# Computational MRI Worksheet 0

**Gary Hui Zhang, PhD**

## Working with 3-D medical images

The objective of this worksheet is to help you become familiar with 3-D medical images, understanding in particular

- how they are represented digitally
- how they can be visualised

## Dataset

We will make use of the BrainWeb simulated brain MRI database, which can be accessed by visiting the URL: https://brainweb.bic.mni.mcgill.ca

The website provides simulated brain MRI data of a great variety, including different contrasts, different spatial resolutions, different levels of noise, and more. Most interestingly, it offers data from both healthy individuals to subjects suffering from multiple sclerosis (MS), a particularly kind of neurological disorder that MRI plays a significant role in helping neurologists to come to a firm diagnosis.

## Task 1

Viewing volumes with ITK-SNAP (http://www.itksnap.org/pmwiki/pmwiki.php).

**Objective:** Through this task, you will develop a good understanding of how to represent a 3-D volume digitally.

1. Go to the BrainWeb URL above, then click the **Normal Brain Database** link.
2. Choose Modality **T1**, Slice thickness **1mm**, Noise **0%**, Intensity non-uniformity **0%**, then click the **Download** button.
3. Select File Format **raw byte** and Compression **none**, then click **Start download** button.

This should give you a file called **t1_icbm_normal_1mm_pn0_rf0.rawb**.

4. Go to the ITK-SNAP URL above, then follow the link for Downloads to download the binary release appropriate for your operating system, and then install the software.
5. Launch ITK-SNAP, then go to the menu bar and select **File**->**Open Main Image …**
6. In the **Open Image** dialog window, use the **Browse** button to locate the file above, then in the **File Format** dropdown menu, select the option **Raw Binary**, then click the **Next** button.
7. In the next page, you will need to provide **Image Header Specification**. The fields to complete here are **Image Dimensions** (the number of pixels along each image dimension) and **Voxel Spacing** (the physical size of a voxel along each image dimension).

These can be found at the top of the webpage where the file was downloaded. In this case, the image dimensions are 181, 217, 181 respectively. This should give you the correct **implied file size** which must be equal to the **actual file size**. Finally, the voxel spacing is 1 for all the dimensions.

8. When you click the **Next** button again, you will see three images appear in the ITK-SNAP main window, with the Open Image dialog summarising the important information about the file being displayed.

You should observe that not all the information in the Open Image dialog has been provided by you. In particular, you will likely know little about the **Orientation** property which has taken a value **RAI**. This property is meant to indicate how the image dimensions relate to the physical dimensions, which each of its three letters correspond to each of the three image dimensions. The code **RAI** in particular means that 1) the first dimension of the image, going through its smallest index to its largest, correspond to going through the rightmost voxel to the leftmost; 2) the second dimension the most anterior (front) voxel to the most posterior (back); 3) the third dimension the most inferior (bottom) voxel to the most superior (top).

In this case, this information was not explicitly specified but instead guessed, and the guess turns out to be wrong. For anyone with some knowledge of brain anatomy, it will be apparent, from looking at the images rendered, that the correct orientation code should be **RPI**.

9. To correct the orientation, we apply a **Reorient** operation, which can be initiated by following the menu option: **Tools**->**Reorient Image …**. Then set the **New Orientation** as RPI and click **Apply**.

In the Main window, you will see three orthogonal slices through the volume, which is the typical way to visualise a 3-D volume. The three slices will intersect at a single point, which is marked out by the crosshairs in each sub-window. By clicking at a different point in any of the windows, you can move the intersection point around, which in turn allows you to choose different slices to visualise.

**Task 2**

Writing your own Matlab/Python code to view volumes.

**Objective:** Through this task, you will learn how to read and write binary files, and viewing images, in Matlab.

Building on the code we have developed during the lecture, turn it into an **ImageViewer** function that can create 3-D orthogonal views for any input volume:

```
>> ImageViewer('t1_icbm_normal_1mm_pn0_rf0.rawb', …)
```

In the above function call, I have intentionally omitted the other necessary arguments. Think carefully what these missing arguments are.