

Tissue-Class Segmentation

Overall Pipeline

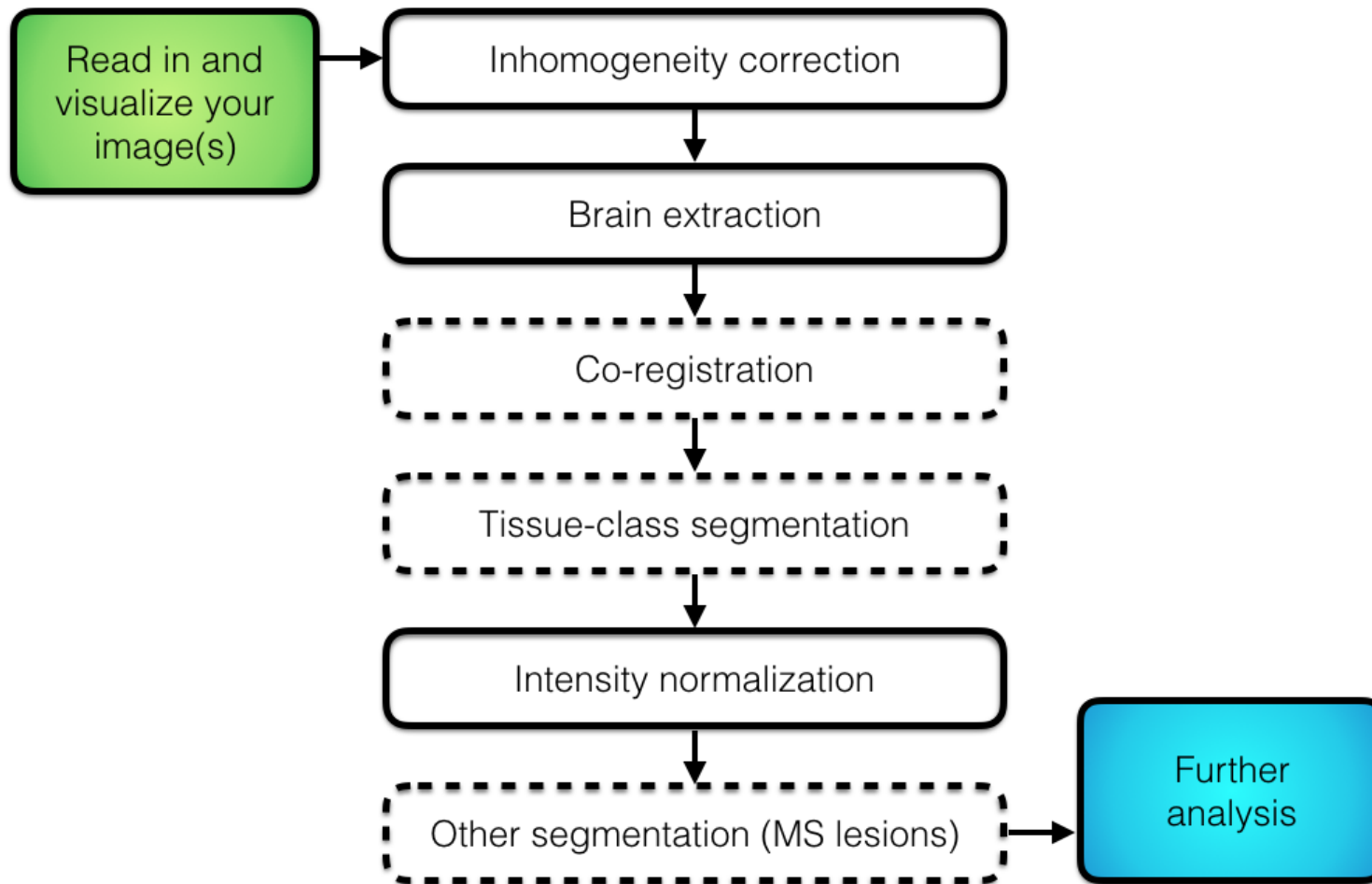


Image segmentation

- We are often interested in subdividing or segmenting the brain into meaningful biological regions of interest (ROIs) for an analysis.
- Examples: tissue segmentation, segmentation of deep gray matter structures, segmentation of pathology (MS lesions, tumors, ...)
- We will perform 3-class tissue segmentation in R using `fslr` and `ANTsR`:
 - Cerebrospinal fluid (CSF)
 - Gray Matter (GM)
 - White Matter (WM)

Loading Data

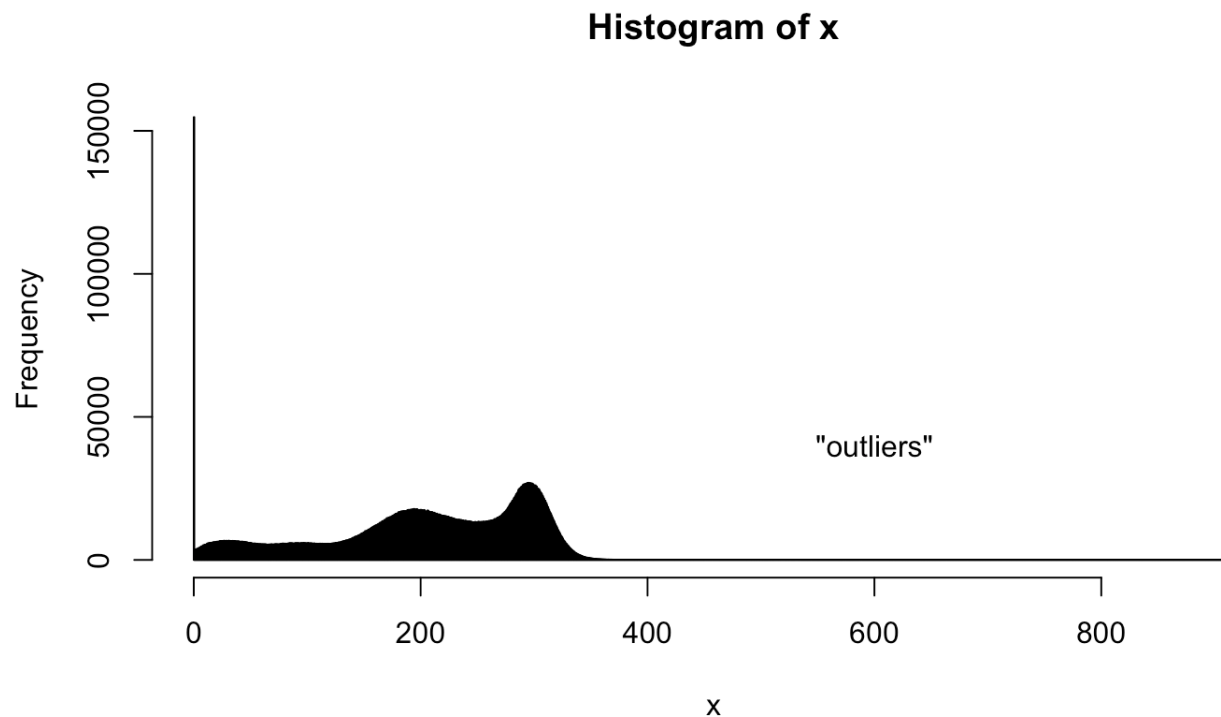
- Let's read in the training T1 and brain mask for training subject 02.

```
library(ms.lesion)
library(neurobase)
all_files = get_image_filenames_list_by_subject(
  group = "training",
  type = "coregistered")
files = all_files$training02
t1 = readnii(files["T1"])
mask = readnii(files["Brain_Mask"])
```

