Tissue-Class Segmentation

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 gray matter structures, segmentation of pathology (MS lesions, tumors, ...)

Goals of this tutorial

- Perform tissue segmentation in R using FSL and ANTs.
- Discuss multi-atlas label fusion techniques for segmentation.
- Perform automatic MS lesion segmentation using OASIS.

Loading Data

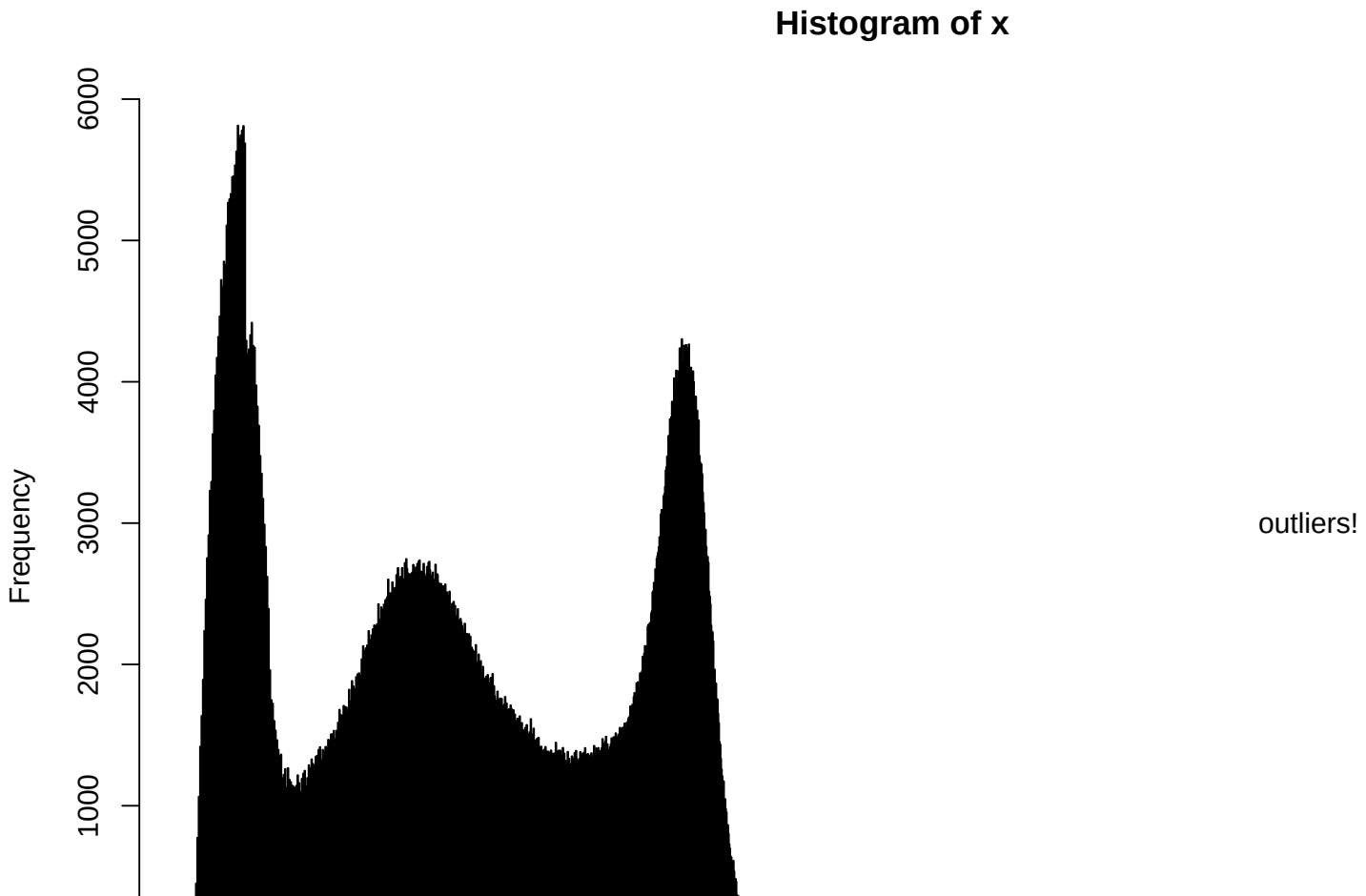
- Let's read in the training T1s and brain mask for subject 05.

```
library(ms.lesion)
library(neurobase)
library(fslr)
library(scales)
library(extrantsr)
all_files = get_image_filenames_list_by_subject(
  group = "training",
  type = "coregistered")
files = all_files$training05 # NOT training subject 1!
t1 = readnii(files["MPRAGE"])
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.

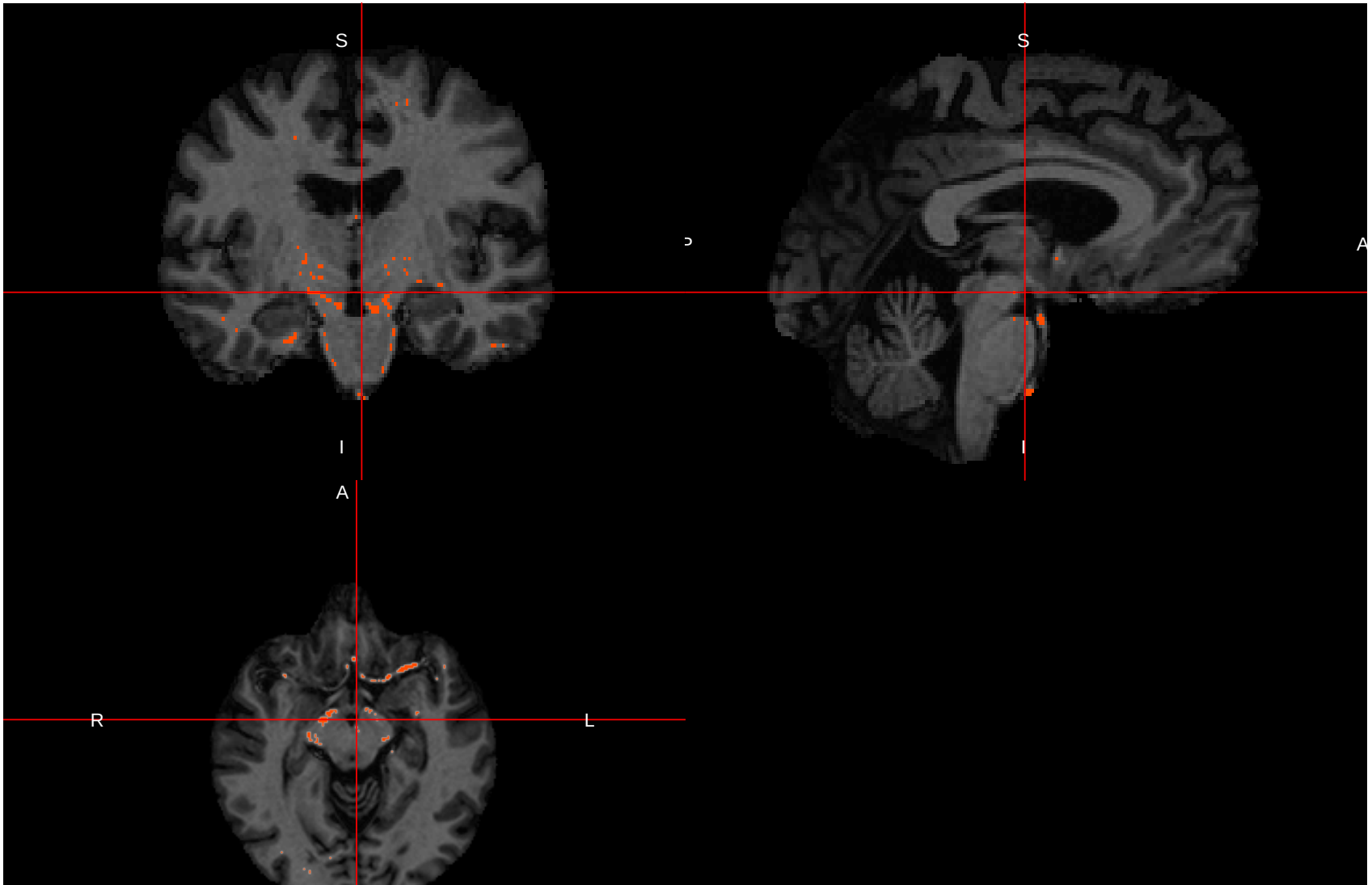
```
hist(t1, mask = mask, breaks = 2000); text(x = 800, y = 300
```



Where are the outliers

We see some values that may have be improperly segmented:

```
ortho2(t1, t1 > 400, xyz = xyz(t1 > 400))
```



Tissue Segmentation using FSL FAST

- FAST is based on a hidden Markov random field model and an Expectation-Maximization algorithm.
- 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.

Tissue Segmentation using FSL FAST

- The `fslr` function fast calls fast from FSL (Zhang, Brady, and Smith 2001). The `--nobias` option tells FSL to not perform inhomogeneity correction (was already performed in `ANTsR`).

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

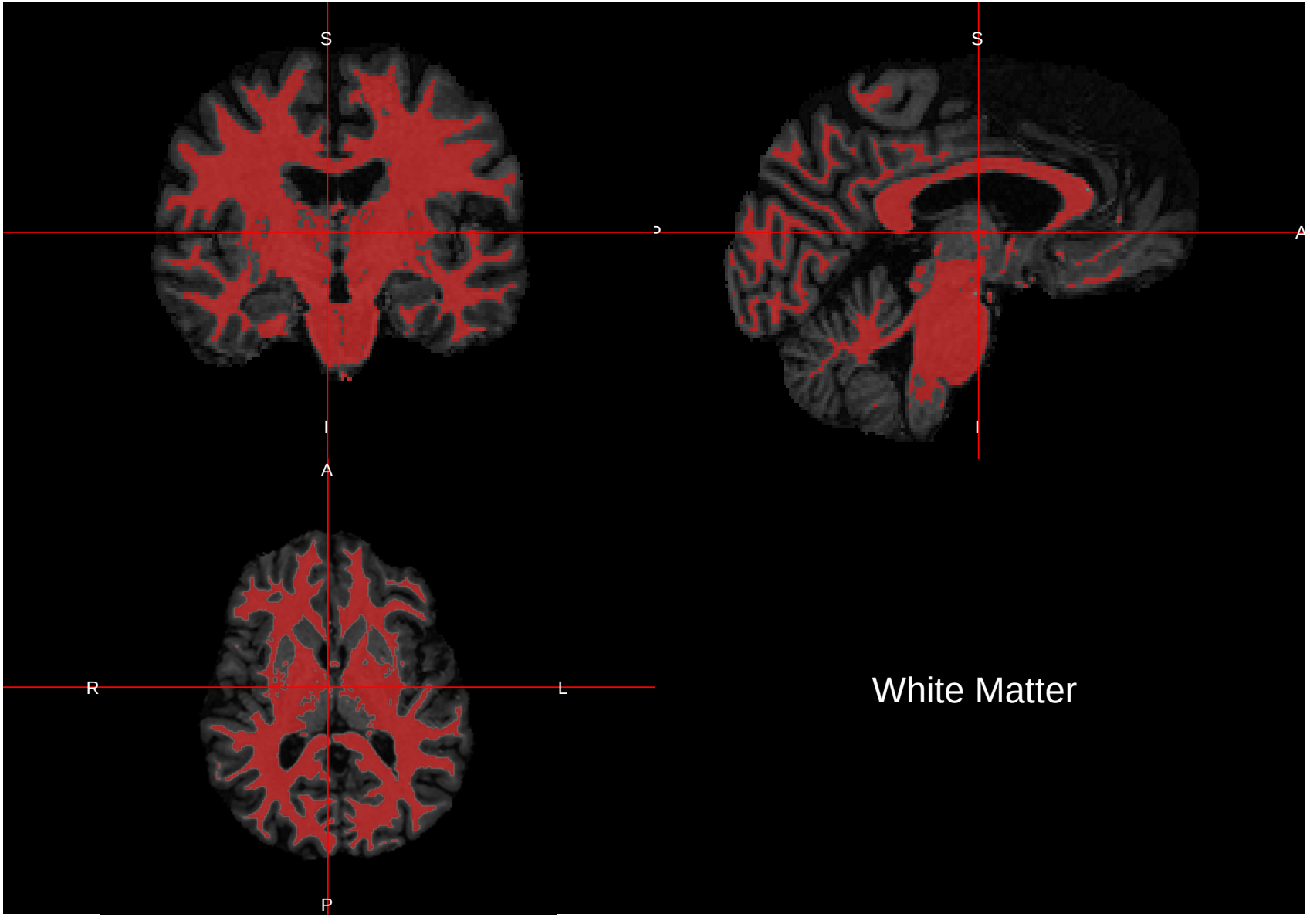
Results

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

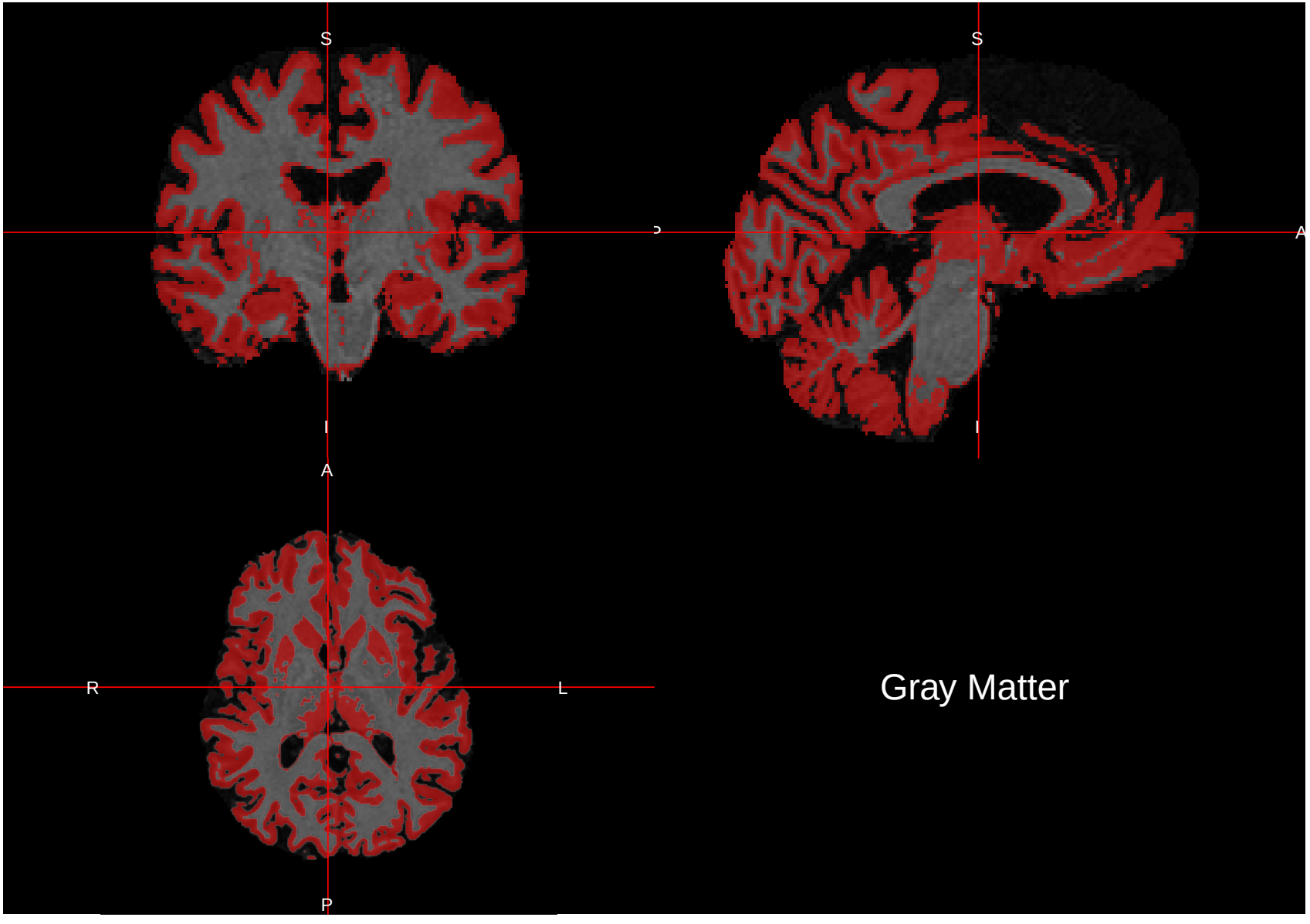
White Matter

