CCSegThickness User Guide

Chris Adamson

June 14, 2015

1 Introduction

The Corpus Callosum (CC) is the largest white matter structure of the human brain and it facilitates interhemispheric cortico-cortical communication. The CC is a collection of fibre bundles that converges into a contiguous mass at the midsagittal plane (MSP) and diverges outwards to the cerebral cortex. Anatomically, the midsagittal slice has been defined as the plane that intersects the following features: the falx cerebri, the cerebral aqueduct, the pineal stalk, the peaked roof of the 4th ventricle, and minimal grey matter in the interhemispheric fissure [1].

This software provides end-to-end pipeline for group-wise studies of MSP CC morphology (thickness profiles and areal parcellations) using a collection of whole-brain T_1-weighted magnetic resonance (MR) images. The pipeline consists of several stages as follows:

1. For each subject's T_1-weighted MR image:
    (a) Midsagittal plane extraction
    (b) CC segmentation (manual inspection/editing tool provided)
    (c) Thickness profile generation
    (d) Areal parcellation
    (e) Projection of segmentation and parcellations to native space
2. Groupwise thickness profile statistical testing with results display

Firstly, the midsagittal plane extraction step performs a rigid body alignment so that the MSP is the middle sagittal slice, which is extracted as a 2D image for CC segmentation. The CC segmentation uses a cascade of atlas- and data-driven techniques. A graphical tool is provided for manual insepction and editing of segmentations. The thickness profile generation step produces a collection of cross-sectional traversals, streamlines, of the CC from inferior to superior boundary in a posterior to anterior trajectory; the arc lengths of these streamlines form a thickness profile. Areal parcellations are computed using Witelson [2] and Hofer and Frahm [3], as well as one that combines these two and includes the rostrum and genu, which we call the Emsell parcellation.

Group-wise statistical comparison functionality is included. Statistical tests include: One-sample Student's and two-sample Welch's T-test. Multiple comparison comparison correction is performed by permutation-based step-down max$T$ p-value adjustment algorithm of [4] and False Discovery Rate to control the Family Wise Error Rate and the False Discovery Rate, respectively. The user may choose any combination of these techniques.

The software is described in detail in the following publications:

- C. Adamson, R. Beare, M. Walterfang, and M. Seal, "Software Pipeline for Midsagittal Corpus Callosum Thickness Profile Processing". Neuroinformatics. 2014;12(4):595-614.
- C. Adamson, A. Wood, J. Chen, S. Barton, D. Reutens, C. Pantelis, et al. "Thickness profile generation for the corpus callosum using Laplace's equation." Human Brain Mapping. 2011;32(12):2131-40.

2 Prerequisites

The software requires Python 2, which is installed by default in most Linux distributions. The following dependencies are required to be installed:

- numpy
- scipy
- h5py
- nibabel
- python-opencv
- pylab
- matplotlib
- skimage
- cython

`--single-subject=<foo>`	Only the specified subject will be processed
`--subjects-include=<comma separated list>`	Comma separated list of subjects for processing
`--subjects-include-file=<file>`	Newline delimited file containing list of subjects for processing
`--subjects-exclude=<comma separated list>`	Comma separated list of subjects to exclude
`--subjects-exclude-file=<file>`	Newline delimited file containing list of subjects to exclude

Table 1: Subject selection options

The following command will install all of the above dependencies in Ubuntu:

- `apt-get install python-numpy python-scipy python-nibabel python-opencv python-pylab python-matplotlib python` `cython`

Users may optionally install `scikits.sparse` `https://pythonhosted.org/scikits.sparse/cholmod.html` to speed up the thickness profile generation component by allowing it to use Cholesky decomposition rather than LU decomposition to solve the Laplace equations.

The software requires `FSL` `http://www.fmrib.ox.ac.uk/fsl/` to be installed and configured. For Debian/Ubuntu installations there are packages available in NeuroDebian `http://neuro.debian.net/`.

2.1 Other tools

The `acpcdetect` tool from ART `http://www.nitrc.org/projects/art/` is used but is included within the installation package and is not required to be externally installed.

3 Installation

Extract the files from the downloaded archive into an empty directory. Optionally, run `make_cython.sh` from a terminal to rebuild the streamline generation module.

4 Worked analysis example

A typical workflow consists of several steps:

- Preparing your data
- Preprocessing of each subject
  - Midsagittal plane extraction
  - CC segmentation with optional manual editing
  - Thickness profile generation and areal parcellation
  - Backprojection of segmentation and parcellations to native space
- Groupwise thickness profile statistical testing with results display

4.1 Preparing your data

Place all T_1-weighted images into a single directory, for example `RawData`, in NIFTI format. The files may optionally be compressed with `gzip`. Create an empty directory, for example `cc_seg`, for output files.

4.2 Preprocessing

The `CCSegProcess` tool is used as a single driver for performing midsagittal plane extraction, CC segmentation, thickness profile generation, and backprojection of results to native space; see 7.1 for a detailed synopsis.

The basic usage is:

```
$  CCSegProcess [options] <input directory> <output directory>
```

The processing steps are executed by specifiying processing directives:

`--do-midsag`	MSP extraction
`--do-seg`	CC segmentation
`--do-thickness`	Thickness profile generation and areal parcellation
`--do-segtonative`	Project segmentation and parcellations to native space

The option `--do-all`, performs all of the operations, i.e. is a synonym for: `--do-midsag --do-seg --do-thickness --do-segtonat`
The user may specify multiple do options.

`CCSegProcess` is a batch processing tool which, by default, processes all subjects in `<input directory>`. The user may customise the subjects for processing using the modifiers in Table 1.

We will now demonstrate a full preprocessing workflow, consider that the subject `foo` is being processed.