Tissue Segmentation: Large Outliers

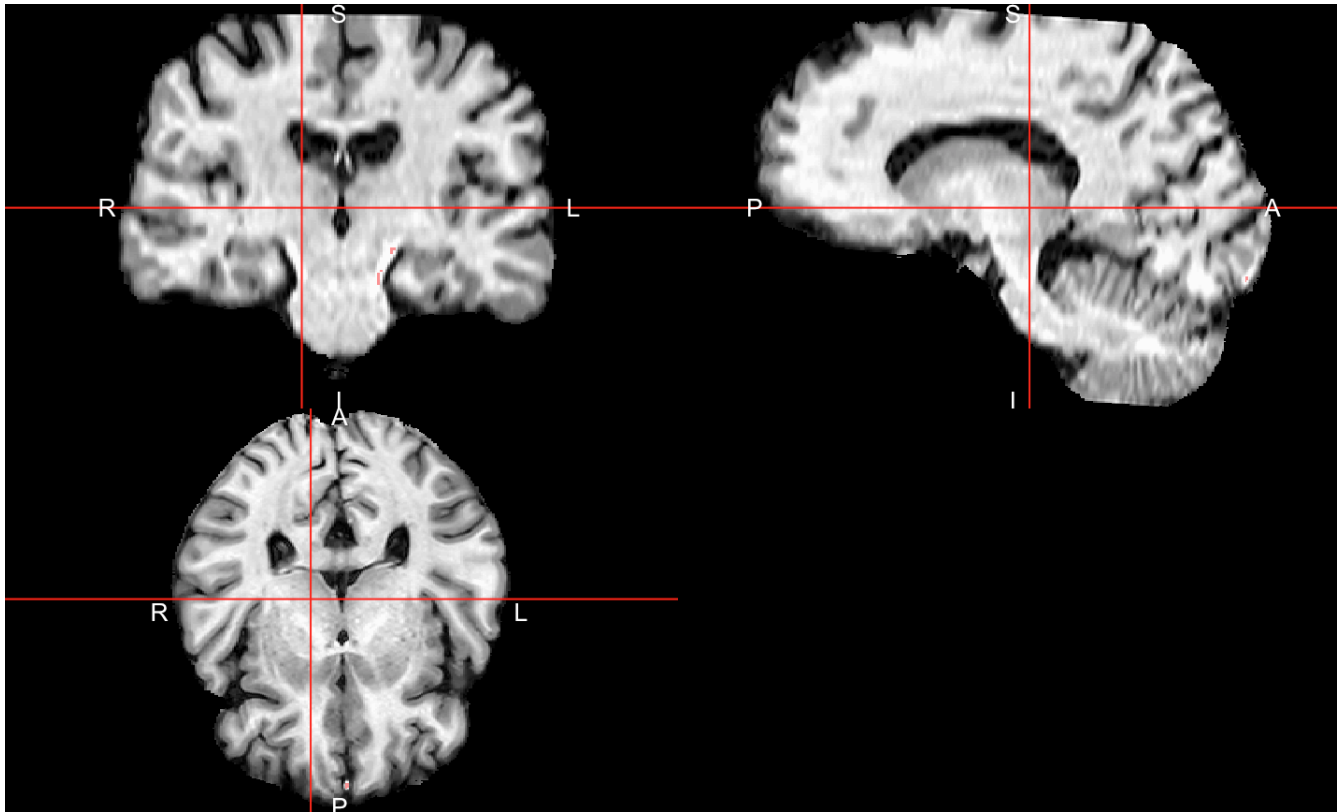
- Many tissue class segmentations are based on k-means clustering.
- These methods can be skewed by large outliers.

```
hist(t1, mask = mask, breaks = 2000); text(x = 600, y = 40000, '"outliers"')
```



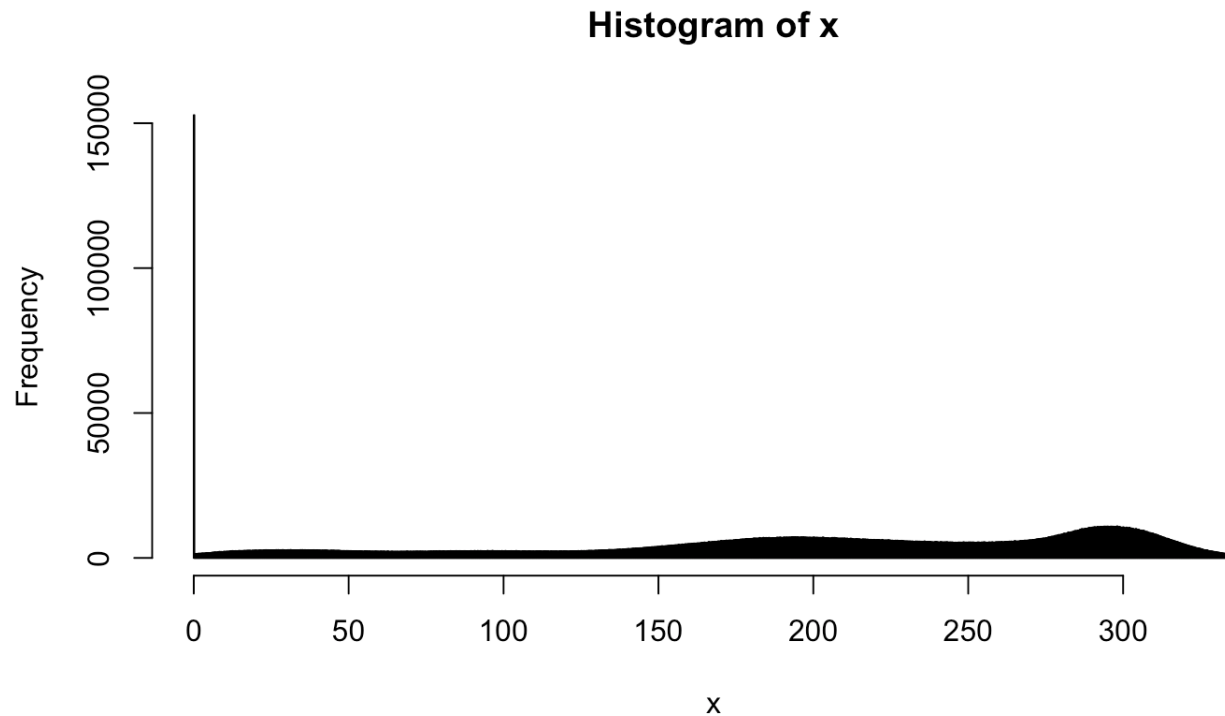
Where are the outliers?

```
ortho2(rt1, t1 > 400, col.y = alpha("red", 0.5)) # xyz - cog of a region
```



What does the histogram look like after truncating outliers?

```
rt1 = robust_window(t1)  
hist(rt1, mask = mask, breaks = 2000);
```



Tissue Segmentation using FSL FAST

- FAST is based on a hidden Markov random field model and an Expectation-Maximization algorithm (Zhang, Brady, and Smith 2001).
- It jointly produces a bias field corrected image and a probabilistic tissue segmentation.
- More robust to noise and outliers than finite mixture model-based methods that do not incorporate spatial information.

The `fslr` function `fast` calls `fast` from FSL. The `--nobias` option tells FSL to not perform inhomogeneity correction (N4 already performed in `ANTsR`).

```
t1fast = fast(t1,  
             outfile = paste0(nii.stub(t1file), "_FAST"),  
             opts = "--nobias")
```

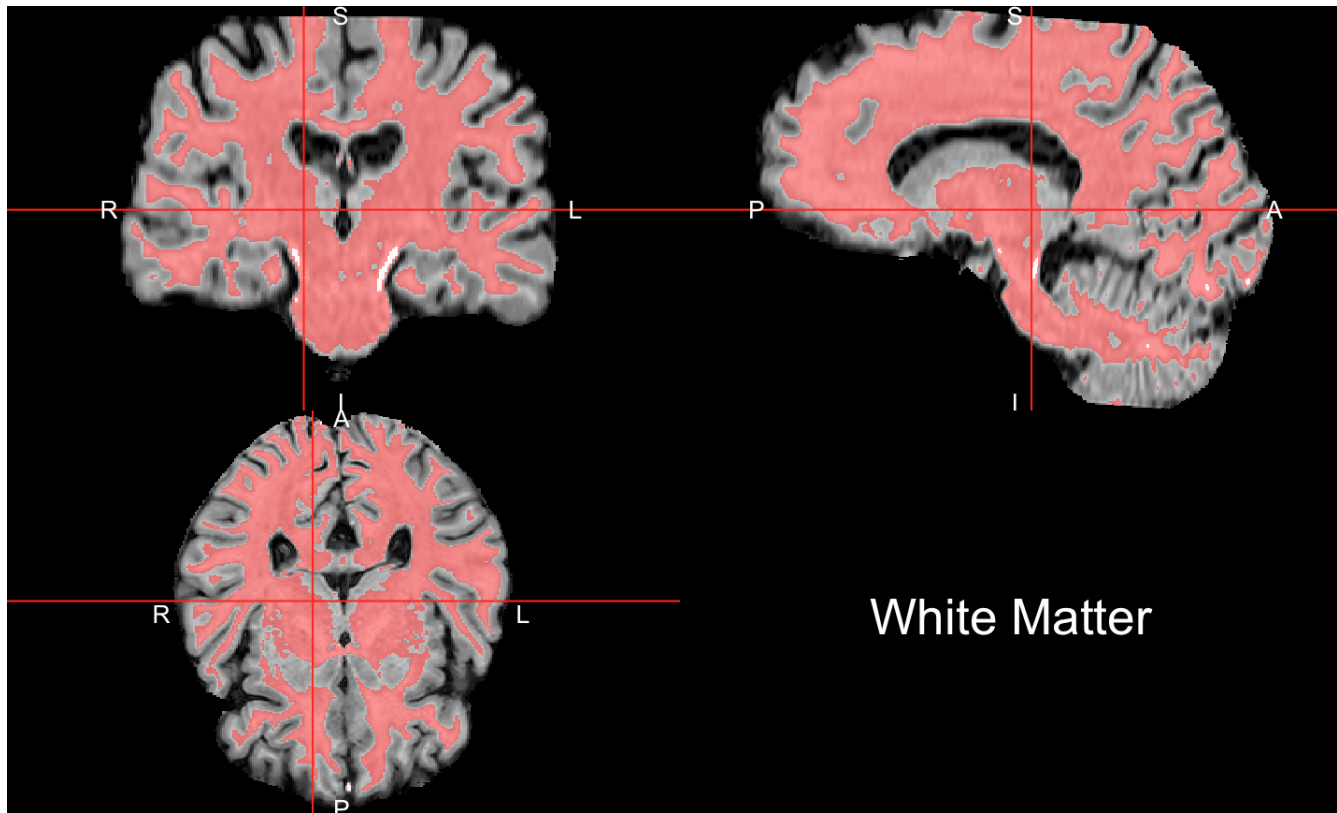

FAST Results

FAST assumes three tissue classes and produces an image with the three labels, ordered by increasing within-class mean intensities. In a T1 image, this results in:

- Level 1: CSF
- Level 2: Gray Matter
- Level 3: White Matter

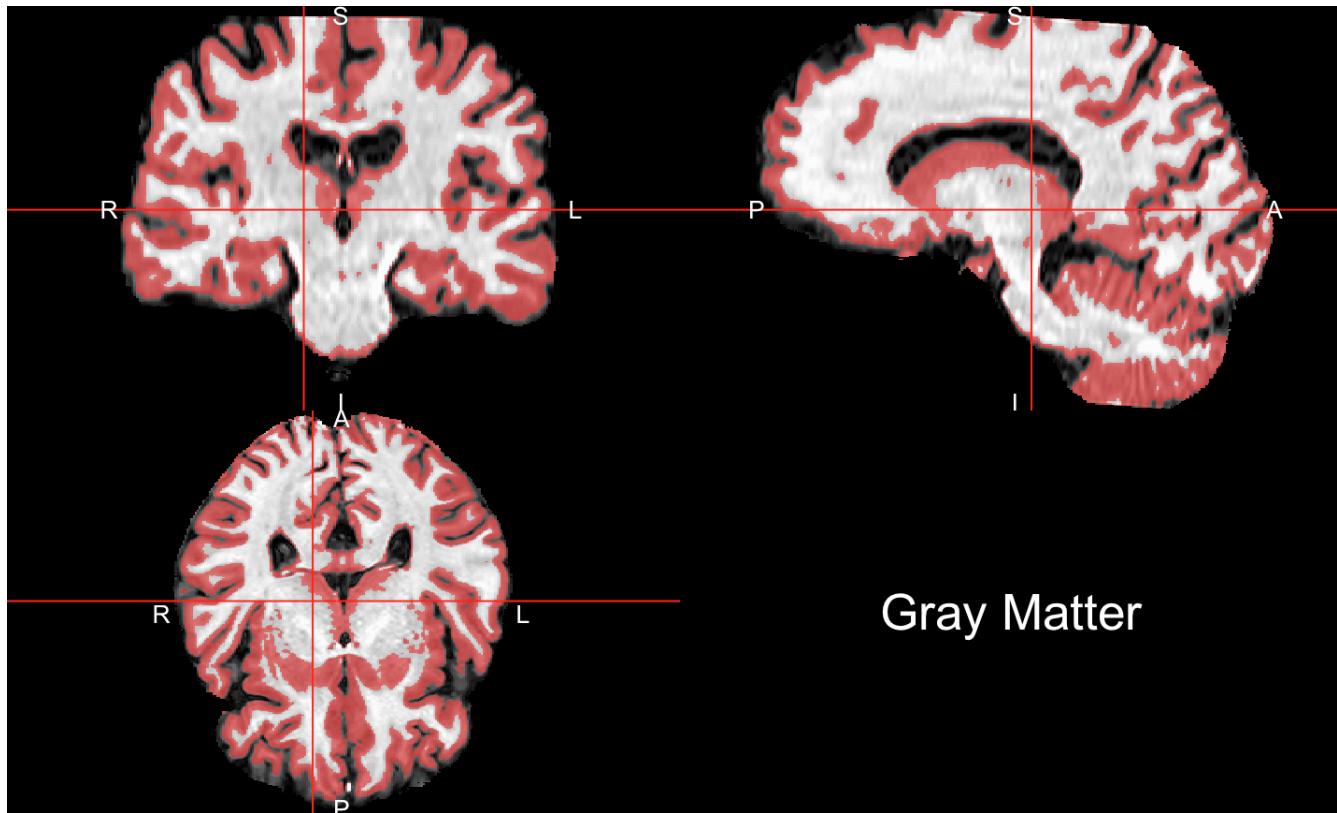
FAST: White Matter

```
ortho2(rt1, t1fast == 3, col.y = alpha("red", 0.5), text = "White Matter")
```



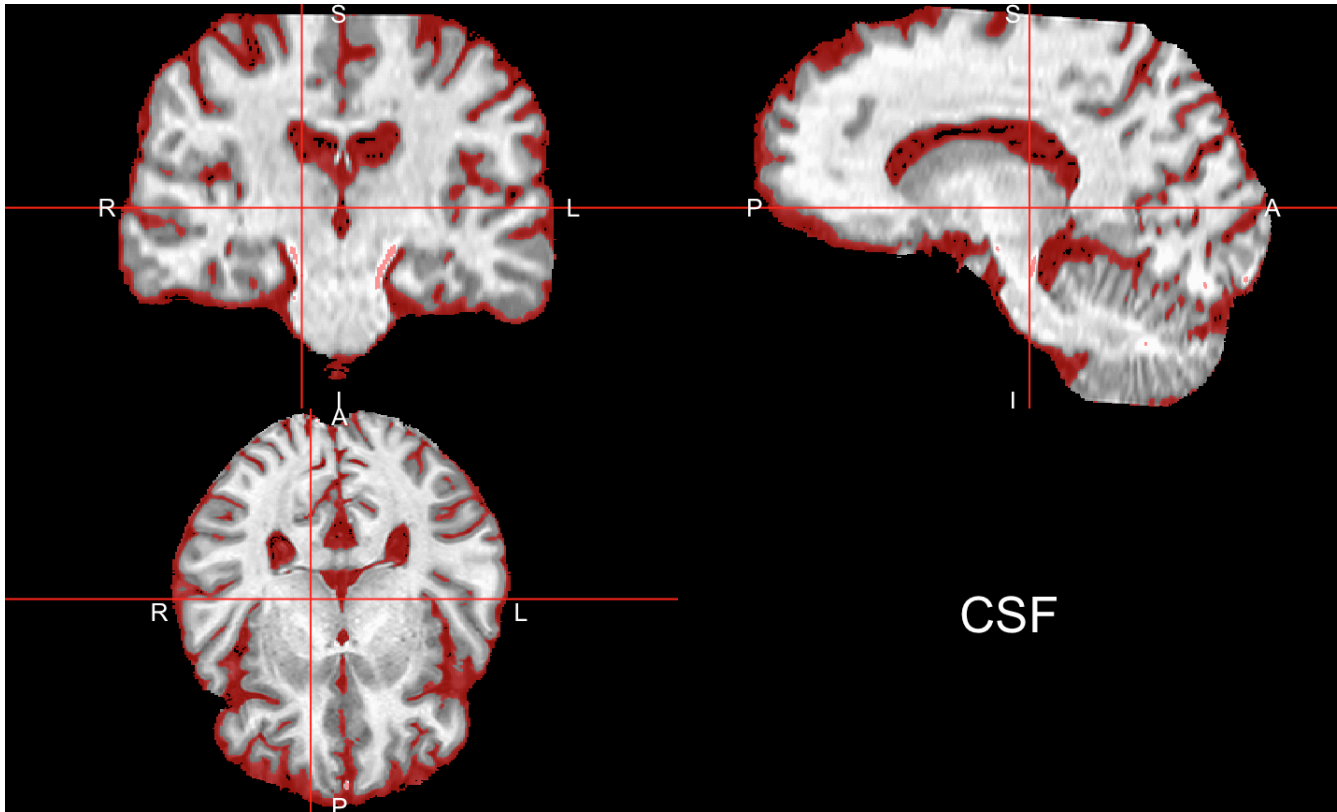
FAST: Gray Matter

```
ortho2(rtl1, t1fast == 2, col.y = alpha("red", 0.5), text = "Gray Matter")
```



