}/tpl-{output_template}_from-{input_template}'
)

# reg['fwdtransforms'] contains 2 files: a warp field (.nii.gz) and affine (.mat)
# Copy them to a permanent location with BIDS-style naming
shutil.copy(reg['fwdtransforms'][0],
            f'{output_path}/tpl-{output_template}_from-{input_template}_mode-image_xfm.nii.gz')
shutil.copy(reg['fwdtransforms'][1],
            f'{output_path}/tpl-{output_template}_from-{input_template}_mode-image_xfm.mat')

# Save inverse too (Asym --> Sym)
shutil.copy(reg['invtransforms'][0],
            f'{output_path}/tpl-{input_template}_from-{output_template}_mode-image_xfm.nii.gz')
shutil.copy(reg['invtransforms'][1],
            f'{output_path}/tpl-{input_template}_from-{output_template}_mode-image_xfm.mat')

# Transform file to use with TemplateFlow
transform_file = f'{output_path}/tpl-{output_template}_from-{input_template}_mode-image_xfm.mat'

# Step 2. Apply the transform to the atlas volume using TemplateFlow + ANTs
ref_img = os.fspath(tflow.get(
    input_space,
    resolution=1,
    desc=None,
    suffix="T1w"
))

# Load in images using ANTs
atlas_img = ants.image_read(input_atlas_file)
reference = ants.image_read(ref_img)

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

# Save out the warped atlas
ants.image_write(atlas_warped, f'{output_path}/warped_atlas.nii.gz')
```

Code snippet 27. Example of how to align a segmentation atlas from MNI152NLin2009bAsym to MNI152NLin2009cAsym space [Python].

12 Further information about mesh-to-vector conversion

For the semi-automated mesh-to-vector conversion process, we used the following smoothing parameters with the `yabplot.build_subcortical_atlas` function:

Table S3. Mesh smoother parameters supplied to `yabplot.build_subcortical_atlas` for each atlas are shown here. The `smooth_i` parameter specifies the number of iterations of smoothing to perform, while the `smooth_f` parameter specifies the smoothing factor (between 0 and 1) that controls the amount of smoothing applied at each iteration.

Atlas name	`smooth_i`	`smooth_f`
aseg subcortex	20	0.9
Melbourne (S1)	15	0.6
Melbourne (S2)	15	0.6
Melbourne (S3)	15	0.6
Melbourne (S4)	15	0.6
AICHA subcortex	5	0.6
Brainnetome subcortex	15	0.8
CIT168 subcortex	12	0.8
Thalamus (HCP)	10	0.6
Thalamus (Thomas)	10	0.6