```
ortho2(t1, t1fast == 3, col.y = alpha("red", 0.5), text = '
```



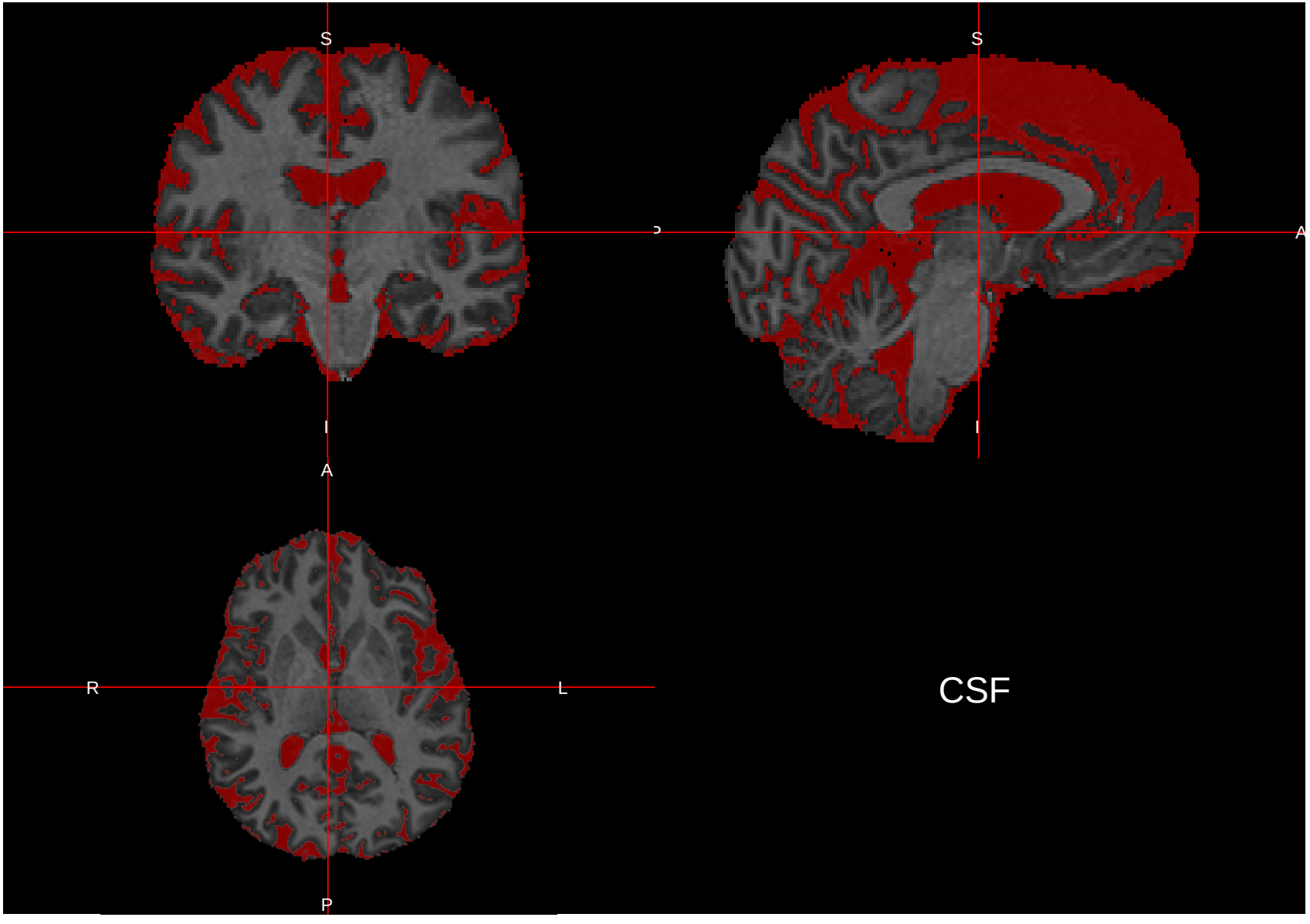
Gray Matter