FAST: CSF

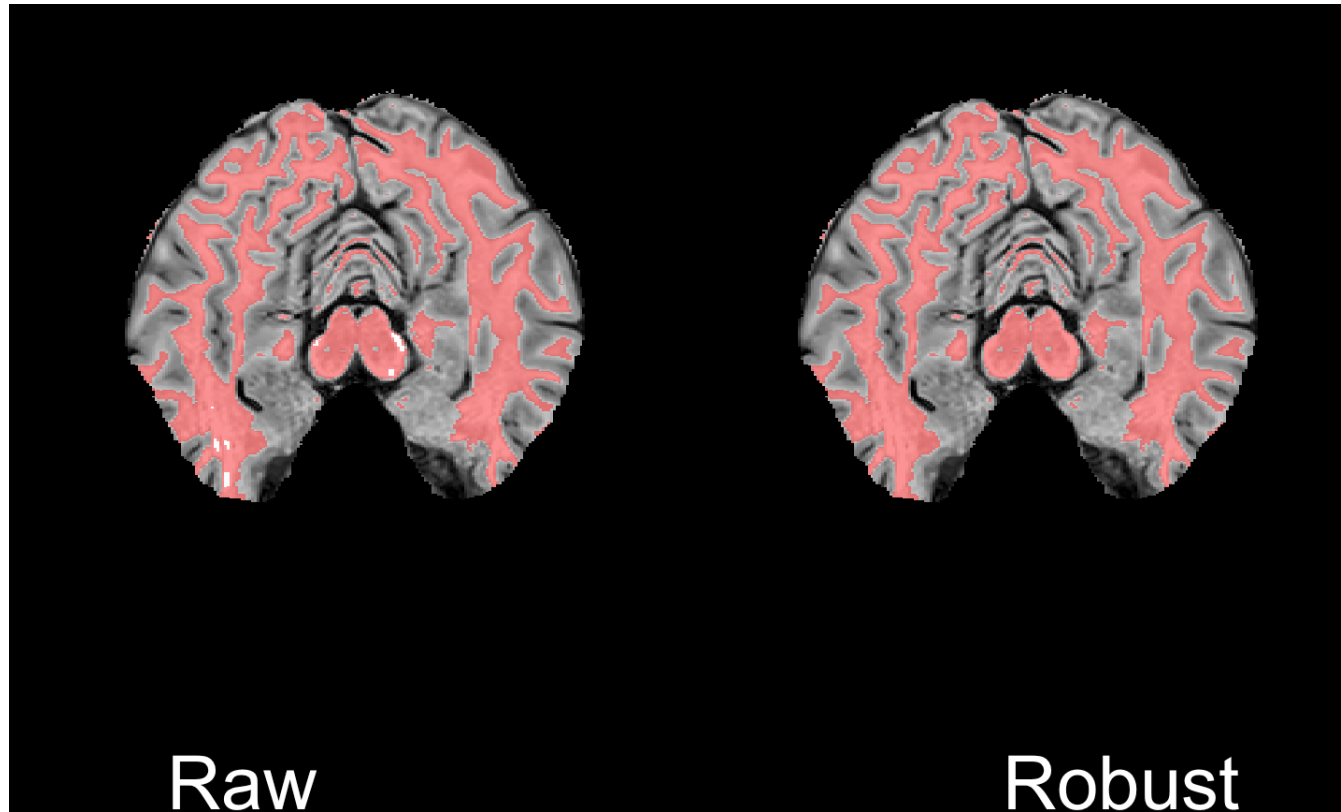
```
ortho2(rtl1, t1fast == 1, col.y = alpha("red", 0.5), text = "CSF")
```



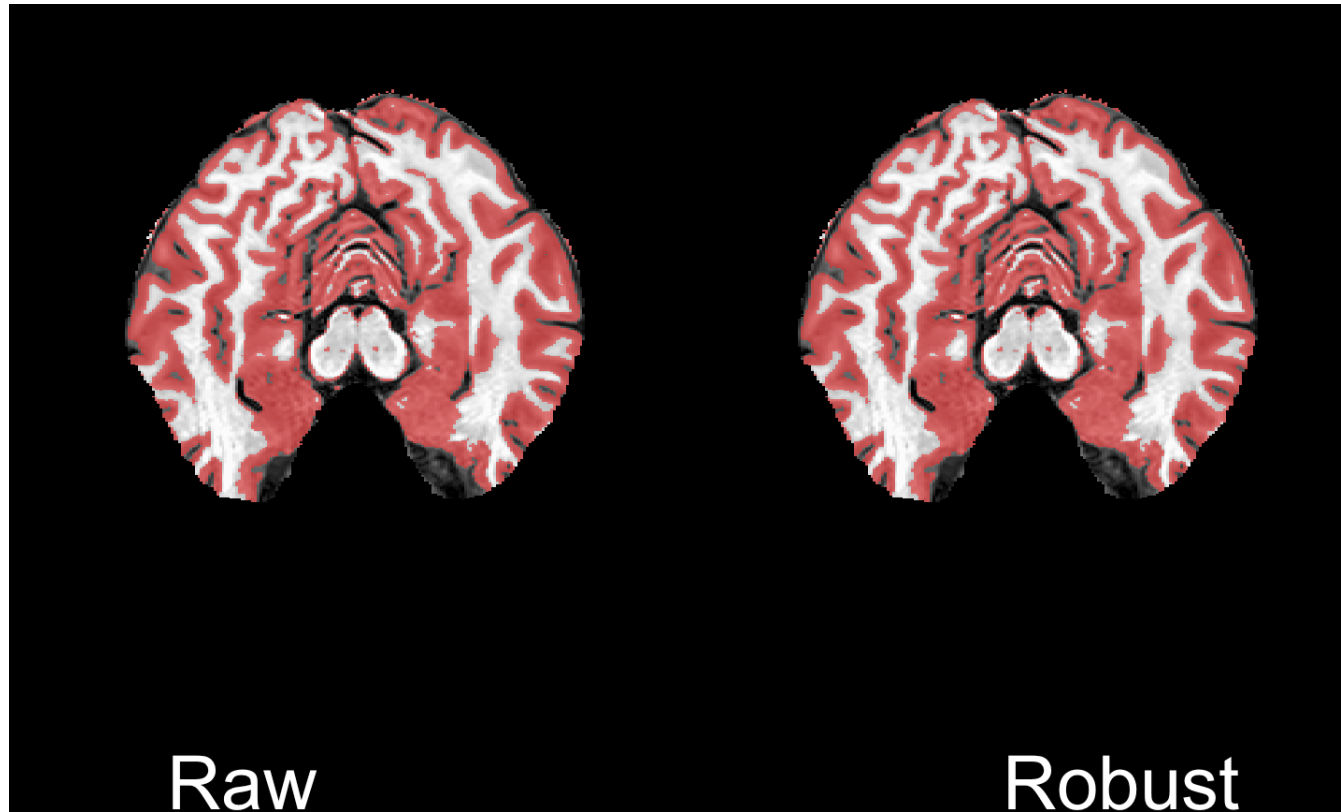
Removing large values: Is there an effect?

```
robust_fast = fast(rtl, # the robust window(tl)
  outfile = paste0(nii.stub(tlfile), "_FAST"),
  opts = "--nobias")
```

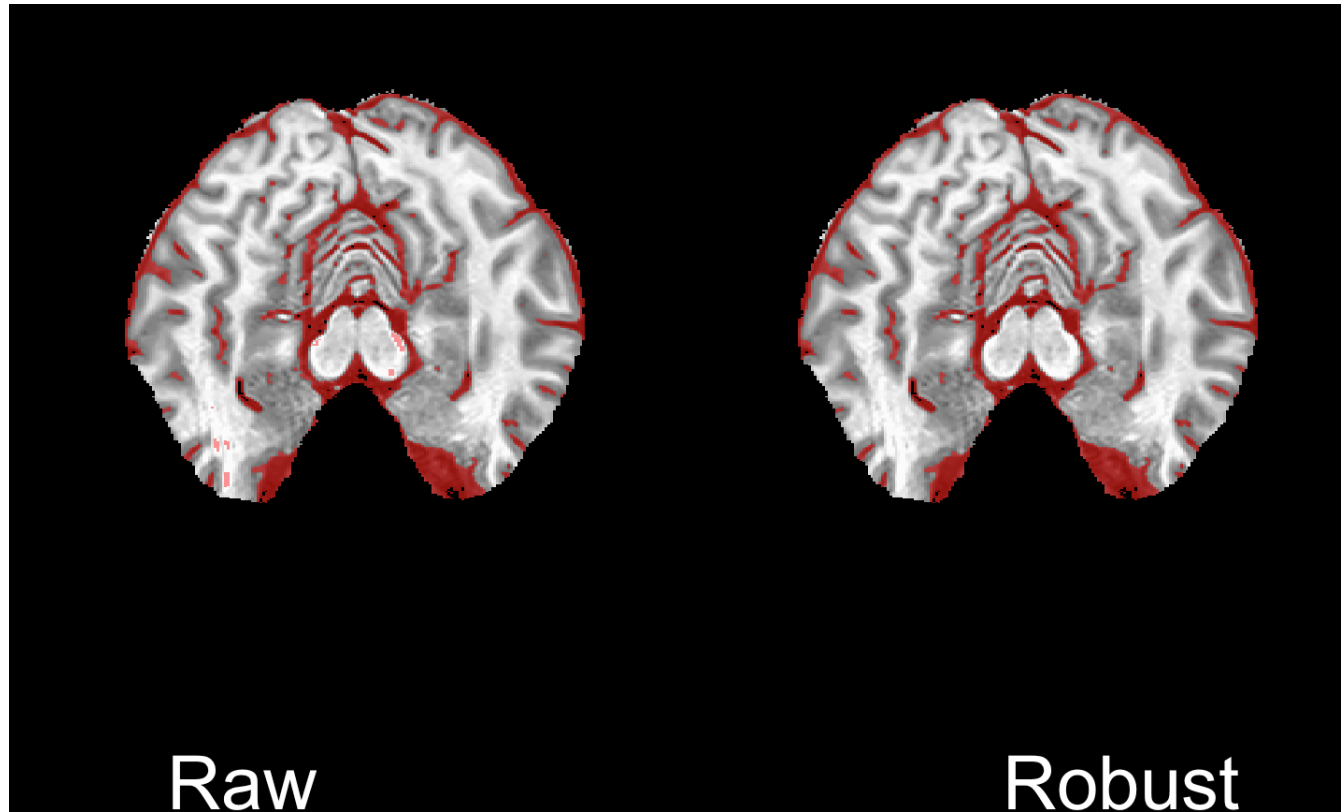
FAST with Window: White Matter



FAST with Window: Gray Matter



FAST with Window: CSF



FAST Results

- Overall the results look good
 - Not much difference after dampening outliers using `robust_window`