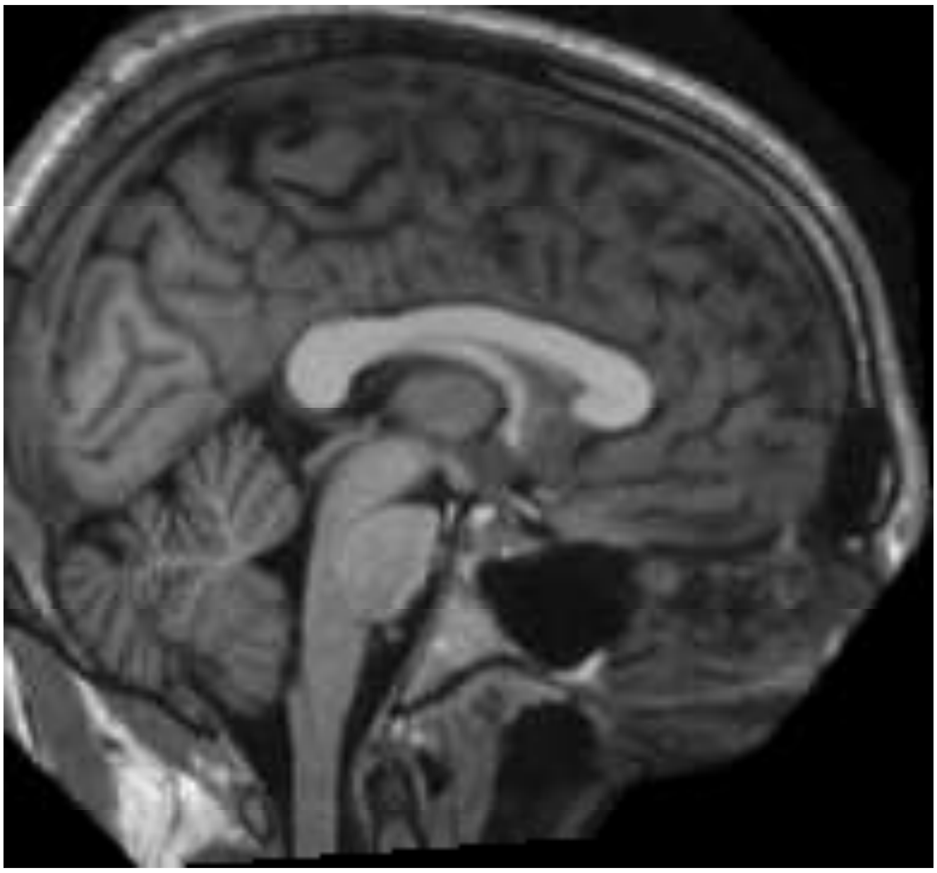


Figure 1: Example midsagittal plane image.

4.2.1 Midsagittal plane extraction

To perform midsagittal plane extraction, the basic usage is:

```
$  CCSegProcess --do-midsag [options] <input directory> <output directory>
```

Output files produced

<output directory>/foo_midsag.hdf5: HDF5 file with the following fields

- **MSPMethod (string):** the method used to perform midsagittal alignment
- **NIIPixdims (vector):** voxel dimensions of the input image
- **flirtCropZerosCols (vector):**
- **flirtCropZerosRows (vector):**
- **flirtMAT** (4×4)**:** affine transformation between input and MSP aligned images
- **flirtTemplateFile (string):** the template image used for MSP alignment, if `--msp-method=flirt` then this will be the MNI152 standard brain, if `--msp-method=acpcdetect` then this will be the MSP aligned image
- **midSagAVW (image):** the extracted MSP image
- **originalNativeFile (string):** the location of the original image
- **originalNativeCroppedFile (string):** the location of the initially cropped image
- **originalOrientationString (string):** the slice orientations of the original image

Graphics files produced

<output directory>/midsag/foo_midsag.png: The extracted midsagittal plane image, see Figure 1.

4.2.2 CC segmentation

To perform CC segmentation, the basic usage is:

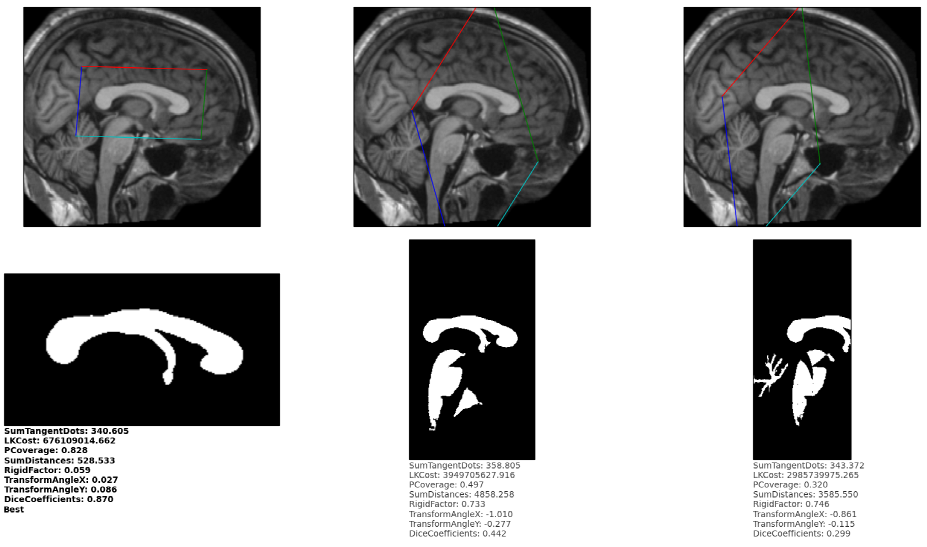
```
$  CCSegProcess --do-seg [options] <input directory> <output directory>
```

Output files produced

<output directory>/foo_seg.hdf5: HDF5 file with the segmentation images.