```
ortho2(t1, t1fast == 2, col.y = alpha("red", 0.5), text = '
```



CSF

```
ortho2(t1, t1fast == 1, col.y = alpha("red", 0.5), text = '
```

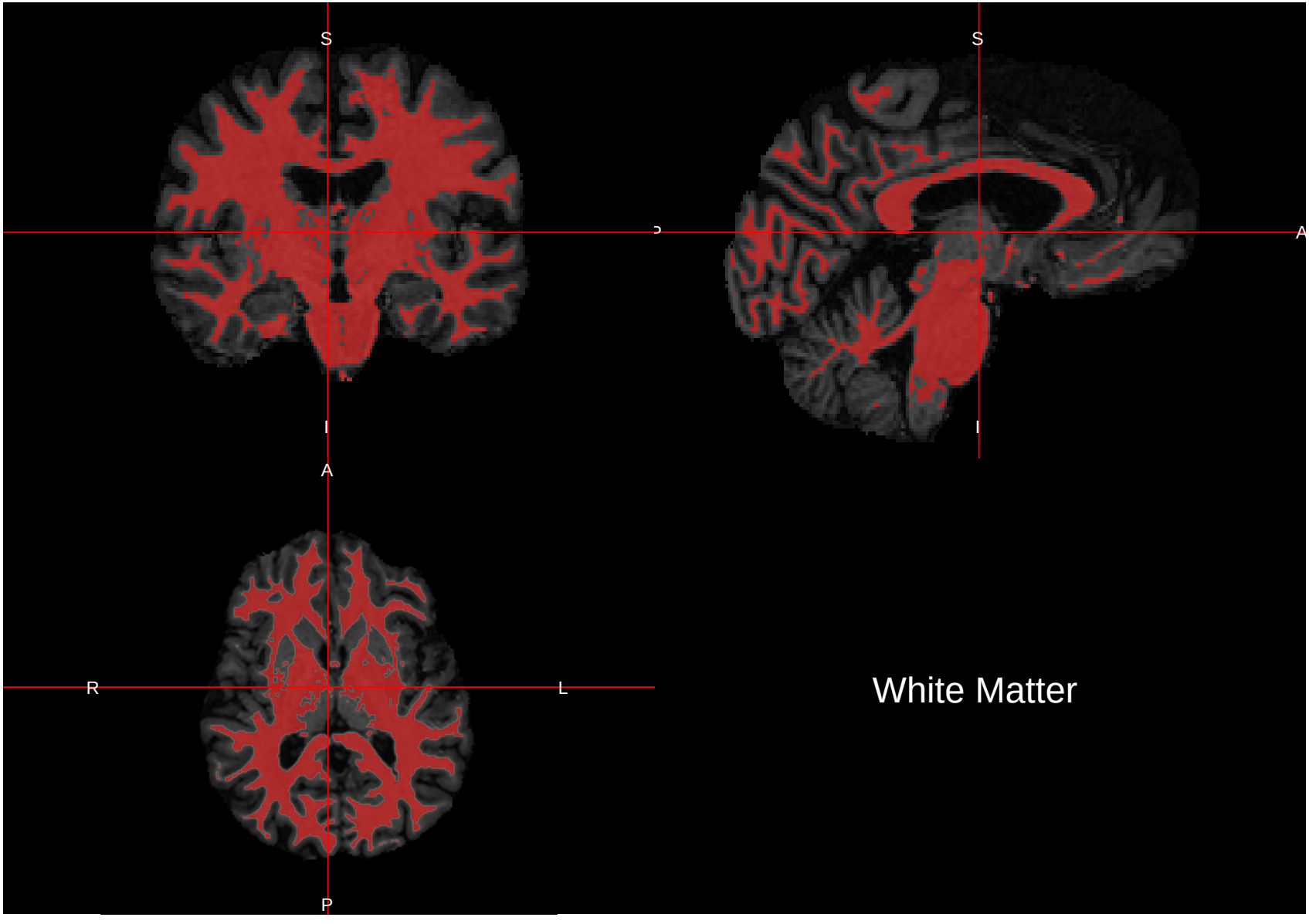


Windowing

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

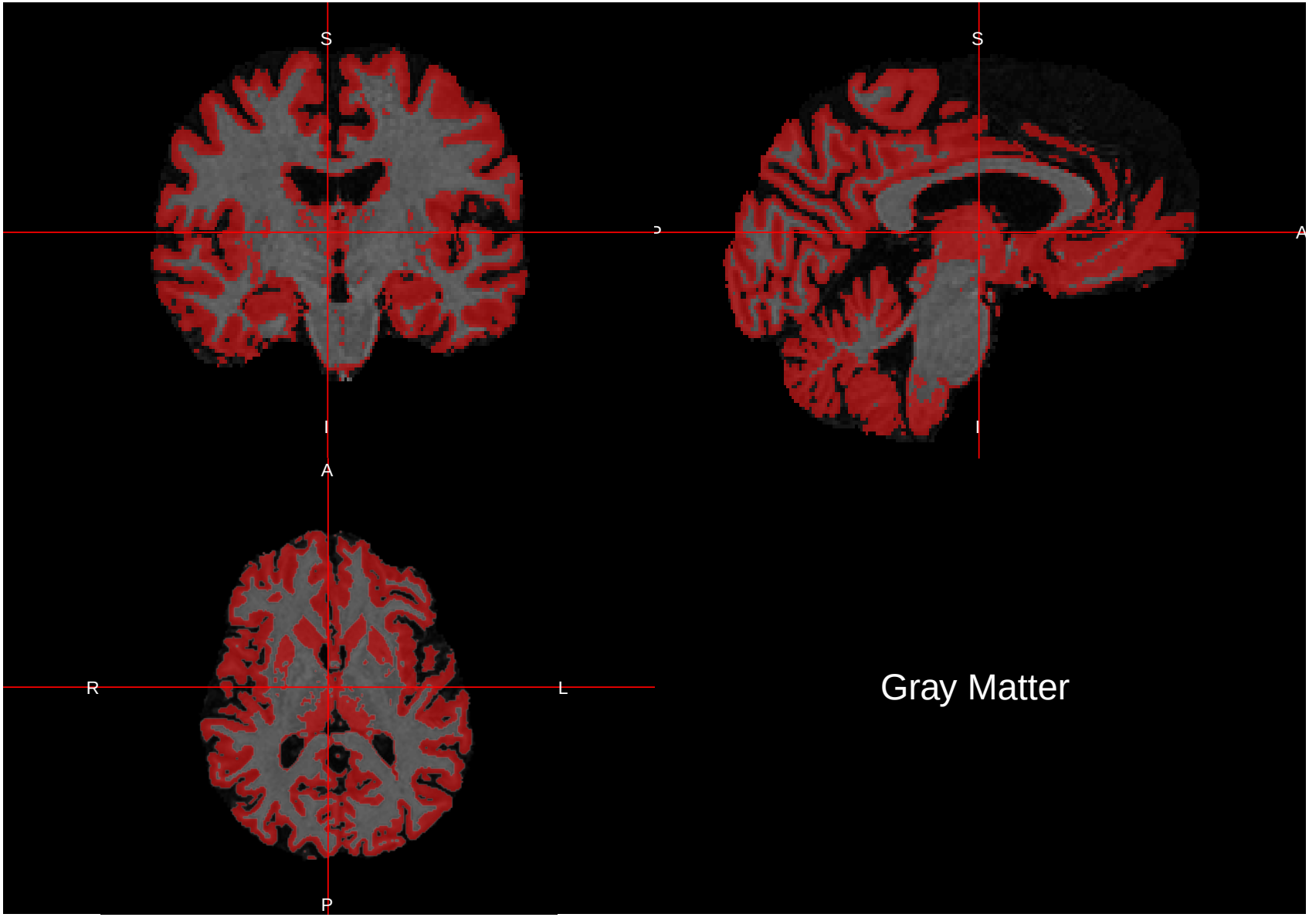
White Matter