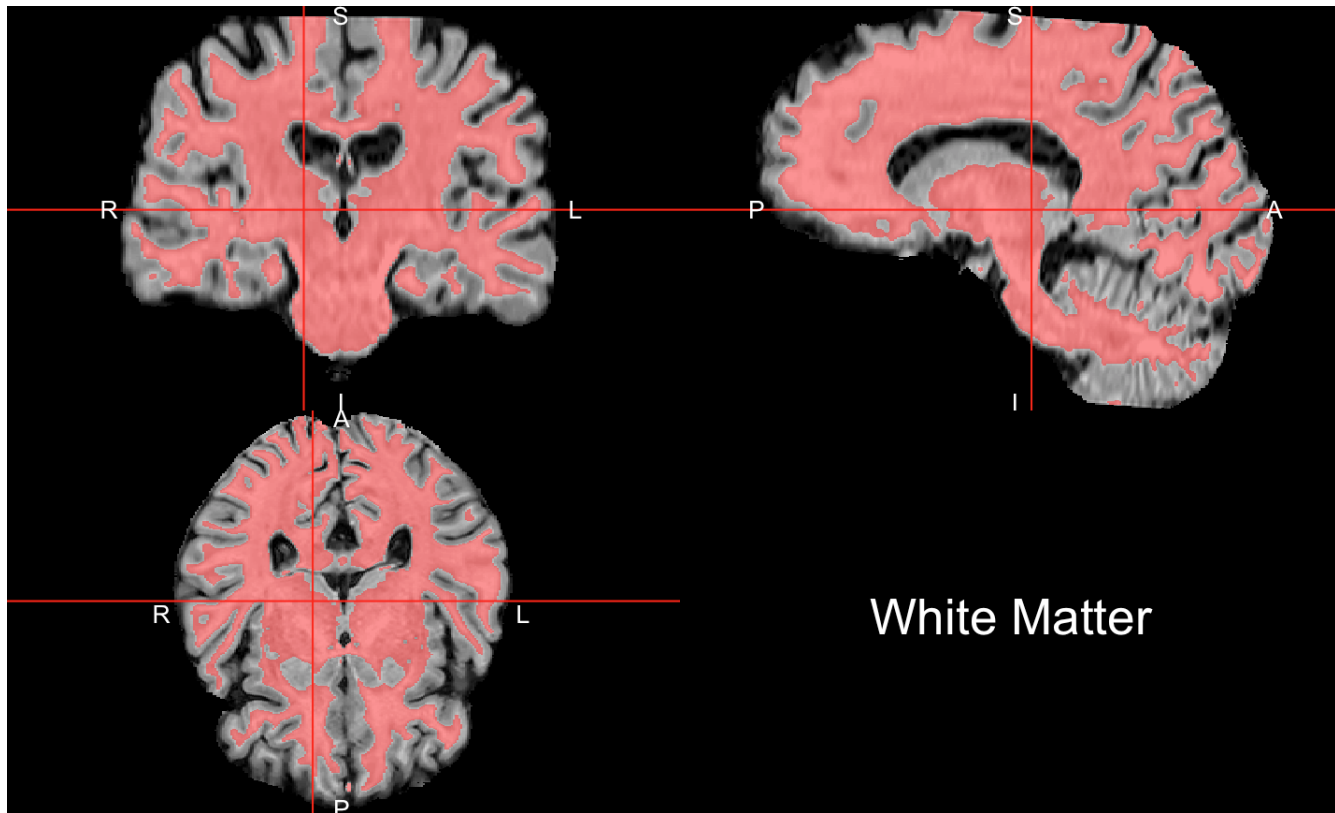
Tissue Segmentation using ANTsR, extrantsr

- Uses Atropos (Avants et al. 2011)
 - 3D K-means clustering + a Markov random field
- The `extrantsr::otropos` function works with `nifti` objects
 - calls `ANTsR::atropos` function

```
t1_otropos = otropos(a = t1, x = mask) # using original data  
t1seg = t1_otropos$segmentation
```

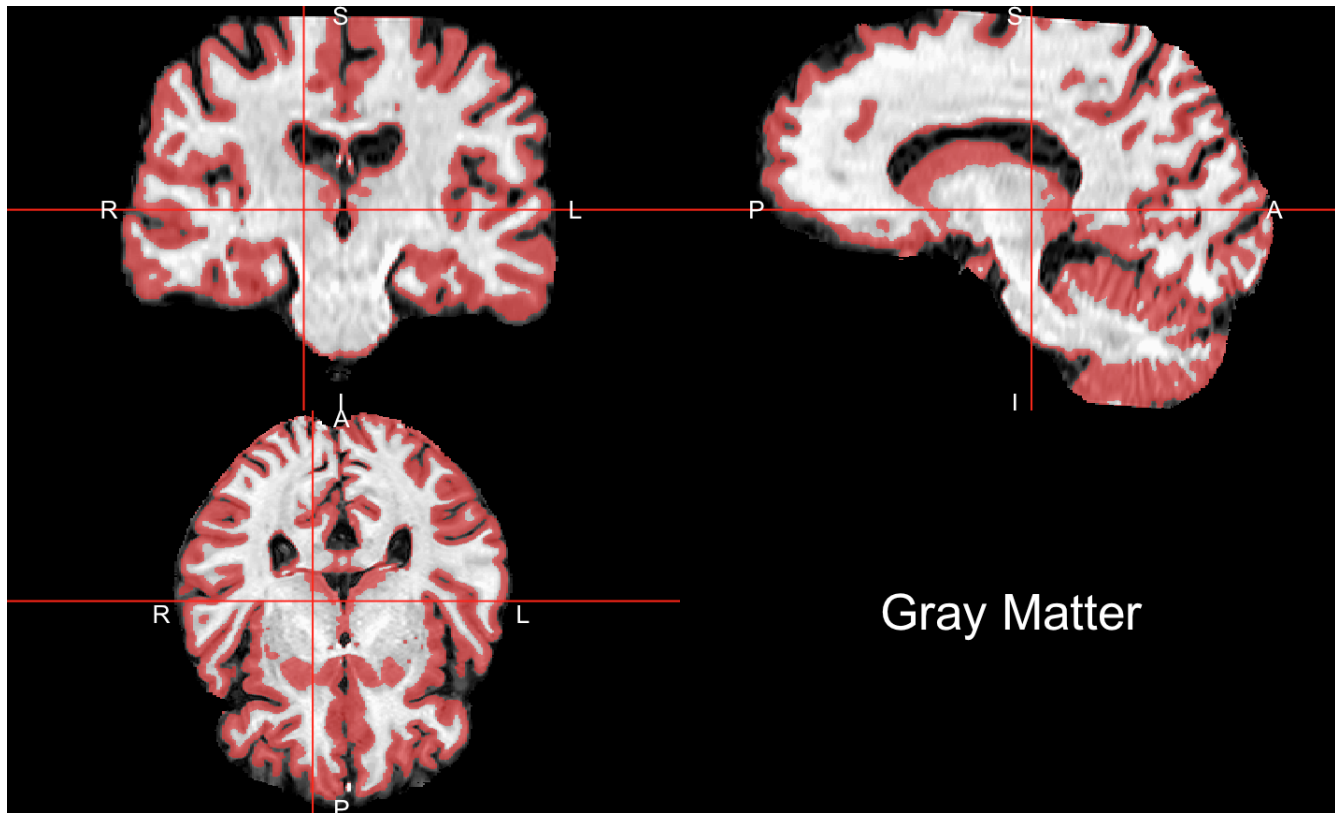
Atropos: White Matter

```
ortho2(rt1, t1seg == 3, col.y = alpha("red", 0.5), text = "White Matter")
```