Graphics files produced

<output directory>/seg/foo_lk.png: Output from initial template CC to original image registration. All initialisation points are shown along with goodness-of-fit measures. See Figure 2.



```
SumTangentDots: 340.605
LKCost: 676109014.662
PCoverage: 0.828
SumDistances: 528.533
RigidFactor: 0.059
TransformAngleX: 0.027
TransformAngleY: 0.086
DiceCoefficients: 0.870
Best
```

```
SumTangentDots: 358.805
LKCost: 3949705627.916
PCoverage: 0.497
SumDistances: 4858.258
RigidFactor: 0.733
TransformAngleX: -1.010
TransformAngleY: -0.277
DiceCoefficients: 0.442
```

```
SumTangentDots: 343.372
LKCost: 2985739975.265
PCoverage: 0.320
SumDistances: 3585.550
RigidFactor: 0.746
TransformAngleX: -0.861
TransformAngleY: -0.115
DiceCoefficients: 0.299
```

Figure 2: Example output from Lucas-Kanade tracking. Each column shows output from a single initialisation. Top row shows midsagittal images with the border of the template after registration with the LK tracker. The bottom row shows the goodness-of-fit metric values. Bolded metrics show the best. `<output directory>/seg/foo_lk.png`.

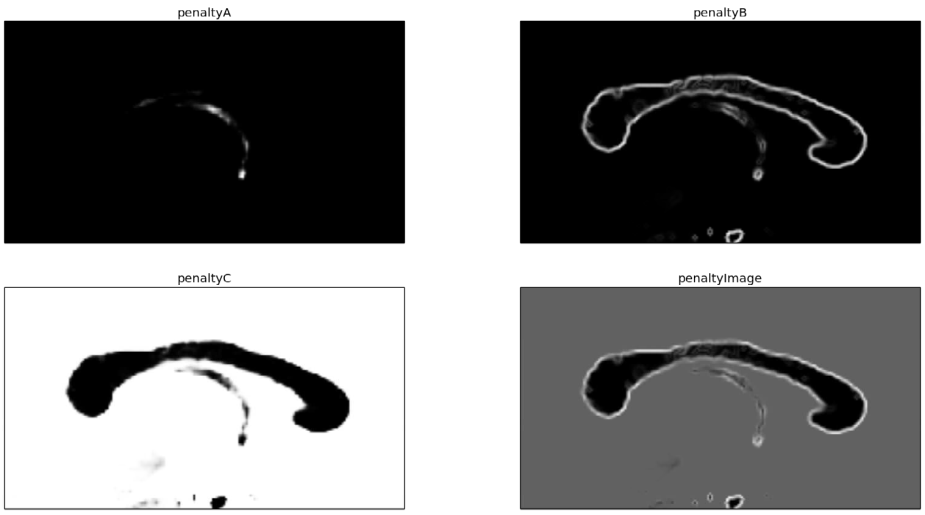


Figure 3: Penalty images. `PenaltyA` is higher for pixels likely to be in the fornix. `PenaltyB` is high were the image gradient is high. `PenaltyC` is high for intensities that are unlikely to belong to the CC. `PenaltyImage` is the final penalty image used for reinitialisation, it should be low in the real CC and higher otherwise. `<output directory>/seg/foo_penalty.png`.

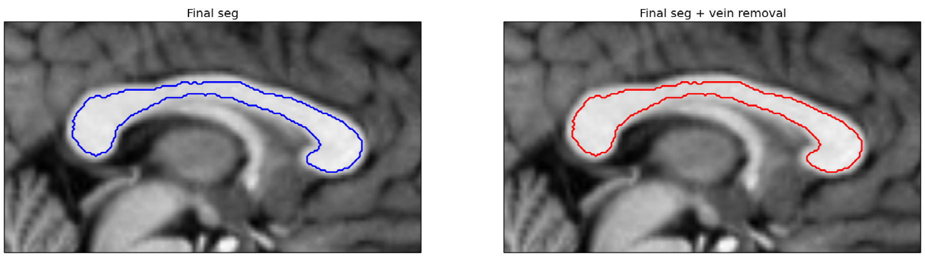


Figure 4: Final segmentations. The left panel shows the final segmentation prior to vein removal, the right panel shows the final segmentation after vein removal. The right panel is the segmentation used for thickness profile generation `<output directory>/seg/foo_seg.png`.

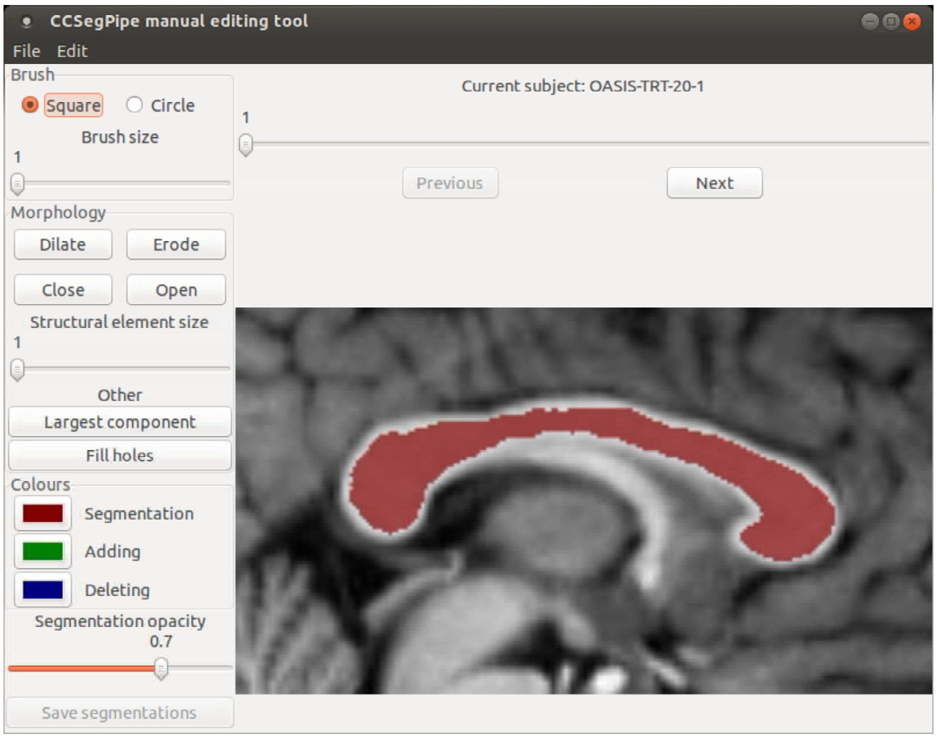


Figure 5: Manual editor screenshot.