```
ortho2(t1, robust_fast == 3, col.y = alpha("red", 0.5), tex
```



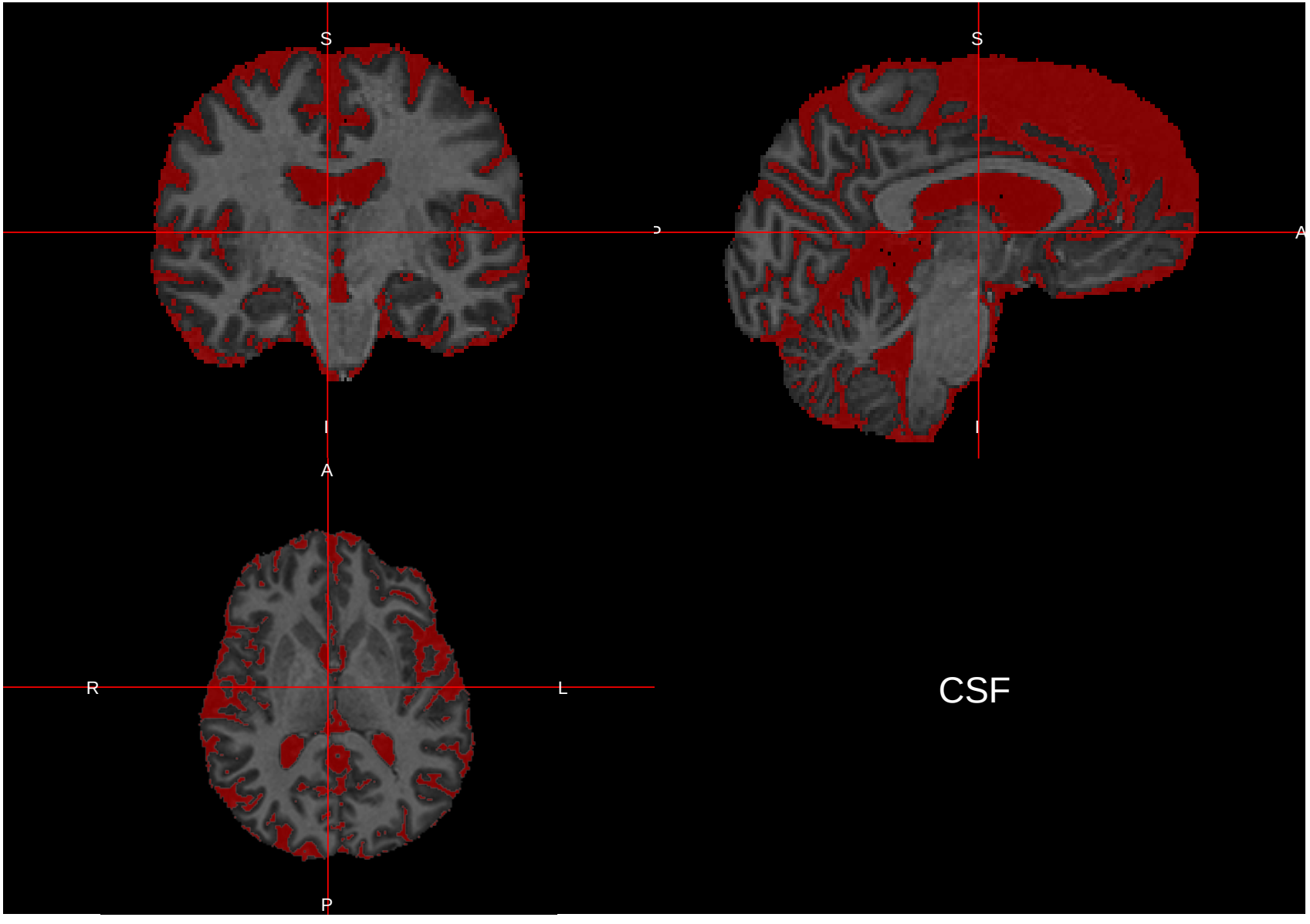
Gray Matter