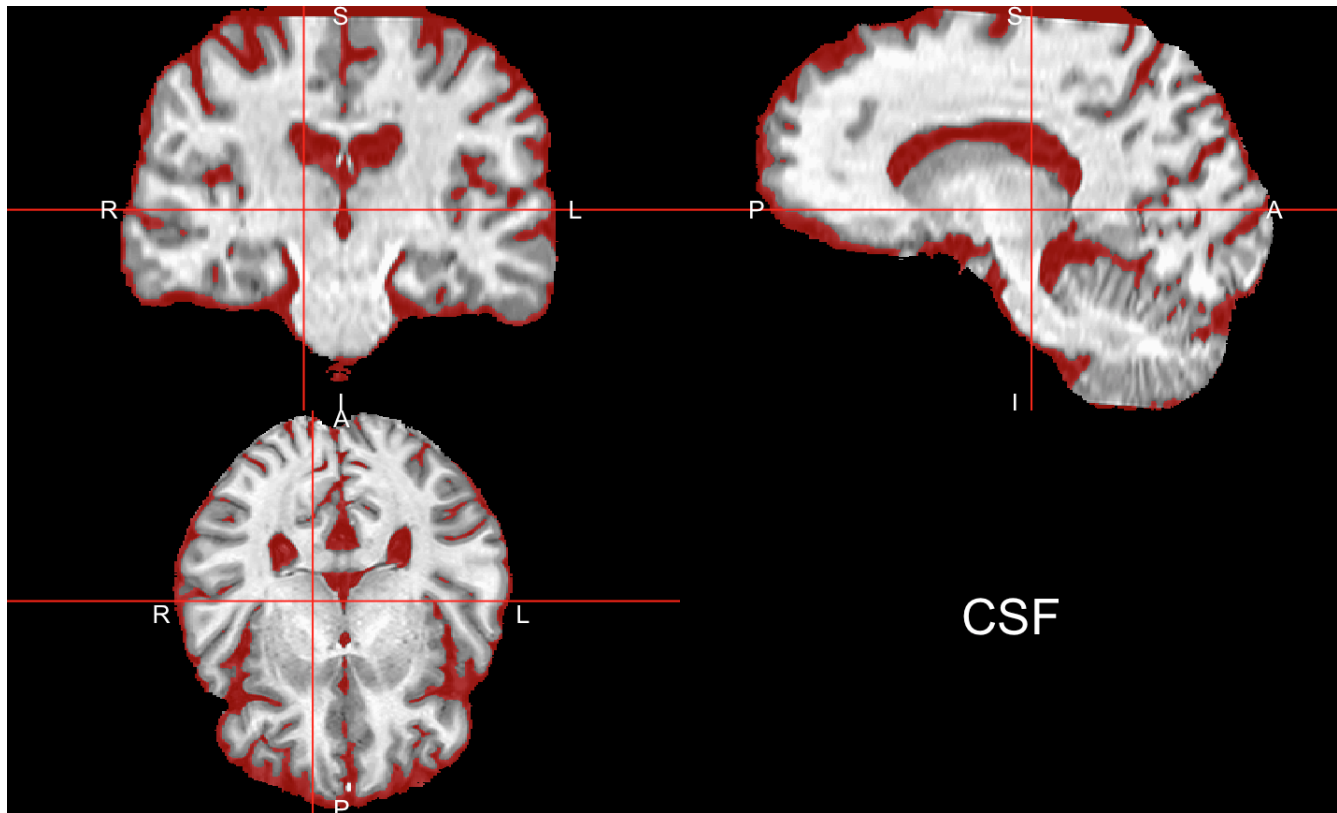
Atropos: Gray Matter

```
ortho2(rt1, t1seg == 2, col.y = alpha("red", 0.5), text = "Gray Matter")
```



Atropos: CSF

```
ortho2(rt1, t1seg == 1, col.y = alpha("red", 0.5), text = "CSF")
```



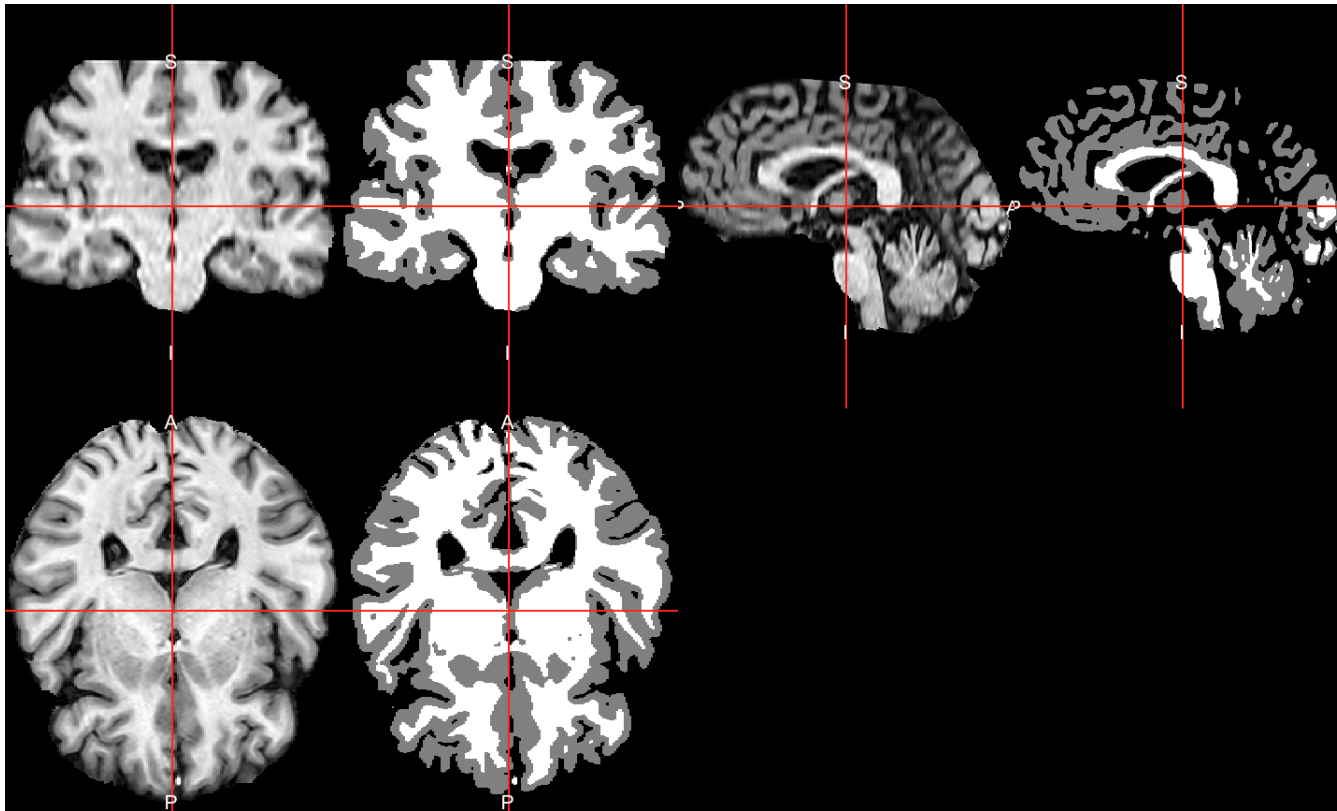
Default Atropos Results

- Overall the results look good
 - The k-means clustering is not affected by outliers in this case
- We will still try using `robust_window`

Atropos using Windowing

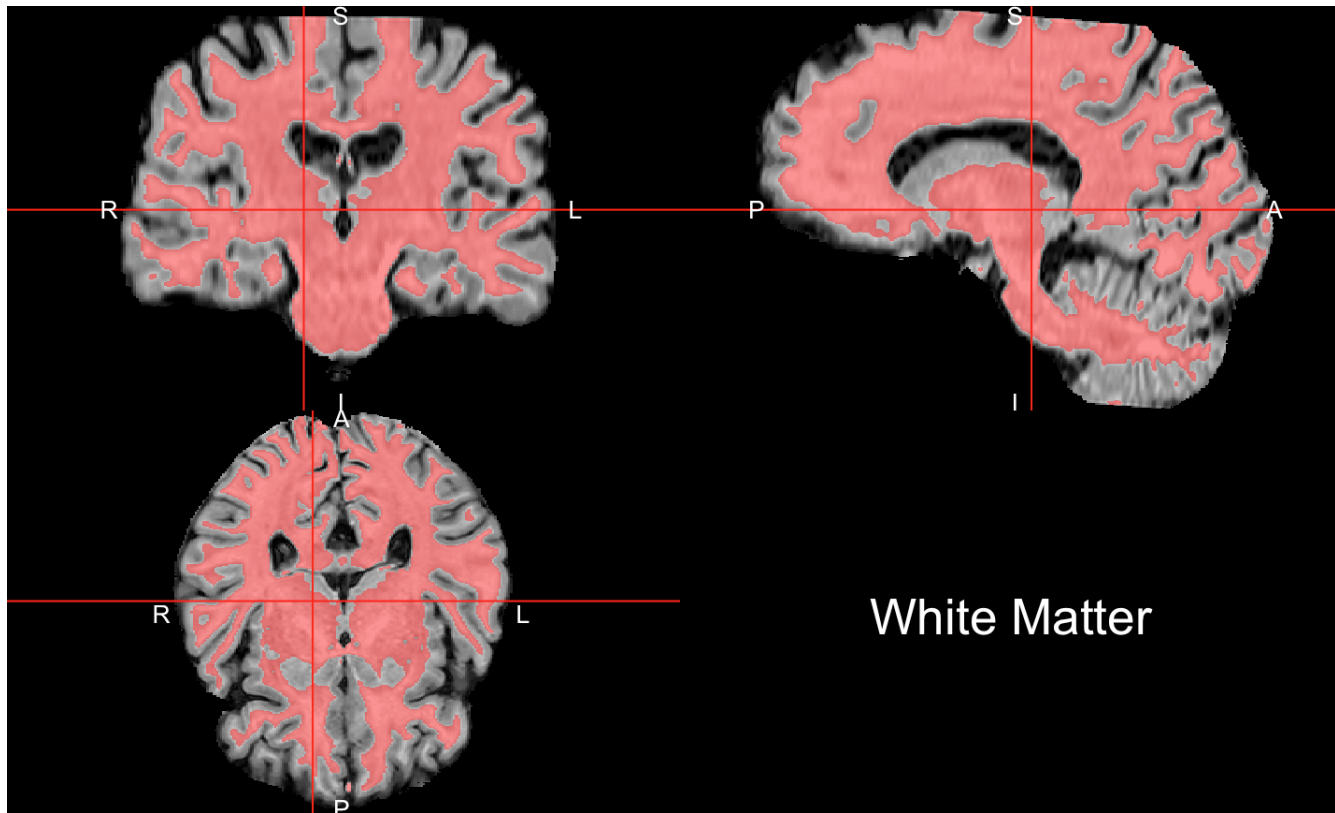
```
robust_t1_otropos = otropos(a = rt1, x = mask) # using robust  
robust_t1seg = robust_t1_otropos$segmentation
```

```
double_ortho(rt1, robust_t1seg)
```