4.2.3 CC segmentation manual editing

`CCSegManedit` is a tool for inspection and manual editing of segmentations. To run it, the command is

```
$ CCSegManedit <output directory>
```

Here, the `<output directory>` is the directory specified to `CCSegProcess` and contains all the segmentation hdf5 files. The tool automatically loads all segmentation files in the output directory for editing. The interface is shown in Figure 5.

Brush: Shape and size of of the brush.

Top scrollbar: Cycles through each subject, the text indicates the currently displayed subject. Subjects are ordered alphabetically.

Morphology: Mathematical morphology operations on the segmentation based on a circular structuring element of size specified by the "Structural element size" slider. The following operations are available:

Dilate: Expands the segmentation.

Erode: Contracts the segmentation.

Close: Dilation followed by erosion. Joins the segmentation over small gaps and acts as a boundary smoothing mechanism.

Open: Erosion followed by dilation. Splits the segmentation at thin connection points.

Other: Other useful operations:

Largest component: Retains only the largest connected component of the segmentation.

Fill holes: Fill completely enclosed holes in the segmentation.

Colours: Colours of the segmentation and pixels that are being added or deleted. Clicking on the colour boxes allows users to change the colours.

Segmentation opacity: How opaque the segmentation appears, 0 is completely transparent, 1 is completely opaque.

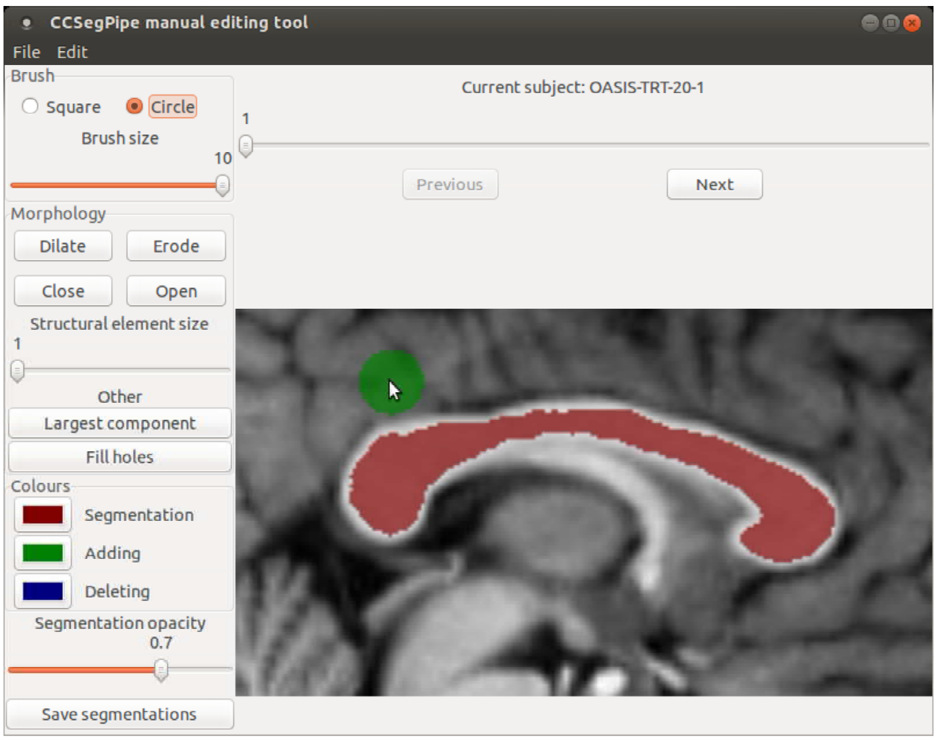
Save segmentations: Save all modified segmentations. All segmentations that have been modified from their original will be saved as `<output directory>/foo_seg_manedit.hdf5`.

Keyboard and mouse actions

Left click: Add to the segmentation. Pixels that will be added are tracked and highlighted by the "adding" colour, see Figure 6.

Right click: Delete from the segmentation. Pixels that will be deleted are tracked and highlighted by the "deleting" colour, see Figure 7.

Ctrl-Z: Undo the last edit.



Current subject: OASIS-TRT-20-1

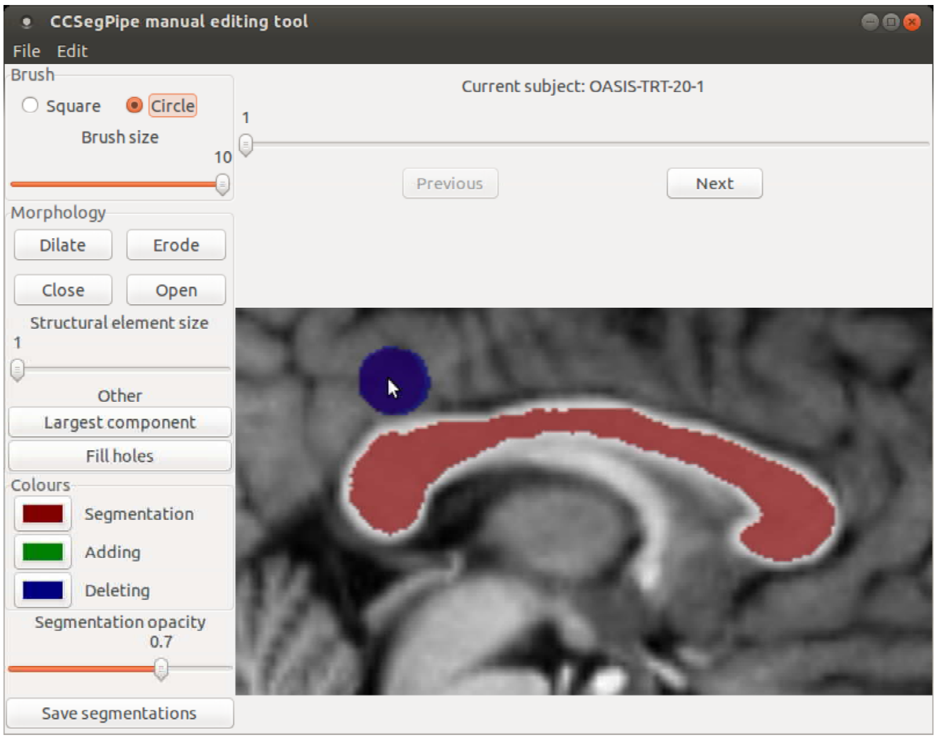
Brush: Square [✓ Circle]