```
ortho2(t1, robust_fast == 2, col.y = alpha("red", 0.5), tex
```



CSF

```
ortho2(t1, robust_fast == 1, col.y = alpha("red", 0.5), tex
```



FAST Results

- Overall the results look good
  - Not much difference after dampening outliers using `robust_window`
- FAST is robust to noise

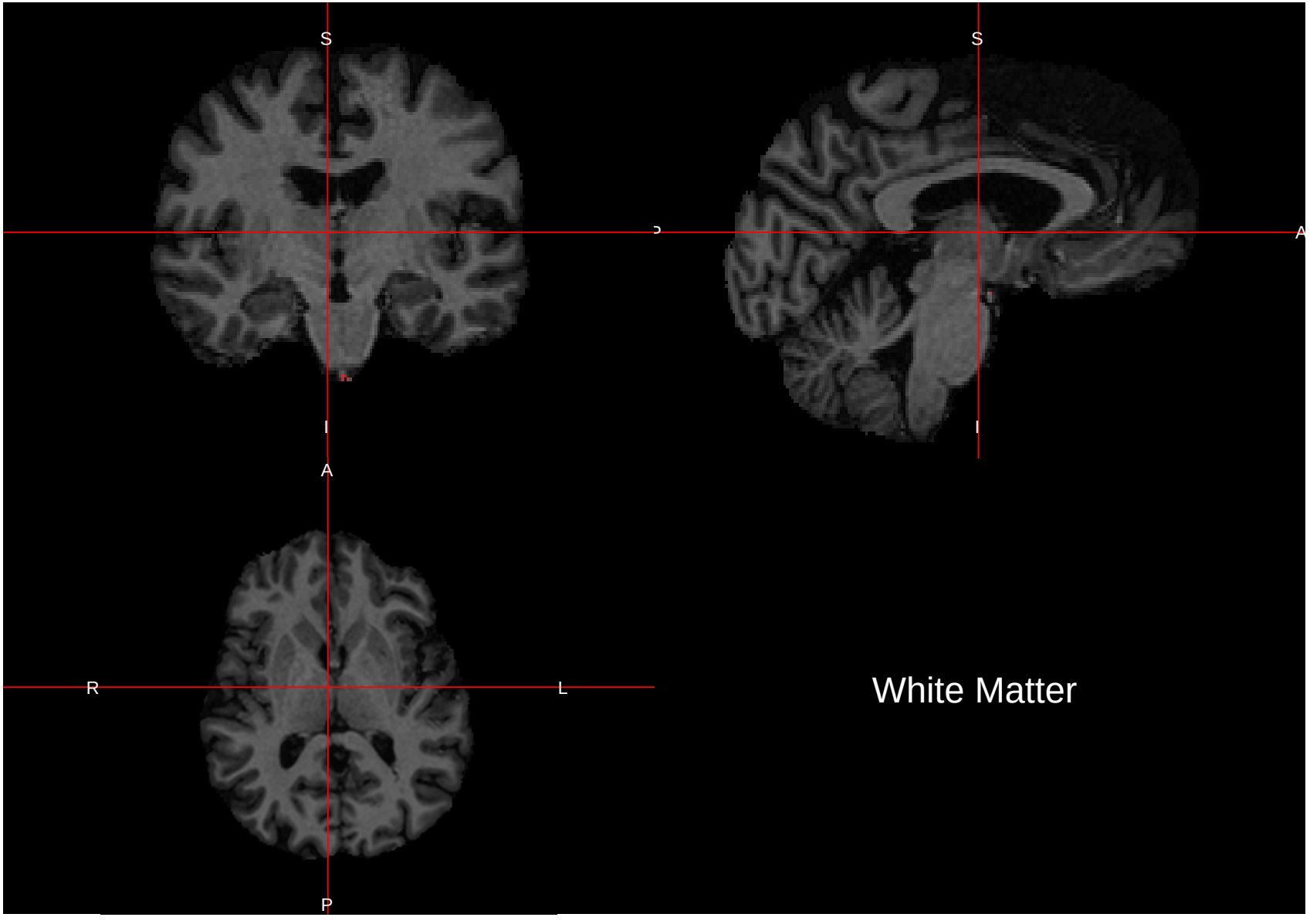
Tissue Segmentation using ANTsR, extrantsr

- Uses Atropos (Avants et al. 2011)
  - 3D K-means clustering + a Markov random field

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

White Matter

```
ortho2(t1, t1seg == 3, col.y = alpha("red", 0.5), text = "W
```

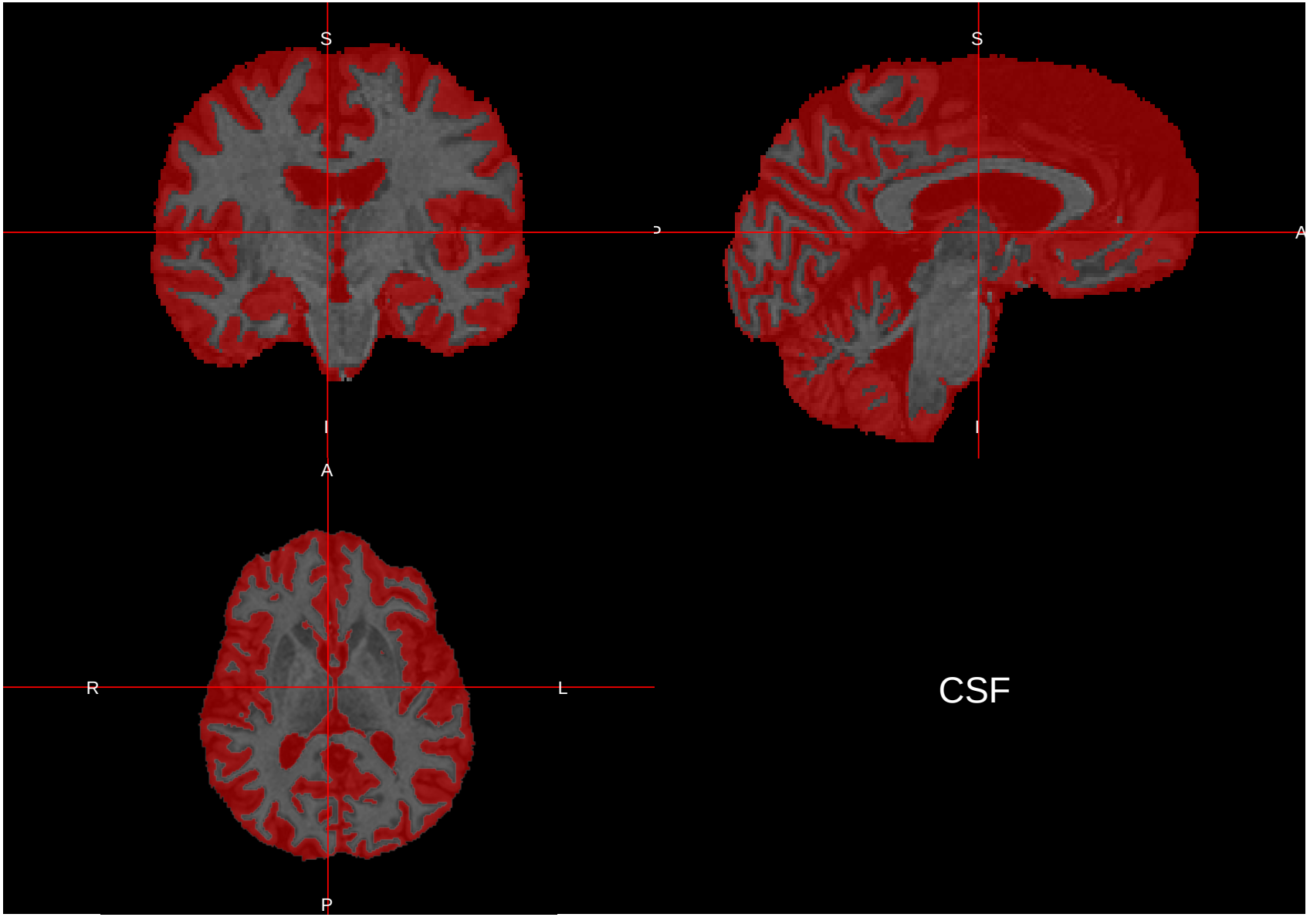


Gray Matter

