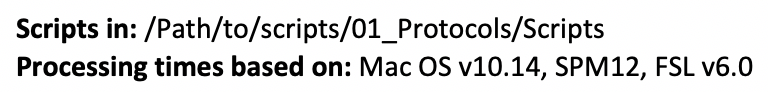
**Project Organization and Storage**

This documentation covers how to effectively organize your project during and after processing and how to effectively and efficiently store all of your project files after the processing phase is complete.

**It is vital to remember that someone else – your PI, a new RS, etc. – will one day need to access your project folder and understand what you did.** With that in mind, there are certain best practices that should be followed for every project:

1. **Documentation.**
   1. The steps to recreate what you did to process your data need to be written down.
   2. See the example pipeline document here: **/01\_Basics/File\_Organization\_Example/01\_Protocols/Project\_Pipeline\_Doc\_Example.docx**
      1. Note the following details to help to create a fully fleshed out, comprehensible document that will help a newcomer to the project understand and recreate what you did:
      2. A close up of a logo

         Description automatically generated
         1. Your name and the date should always be present.
      3. 
         1. The location of your scripts is helpful and the exact software you use (including version #s) is necessary.
      4. A screenshot of a cell phone

         Description automatically generated
         1. Each step of the pipeline succinctly includes all info necessary:
            1. The script used to perform the step.
            2. A brief description of what each script/step accomplishes.
            3. The time each script will take to run for each subject (where applicable/possible).
            4. Any notes about script adjustments that will be necessary to make it run correctly.
            5. The location of file outputs.
      5. A screenshot of a cell phone

         Description automatically generated
         1. Flowcharts can be extremely helpful, especially if your processing pipeline is particularly complicated.
         2. It is possible to make flowcharts in PowerPoint or Word, as well as free online resources such as lucidchart.com
      6. A picture containing star

         Description automatically generated
         1. It is also helpful to show screen shots of what your outputs should look like.
            1. There is no guarantee that the person looking over/trying to recreate your pipeline will be familiar with any given processing set.
   3. **The most important takeaway about documentation** (other than the fact that you should document your processing to begin with) **is that you should err on the side of more information**.
      1. The more info you can provide other people about your processing sets, the better.
2. **Comprehensible File Organization**
   1. Your folder organization and file/folder naming conventions should do a lot of informative work on their own. This is done in two main ways:
   2. **Folder naming/organization**
      1. It is a good idea to store outputs of each discrete processing step in a new folder.
      2. If you look at the File\_Organization\_Example folder structure, you can see that the subfolder naming convention tells us a lot before we even look at the files. We can tell where our raw data is stored (**02\_Data/01\_Raw\_Data**) as well as where we will find our registered images (**03\_Analysis/02\_Image\_Registration**), etc.
      3. The folder naming convention of “##\_[Processing Step]” tells us the order in which each step was performed.
   3. **File naming/organization**
      1. File names are also important for conveying information at a glance.
      2. A file name such as “Subj1\_bet\_0.4.nii” tells us that this image has been skull stripped at the threshold of 0.4. If this file were stored in the folder /03\_Analysis/01\_BET, we would know that it was done during the first step of analysis right after preprocessing, based on folder structure.
      3. When possible, avoid possibly confusing terms such as “final” in your file names. Try to stick to descriptive additions.
3. **Post-Project Storage**
   1. After your processing is complete, you will need to store all of your files, scripts, and notes. It is necessary to put some thought in to this storage process in order to make accessing the files possible without too much trouble in the future, and also use repository/archive space efficiently.
   2. **Remove extraneous files**.
      1. If you have duplicate files in your project folder or if you have files in your project folder that are stored elsewhere, consider deleting these files from your project folder before archiving/storing it.
      2. For instance, you may have preprocessed scalar maps or raw dicoms both in your project folder as well as in another location. These would be good candidates for removal before long-term storage.
   3. **Compress as much as possible**.
      1. Once you have gotten rid of files that you do not need, it is vital to compress the files you will need to store.
      2. This may involve compressing entire folders, individual files, or both.
      3. **Note**: Do not compress already compressed files.
      4. For example: If you have a folder of compressed niftis (with the file extension .nii.gz), do not then compress the entire folder.

**Further notes on data storage:**

**Places to store data**:

1. Box or another cloud storage platform. On these platforms, space is less of an issue, but it is still good practice to delete extraneous files and compress all other files.
2. Your personal repository or archive. In this case, storage is up to you, but the above best practices are still recommended.
3. Your PI’s research repository space or archive. In this case, good storage practices are the most vital as every PI has their own preferences and space is at a premium on places like the /fs/research/ drives and other external drives used for long-term storage.

**Methods of compressing data**:

1. Right click a file or folder and select “Compress”. Only use if you need to compress an entire folder of unzipped files or an individual file.
2. Compress via the Terminal.
   1. Use the command `gzip` as it is the fasted and most space saving bash command for zipping.
   2. Example usage:
      1. gzip -r /path/to/folder
      2. “-r” stands for “recursive”. This will search through all folders and sub folders to zip only files that are unzipped.

**Accessing compressed data**:

1. It is generally recommended to unzip files via command line but using a command similar to the one you used to zip the files: gunzip -r /path/to/folder