Brush size 10

Structural element size 1

Segmentation opacity 0.7

Figure 6: Manual editor screenshot. Left mouse button is clicked, pixels marked will be added to the segmentation.



Current subject: OASIS-TRT-20-1

Brush: Square [✓ Circle]

Brush size 10

Structural element size 1

Segmentation opacity 0.7

Figure 7: Manual editor screenshot. Right mouse button is clicked, pixels marked will be deleted from the segmentation.

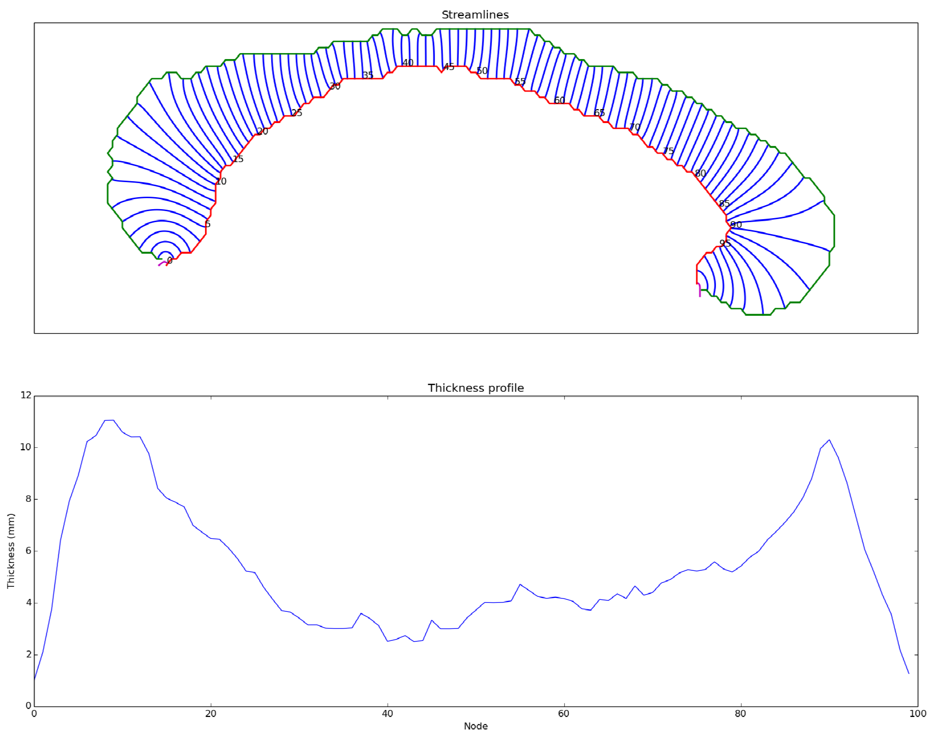


Figure 8: Top panel - inferior (red) and superior (green) boundary contours with cross-sectional streamlines (blue) with node numbers marked. Bottom panel - Thickness profile, x-axis corresponds to node numbers in top panel.

4.2.4 Thickness profile generation

To perform thickness profile generation, the basic usage is:

```
$ CCSegProcess --do-thickness [options] <input directory> <output directory>
```

If manual edits have been performed, use the following switch to only process segmentations that have manual edits:

```
$ CCSegProcess --do-thicknessmanedit [options] <input directory> <output directory>
```

Output files produced

<output directory>/foo_thickness.hdf5: HDF5 file with the thickness profiles and parcellations.

Graphics files produced

<output directory>/seg/foo_thickness.png: Top panel - inferior (red) and superior (green) boundary contours with cross-sectional streamlines (blue) with node numbers marked. Bottom panel - Thickness profile, x-axis corresponds to node numbers in top panel.

<output directory>/parcellations/foo_parcellations.png: Top panel - Witelson parcellation of the segmentation, Middle panel - Hofer and Frahm parcellation of the segmentation, Bottom panel - "Emsell" pacellation of the segmentation.

4.3 Projection of segmentations into native space

To place the segmentations and parcellations back into native space, the basic usage is:

```
$ CCSegProcess --do-segtonative [options] <input directory> <output directory>
```

Options

–dilate-parasag: Dilate the segmentation to occupy the left and right parasagittal slices in the segmentation space prior to projection back to native space. This makes the segmentation "thicker."

Output files produced

<output directory>/foo_seg_native.nii.gz: Segmentation in native space.

<output directory>/foo_parcellation_witelson_native.nii.gz: Witelson parcellation in native space.

<output directory>/foo_parcellation_hoferfrahm_native.nii.gz: Hofer and Frahm parcellation in native space.

<output directory>/foo_parcellation_emsell_native.nii.gz: "Emsell" parcellation in native space.