```
ortho2(t1, t1seg == 2, col.y = alpha("red", 0.5), text = "G
```

CSF

```
ortho2(t1, t1seg == 1, col.y = alpha("red", 0.5), text = "C
```



Default Atropos Results

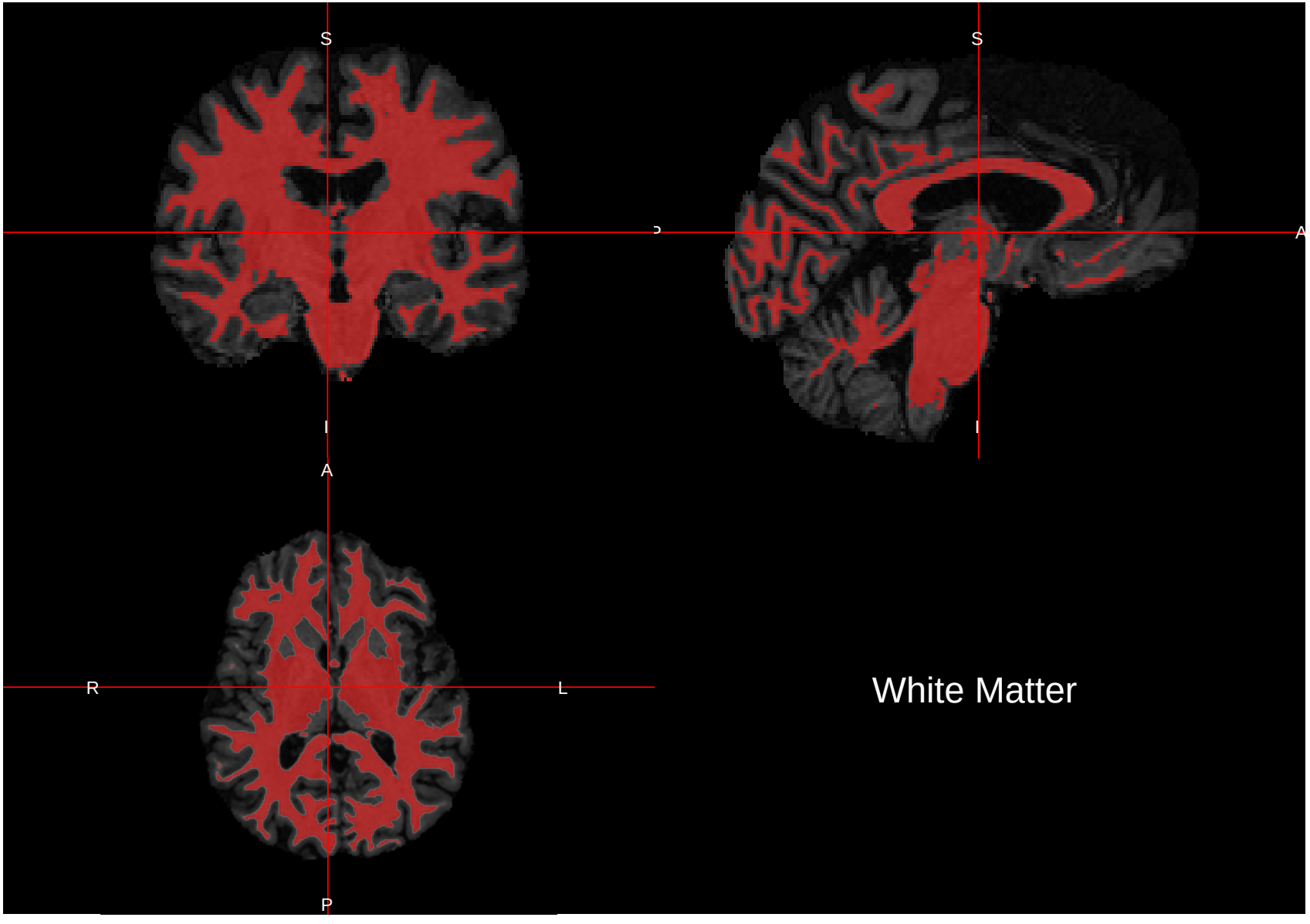
- Overall the results do not look good
  - We will use `robust_window`
- The k-means clustering is affected by large outliers

Tissue Segmentation using ANTsR, extrantsr