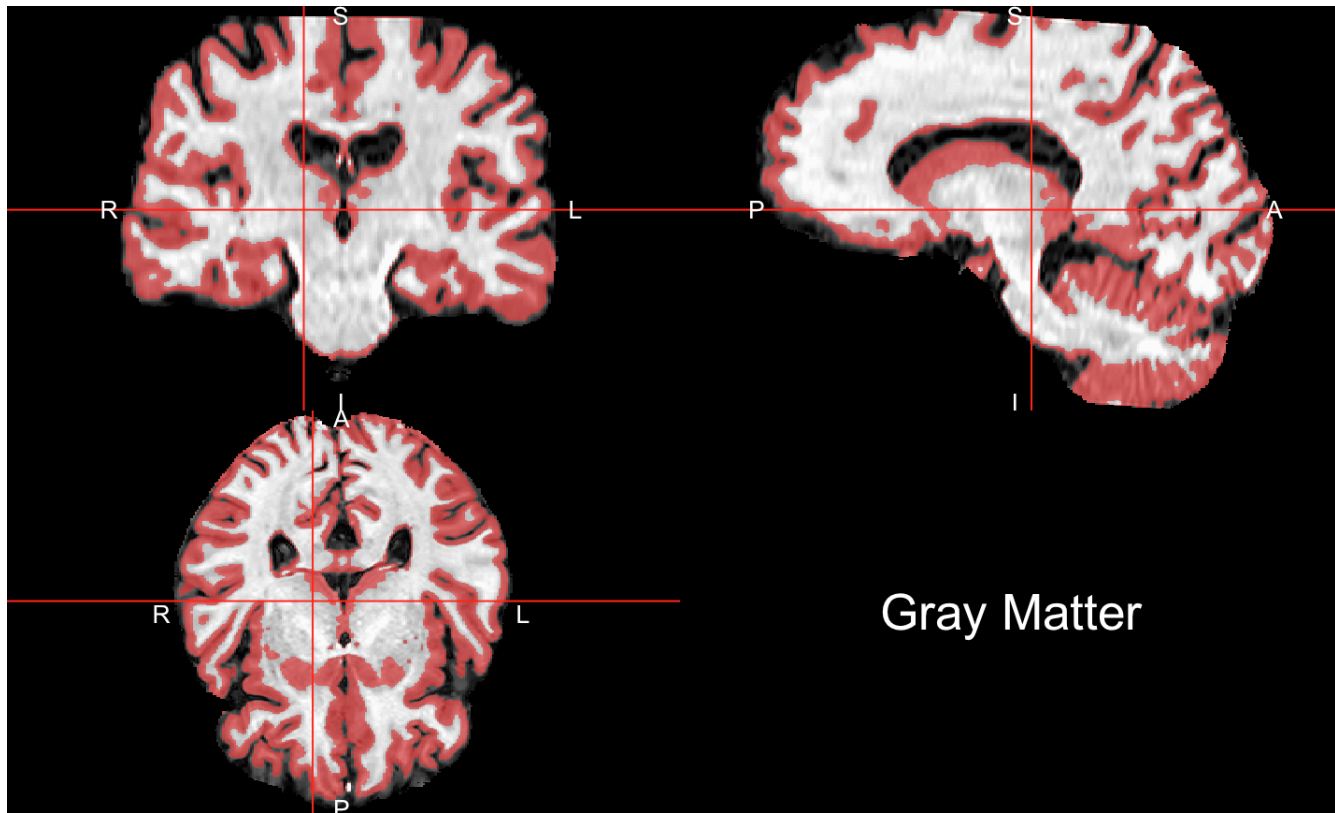
Atropos with Window: White Matter

```
ortho2(rt1, robust_t1seg == 3, col.y = alpha("red", 0.5), text = "White Matter")
```



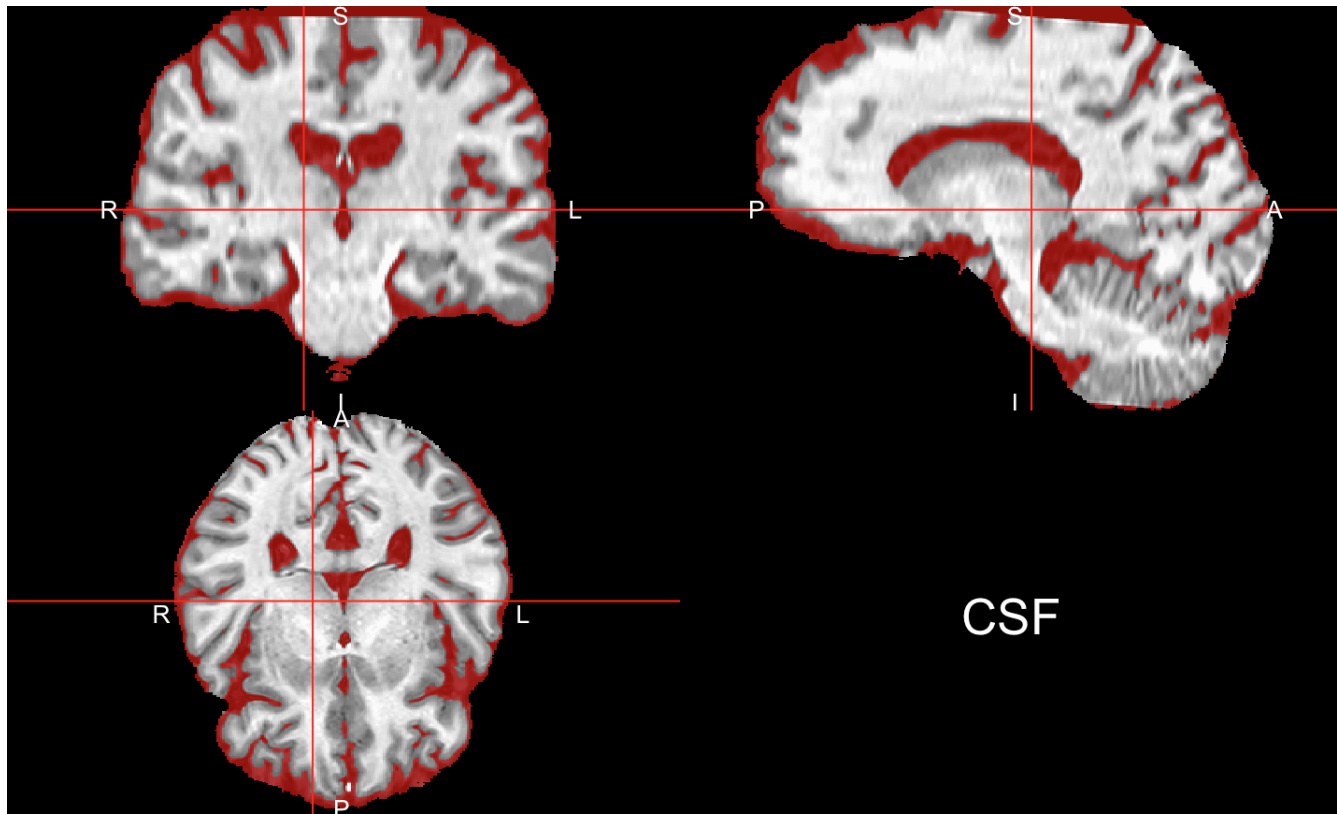
Atropos with Window: Gray Matter

```
ortho2(rt1, robust_t1seg == 2, col.y = alpha("red", 0.5), text = "Gray Matter")
```