Temporary output files

- <output directory>/foo_native_cropped.nii.gz
- <output directory>/foo_native_cropped_to_template.mat

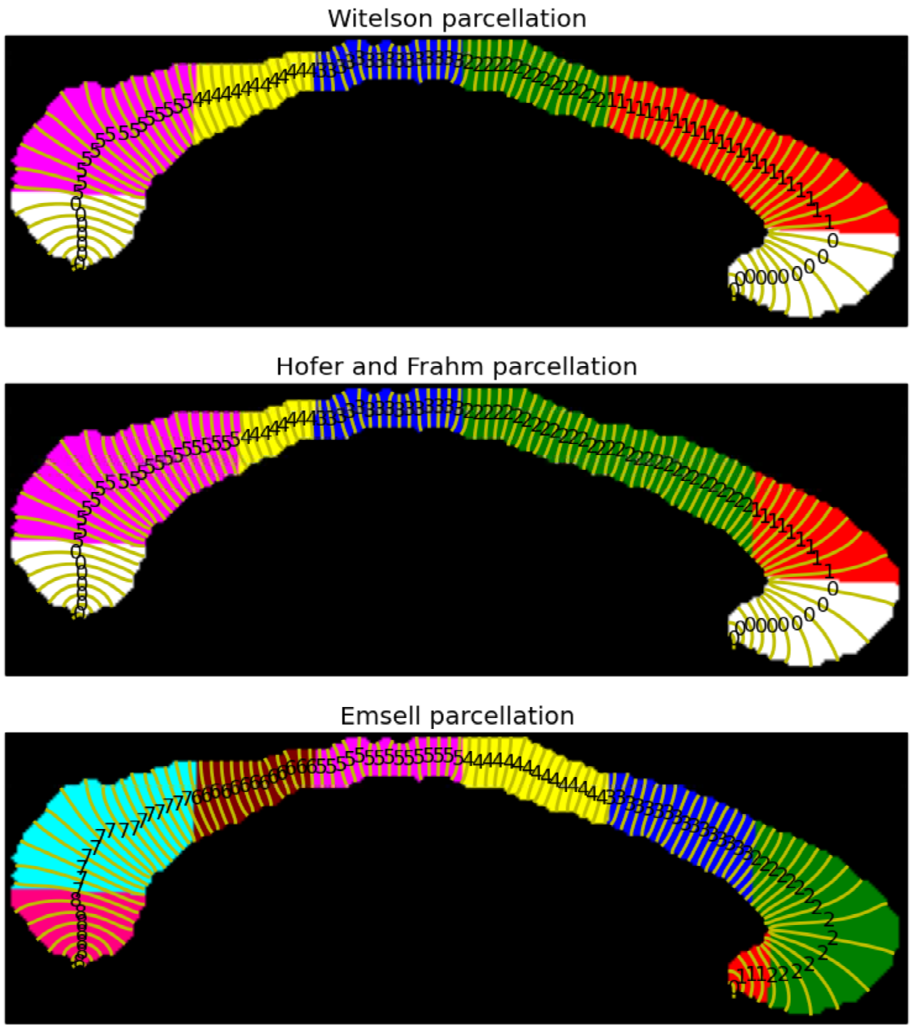


Figure 9: Top panel - Witelson parcellation of the segmentation, Middle panel - Hofer and Frahm parcellation of the segmentation, Bottom panel - “Emsell” pacellation of the segmentation.

- <output directory>/foo_native_cropped_to_template.nii.gz
- <output directory>/foo_parcellation_emsell_native_axial_cropped.nii.gz
- <output directory>/foo_parcellation_emsell_native_axial.nii.gz
- <output directory>/foo_parcellation_hoferfrahm_native_axial_cropped.nii.gz
- <output directory>/foo_parcellation_hoferfrahm_native_axial.nii.gz
- <output directory>/foo_parcellation_witelson_native_axial_cropped.nii.gz
- <output directory>/foo_parcellation_witelson_native_axial.nii.gz
- <output directory>/foo_seg_native_axial_cropped.nii.gz
- <output directory>/foo_seg_native_axial.nii.gz
- <output directory>/foo_template.nii.gz

4.4 Statistical testing

Mass univariate, nodewise, groupwise statistical testing can be performed with the tool `CCSegStatTest`. The basic usage is:

```
$  CCSegStatTest [options] <output directory> <test type> <group label file> <output hdf5 file>
```

Main arguments

`<output directory>`: the directory containing the thickness hdf5 files, should be the same as the `<output directory>` option used for `CCSegProcess`.

`<test type>`: the statistical test type to perform, supported values are:

2sample Welch's 2-sample T-test

paired Student's 1-sample paired T-test, tests for non-zero mean of

`<group label file>`: a two-column, tab delimited text file containing the group labels for each subject; the required format depends on the test type

2sample: Each line contains the subject id and its corresponding group label: <group label> <subject id>. the file must contain two unique group labels, <group label 1> and <group label 2>, T-test will be conducted on difference between <group 1> and <group 2>.

paired: The first line contains two group labels <group label 1> <group label 2>, then each line contains subject id pairs: <subject id 1> <subject id 2>. T-test will be performed on non-zero mean of <group 1> - <group 2>.

`<output hdf5 file>`: Results of the test, contains the following fields:

groupLabels: Group labels found in the group file.

observedT: Node wise T statistic values.

observedP: Node wise p values.

observedPFDR: Node wise p-values, corrected for False Discovery Rate.

permP: Node wise p-values, based on step-down $maxT$ Family Wise Error correction [4]

omnibusP: Omnibus p-value for group difference.

OPTIONS

`--num-perms`: The number of permutations for the permutation testing, 100000 by default.

`--cull-percent`: The meaning of thickness at the genu and splenium is unclear, this parameter dictates the number of nodes ignored from the posterior and anterior edges, 10 by default.

Examples of performing a 2-sample and 1-sample paired test will now be given.

