Basic guidelines

The following are some suggested guidelines for handling your data. Please feel free to contact us if you have any questions about how to implement these.

Backing up your data

- Back up your most important data (usually those related to a publication) to multiple locations, e.g. an external hard drive, the cloud, or the university file storage service.
- **For microscope images, you should save the files in its native format.** For example, since most of the microscopes in the facility are Nikon, you should save the files in the ND2 format. We suggest this because the native format preserves both the *raw image* (the actual intensity values measured by the camera) and the *metadata* (which is additional information, such as which optical configurations were used).
  - An alternative is to save the images in **uncompressed TIFF format**. However, doing so will usually result in some loss of information (usually the metadata).
- Whether or not you should save a copy of the processed data depends on whether the data was generated using a script or if there was manual modifications. If the data can be generated by running a script, it is usually not necessary to save it since it's reproducible. However, if you made any manual modifications, you should save a copy of the original and the edited version.

Documenting your analysis

- **Treat your image analysis protocols as you would your wet lab protocols.** This means writing down each step in your lab notebook. Pay particular attention to include any post-processing (e.g., cropping images, trimming movies), as well as all the functions and settings that you used in a step-by-step manner.

Managing code

- If you are using a script to analyze your images (e.g., MATLAB or Python), you should use version control software and publish the code you've used in a repository online. There are plenty of good and free tools for this: For version control, git has become the de facto standard. For online repositories, use either GitHub or GitLab, both of which provide free accounts.
  - Your online repository should include information about how to get your code up and running, including specifying any dependencies and basic operating instructions.
  - If your code is dependent on the script that someone else wrote, you should not modify the original files directly. Rather, you should write new scripts that calls the original functions if possible.
  - BIT often run workshops teaching people how to use Git.

- Document your code as you go by adding comments. You don't need to explain every line, but you should explain what a block of code is intended to do.
- Refactor your code. Refactoring is the process of improving code, without necessarily introducing new functionality. Think of it as revising a draft - the first version of code is often messy and overly complicated. Refactoring allows you to tidy things up, revise the data structures and in general, make it easier to work with the code.
  - Refactoring should be done as you go. Don't wait until the whole code is written before going back - it will likely be too big and unwieldy to work with.
  - Yes, this means "taking some time off" your experiment. In reality, refactoring will allow you to reuse your code more easily in the future and will likely lead to much easier maintenance.
  - Some common issues with code can be found in this Wikipedia article.

Revision #3
Created 28 July 2023 21:33:47 by Jian Tay
Updated 8 August 2023 22:16:43 by Jian Tay