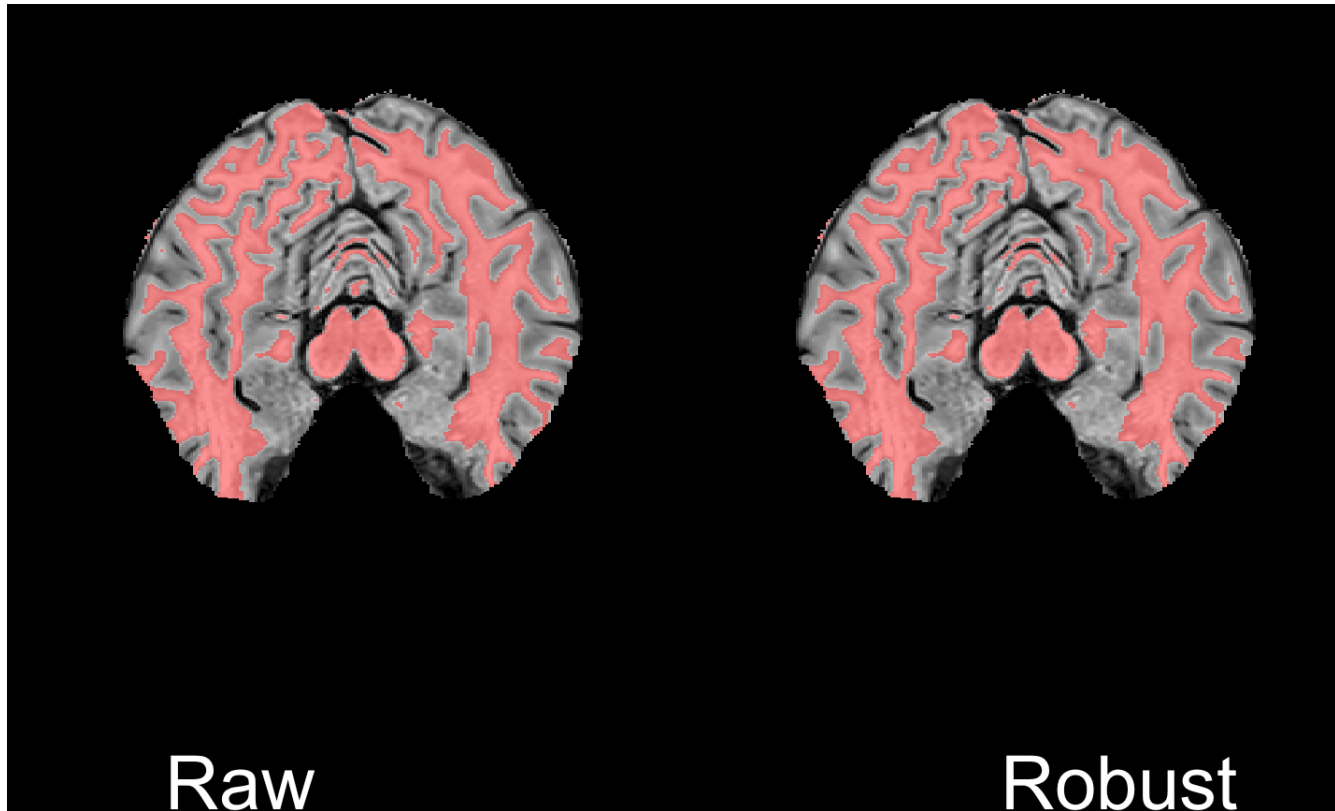
Atropos with Window: CSF

```
ortho2(rt1, robust_t1seg == 1, col.y = alpha("red", 0.5), text = "CSF")
```

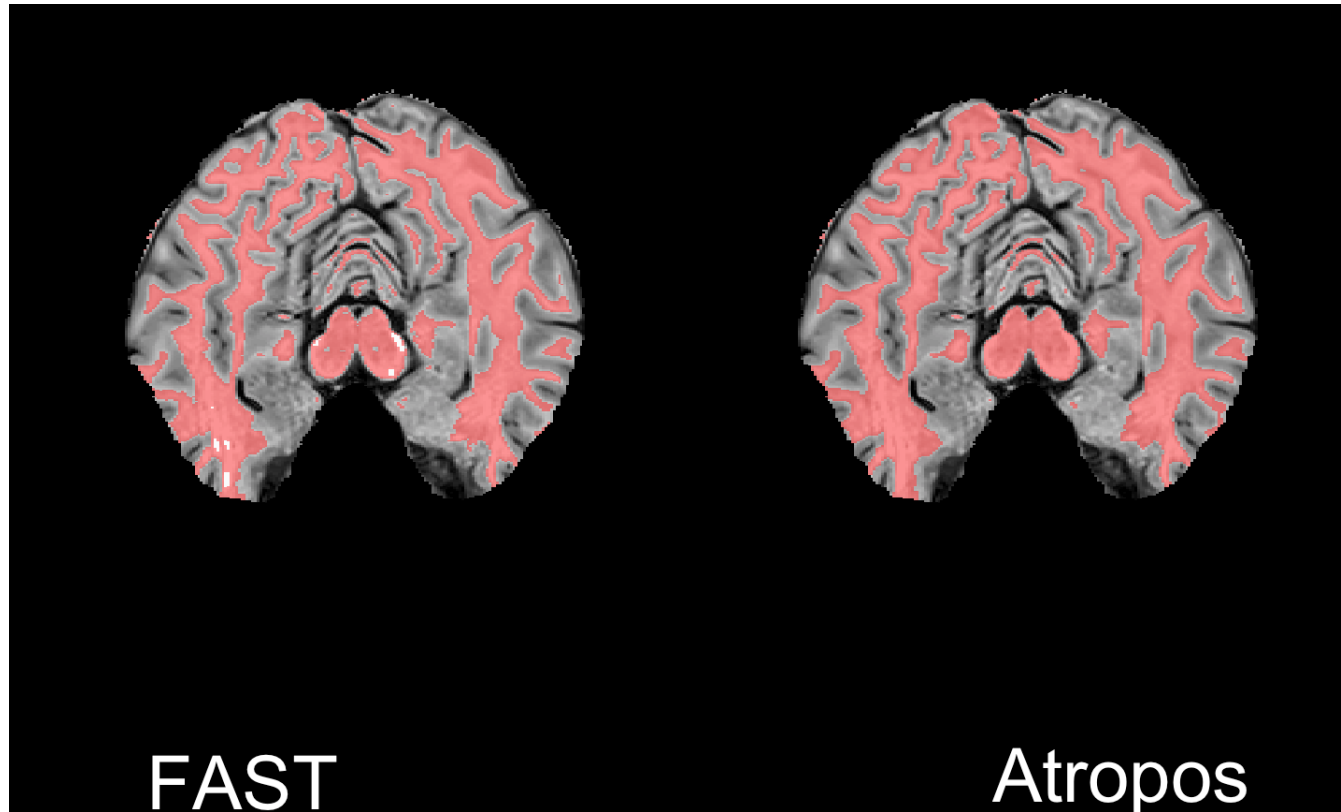


Atropos with Window Results

- Overall the results look like they reasonably separate the classes
 - No ground truth
- Winsorizing large outliers did not make a difference for this image



Atropos WM vs. FAST WM



Estimating the Volume of Each Class

We can create a table which will count the number of voxels in each category:

```
tab_fsl = table(robust_fast[ robust_fast != 0])
tab_ants = table(robust_t1seg[ robust_t1seg != 0])
names(tab_fsl) = names(tab_ants) = c("CSF", "GM", "WM")
tab_fsl
```

	CSF	GM	WM
	1657792	3247666	2792615

```
tab_ants
```

	CSF	GM	WM
	1568957	3217615	3062909

```
out_mask = robust_fast != 0 | robust_t1seg != 0
table(robust_fast[ out_mask ], robust_t1seg[ out_mask ])
```

	1	2	3
0	151408	0	0
1	1417317	240475	0
2	232	2973999	273435
3	0	3141	2789474

Estimating the Volume of Each Class

By multiplying by the voxel resolution (in cubic centimeters) using the `voxres` function, we can get volumes

```
vres = oro.nifti::voxres(t1, units = "cm")  
print(vres)
```

```
[1] 0.0001757813
```

```
vol_fsl = tab_fsl * vres  
vol_fsl
```

```
      CSF      GM      WM  
291.4088 570.8788 490.8894
```

```
vol_ants = tab_ants * vres  
vol_ants
```

```
      CSF      GM      WM  
275.7932 565.5964 538.4020
```

Website

http://johnmuschelli.com/imaging_in_r

References

Avants, Brian B, Nicholas J Tustison, Jue Wu, Philip A Cook, and James C Gee. 2011. "An Open Source Multivariate Framework for N-Tissue Segmentation with Evaluation on Public Data." 9 (4). Springer: 381–400.

Zhang, Yongyue, Michael Brady, and Stephen Smith. 2001. "Segmentation of Brain MR Images Through a Hidden Markov Random Field Model and the Expectation-Maximization Algorithm." 20 (1): 45–57. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=906424.