4.4.1 Example 2-sample T-test

Group label file for (`grouplabels.txt`) two groups: controls (CTL), and patients (PAT):

```
CTL OASIS-TRT-20-1
CTL OASIS-TRT-20-10
CTL OASIS-TRT-20-11
CTL OASIS-TRT-20-12
CTL OASIS-TRT-20-13
CTL OASIS-TRT-20-14
CTL OASIS-TRT-20-15
CTL OASIS-TRT-20-16
CTL OASIS-TRT-20-17
CTL OASIS-TRT-20-18
PAT OASIS-TRT-20-19
PAT OASIS-TRT-20-2
PAT OASIS-TRT-20-20
PAT OASIS-TRT-20-3
PAT OASIS-TRT-20-4
PAT OASIS-TRT-20-5
PAT OASIS-TRT-20-6
PAT OASIS-TRT-20-7
PAT OASIS-TRT-20-8
PAT OASIS-TRT-20-9
```

```
   CCSegStatTest command:
   $  CCSegStatTest cc_seg 2sample grouplabels.txt 2sample_out.hdf5
```

4.4.2 Example 1-sample T-test

Group label file for (`grouplabels.txt`) two groups: controls (CTL), and patients (PAT):

```
CTL PAT
OASIS-TRT-20-1 OASIS-TRT-20-10
OASIS-TRT-20-11 OASIS-TRT-20-12
OASIS-TRT-20-13 OASIS-TRT-20-14
OASIS-TRT-20-15 OASIS-TRT-20-16
OASIS-TRT-20-17 OASIS-TRT-20-18
OASIS-TRT-20-19 OASIS-TRT-20-2
OASIS-TRT-20-20 OASIS-TRT-20-3
OASIS-TRT-20-4  OASIS-TRT-20-5
OASIS-TRT-20-6  OASIS-TRT-20-7
OASIS-TRT-20-8  OASIS-TRT-20-9
```

```
   CCSegStatTest command:
   $  CCSegStatTest cc_seg paired grouplabels.txt paired_out.hdf5
```

5 Statistical test results display

Currently, there is only a MATLAB script for performing visualisation of statistical testing results. The files are in the `matlab` subdirectory of the installation directory. To use it add the path of that directory to your MATLAB path.

The results of the statistical testing can be displayed with the cc_seg_stattest_display_results function:

```
>> help cc_seg_stattest_display_results
  cc_seg_stattest_display_results(OutputFile, PType, param1, val1, param2, val2, ...)

  DESCRIPTION
  Displays the results of groupwise statistical testing from CCSegStatTest
  PARAMETERS
  OutputFile (string): hdf5 file where results of CCSegStatTest were saved, the following
  fields are required
  permP: permutation test p-values
  observedT: T statistics from observed distribution
  observedP: p-values from observed distribution
  observedPFDR: FDR corrected p-values from observed distribution
  PType (string): determines the type of p-values to display, the
  following types are supported:
  'observed': the observed p-values
  'FDR': the FDR-corrected p-values
  'permutation': the permutation-test p-values
  OPTIONAL KEY/VALUE PAIR PARAMETERS
  'GroupLabels' (cell array of strings) [2]: a two-element cell array
  with the labels of the groups. {'Group 1', 'Group 2'} by default
'OutputFile' (string): the figure will be saved to this file, format
will be a png file
```

The OutputFile parameter should point to the HDF5 file created by `CCSegStatTest`. The PType refers to the type of p-values to display; supported values are:

- '**observed**' – the observed p-values
- '**FDR**' – the FDR-corrected p-values
- '**permutation** ' – the permutation-test p-values