```
robust_t1_otropos = otropos(a = rb, x = mask) # using robus
robust_t1seg = robust_t1_otropos$segmentation
```

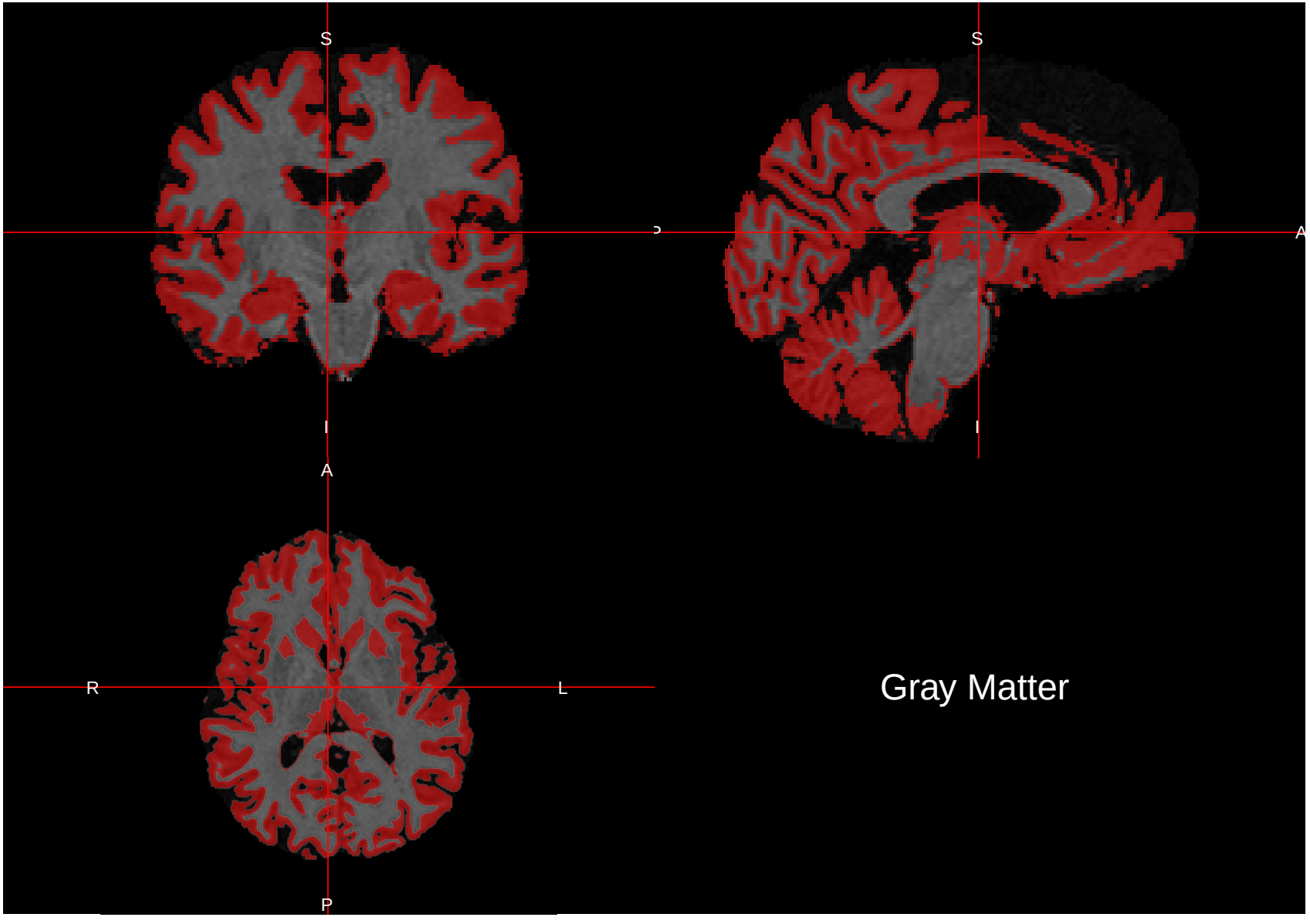
White Matter

```
ortho2(t1, robust_t1seg == 3, col.y = alpha("red", 0.5), te
```



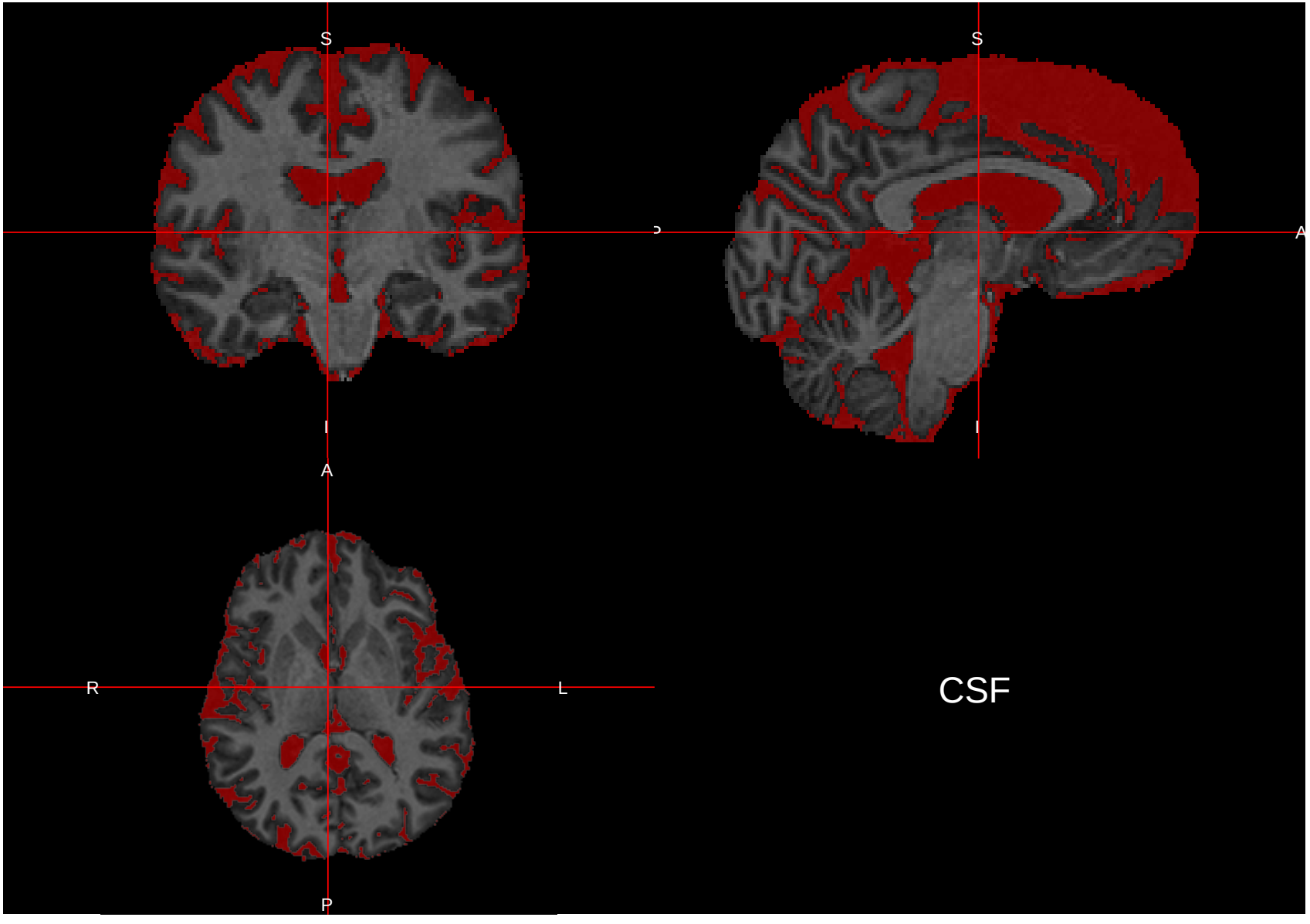
Gray Matter

```
ortho2(t1, robust_t1seg == 2, col.y = alpha("red", 0.5), te
```



CSF

```
ortho2(t1, robust_t1seg == 1, col.y = alpha("red", 0.5), te
```



Robust Atropos Results

- Overall the results look like they reasonably separate the classes
  - No ground truth
  - Better than running Atropos on the raw data
- The k-means clustering can be aided by Winsorizing these large outliers

Estimating the Volume of Each Class

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

```
tab_fsl = table(t1seg[ t1seg != 0])
tab_fsl
```

```
     1      2      3
883978 640960    768
```

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." *Neuroinformatics* 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." *Medical Imaging, IEEE Transactions on* 20 (1): 45–57. `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=906424`.