Using a Personal Folder on the Scratch

1. First, some motivation: If using the scratch directory as above works perfectly, why would you bother making a personal directory on there? There are a couple of reasons.
   1. Video files and data sets can be huge files. There is only about 120 GB of storage allowed per user within /home, which isn’t very many data sets’ worth. Putting them on the scratch doesn’t count towards your personal storage, so you can put essentially any amount of data on there that you want.
   2. If you run jobs in parallel, the job number changes between the code that creates the distributed job and the functions that it runs. This was mentioned in step 4.4. Because the method used above to copy to and access the scratch creates a folder identified by job number that is deleted after that job number ends, we wouldn’t be able to use the distributed computing if this were the only way to use the scratch.
      1. A .serial file that will result in this error is included in the codes linked to this step. It is called “j\_scratch\_fail.serial”. Run that code and type “bjobs” at various times while it is running. Once it has finished running, check “Task1.out.mat” in the “Job1” folder in the folder chosen as your Job Storage Location. You will see the error that the file could not be loaded. If you edit the serial file to call load\_scratch2 with $MYSANSCRATCH and the matrix size as inputs, the function will work fine.
   3. The solution to both of these issues is to create a folder within /sanscratch identified by your name.
      1. Always remember that the folders on the scratch are temporary folders – even your personal one might get accidentally deleted and there is no automatic backup – so never put anything on there that isn’t backed up elsewhere.
2. When using your own /sanscratch directory, there is no need to define the MYSANSCRATCH variable in your .serial file, nor is there a need to define how much space needs to be allocated. All you need to do is change into that directory and copy over anything that isn’t already in there.
3. There are three working .serial files in the folder attached to this step: h\_scratch1.serial, h\_scratch2.serial, and h\_scratch3.serial.
   1. The first copies files and matrices over to the home scratch directory and then calls a function that loads a particular matrix, takes its transpose, and then saves it to the home directory.
   2. The second starts out the same, but differs because it calls a distributed computing function.
   3. The third requires that you first copy the files and functions over to your scratch directory before you run the .serial file; it does not include lines for copying those files over. If you put video files or large data sets on the scratch with the intention of analyzing them, this is the model that you should follow.
      1. If you are copying large files onto the cluster, you should not do it by dragging from one folder to another in the ftp client like you may have been doing for smaller files. When large files are copied in this way, the cluster has the potential to get overloaded and crash.
      2. To learn a safe way to copy large files from a Windows computer, email Henk and ask him.
      3. To copy large files from a Mac, open up Terminal but do not log into the cluster. Instead, use the following command: “rsync –vac --bwlimit=5000 [file location on your computer] petaltail:/sanscratch/[folder]”. This allows the cluster to control how quickly the data is being uploaded, avoiding overloads and crashes.

Written by Brendan Cole, July 2015.