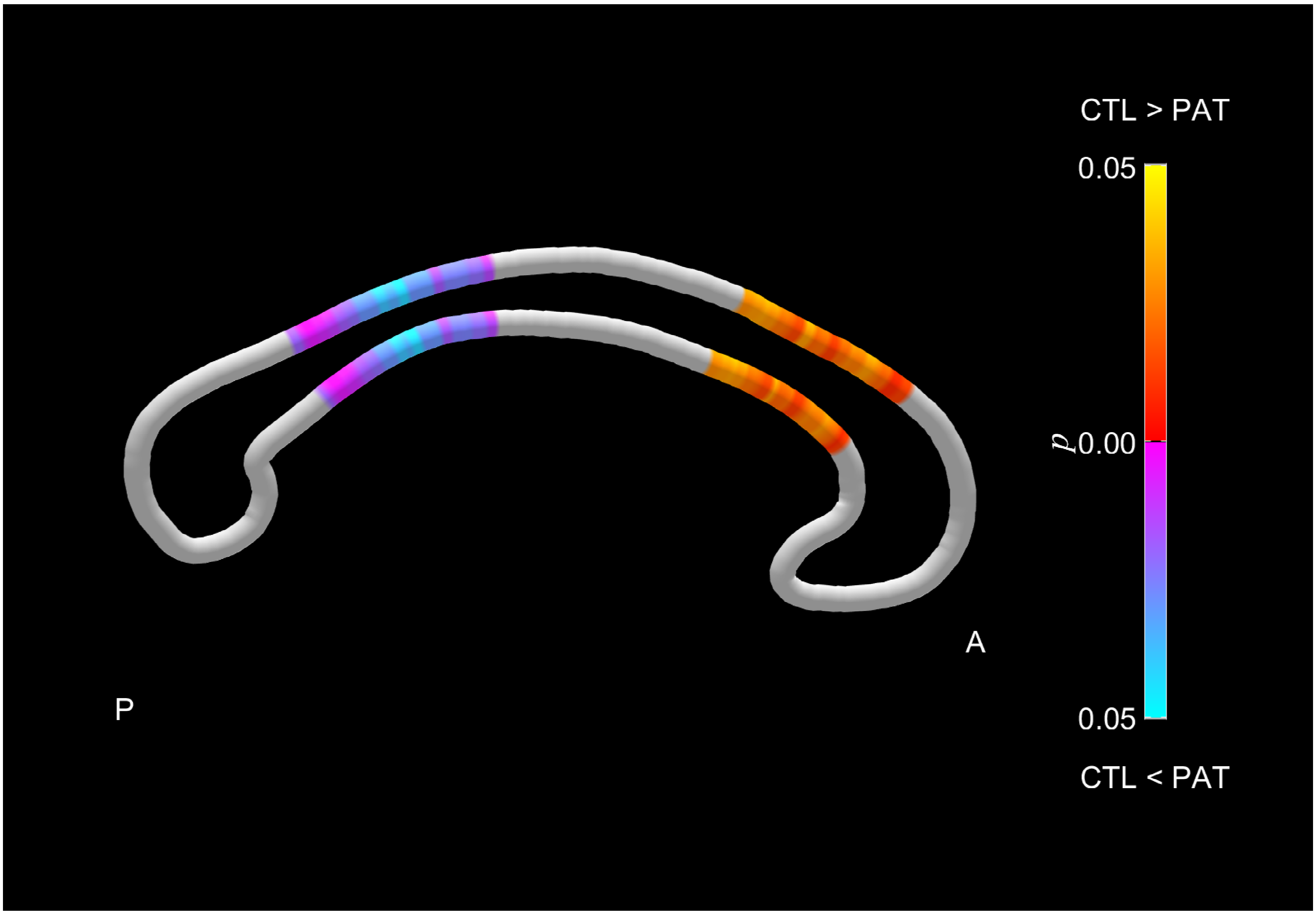


Figure 10: Statistical results display example.

The optional parameters are:

- GroupLabels – the legend has group labels that indicate the direction of the differences for significant *p*-values. This parameter specifies these labels. The parameter is {'Group 1', 'Group 2'} by default.

An examplary call to the display function is:

```
>> cc_seg_stattest_display_results('cc_seg_stats.hdf5', 'perm', 'GroupLabels', {'CTL', 'PAT'});
```

where '**cc_seg_stats.hdf5**' was the file containing the output of `CCSegStatTest` and {'CTL', 'PAT'} were the group labels. Figure 10 shows an example output for a fictitious analysis. The colour scheme for significant *p*-values indicates the direction of the difference between groups.

6 Other tools

6.1 CCSegThicknessToCSV

`CCSegThicknessToCSV` exports thickness profiles and stats for the parcellation schemes to CSV files.

```
Usage: ./CCSegThicknessToCSV [options] <input directory> <output directory>

Prints thickness profiles to <output directory>/all_thickness_profiles.csv
Each row is the subject followed by the nodal thicknesses

Prints stats of the thickness profiles according to the parcellations in <output directory>/parcellation_stats/
For each parcellation scheme: witelson, hoferfrahm, emsell, the following files are produced:
<output directory>/parcellation_stats/<scheme>_stats_area.csv: the areas of each region
Within the following files, the stats refer to the thicknesses of the nodes within each region
<output directory>/parcellation_stats/<scheme>_stats_max.csv: the maximum thickness within each region
<output directory>/parcellation_stats/<scheme>_stats_min.csv: the minimum thickness within each region
<output directory>/parcellation_stats/<scheme>_stats_median.csv: the median thickness within each region
<output directory>/parcellation_stats/<scheme>_stats_std.csv: the standard deviation of the thicknesses within ea
<output directory>/parcellation_stats/<scheme>_stats_var.csv: the variance of the thicknesses within each region
```

This command supports the subject selection options given in Table 1.

7 Command references

7.1 `CCSegProcess`

SYNOPSIS

```
$  CCSegProcess [options] <input directory> <output directory>
```

OPTIONS

Processing directives (`--do-*` options)	
Command line switch	Description
`--do-midsag`	MSP extraction
`--do-seg`	CC segmentation
`--do-thickness`	Thickness profile generation and areal parcellation
`--do-segtonative`	Project segmentation and parcellations to native space
Other processing directives	
`--do-thicknessmanedit`	Perform thickness profile generation and areal parcellation only if a manually edited segmentation is found
`--do-all`	Do all steps, synonym for: `--do-midsag` `--do-seg` `--do-thickness` `--do-segtonative`
`--redo-done`	Redo analysis if output file is found, this option is off by default
`--save-graphics`	Creates graphics files in subdirectories of the output directory that shows the output of each step
`--do-midsag` options	
`--msp-method=`	Tool used for midsagittal plane alignment; supported methods are: `--msp-method=acpcdetect` for ART's *acpcdetect* (default) `--msp-method=flirt` for FSL's *FLIRT*
`--do-seg` options	
`--seg-no-lk`	Disable Lucas-Kanade affine registration, use if no outputs are correct
`--do-thickness` options	
`--num-thickness-nodes=`	Number of thickness profiles to generate, 100 by default
`--do-segtonative` options	
`--dilate-parasag`	Dilate the the segmentation and parcellation masks to the left and right parasagittal slices prior to backprojection to native space

References

[1] T. N. Mitchell, S. L. Free, M. Merschhemke, L. Lemieux, S. M. Sisodiya, and S. D. Shorvon, "Reliable callosal measurement: population normative data confirm sex-related differences," *American Journal of Neuroradiology*, vol. 24, no. 3, pp. 410–418, 2003.

[2] S. F. Witelson, "Hand and sex differences in the isthmus and genu of the human corpus callosum: A postmortem morphological study," *Brain*, vol. 112, no. 3, pp. 799–835, 1989.

[3] S. Hofer and J. Frahm, "Topography of the human corpus callosum revisited – comprehensive fiber tractography using diffusion tensor magnetic resonance imaging," *NeuroImage*, vol. 32, no. 3, pp. 989–994, 2006.

[4] P. H. Westfall and S. S. Young, *Resampling-Based Multiple Testing: Examples and Methods for p-Value Adjustment (Wiley Series in Probability and Statistics)*, 1st ed. Wiley-Interscience, Jan. 1993.