7-26-17

Processing Images for MAP-Mapping

**All scripts needed are in in this directory**

**Just copy all scripts (.py and .slurm and .m) to your folder at the beginning.**

**Stitch LSM files after imaging**

(if you happen to mistakenly save your files as czi, I have some similar scripts in the directory /n/schier\_lab2/users/sthyme/oldczipythonfiles/). See bottom of the file for some notes on running with czi files.

Modify fastqc\_stitching.slurm to have the correct directory where your images are (blue), the correct numbers of images (the line array=, not shown here), and the prefix of the files in the folder (red).

xvfb-run -a /n/schierfs2/projects/ImageRegistration/Fiji.app/ImageJ-linux64 -macro /n/schierfs2/projects/ImageRegistration/Fiji.app/scripts/cmdline\_lsmStitchAndPrepareCZIsForWarping\_cluster.m "atxn7\_2017\_07\_04\_\_20\_49\_15\_\_p$SLURM\_ARRAY\_TASK\_ID.lsm /n/regal/schier\_lab/sthyme/atx7/ 486.4"

The tricky part here is that lsm files are named p001 or p01 depending on whether you generated >100 files or >10. This is not dealt with in the script, so you have to run 3 scripts for every run (of >100, so 2 scripts if you have 10-99 files). Ie, the prefix will say \_p0 for array set 10-99 or \_p00 for array set 1-9, and \_p for >99

You can use the script stitching\_queuecheck.py to generate a list of ones that failed (not all computers on the cluster will produce output, unfortunately). To use this script you have to modify the range line ( fullset = range(1,294) ) at the top with 1+finalfilenumber (ie, 294 if you have 293 lsm files).

After it’s finished, I would also clean up directory by making a folder called lsmfiles and moving all your original lsm files here. You probably do not need to keep these original images, since they are captured in the stitched images, but I’d check to be sure stitching worked before deleting them.

Change the file in a text editor, such as Vi.

Using Vi:

Change the directories and the numbers and names of files:

Note: Useful commands for vi mode

vi <name of file or script > : enter vi mode

To edit text, hit “I”

To exit text-edit mode, hit “esc”

Type “:” to move cursor to bottom and execute commands

w = save

q = quit

q! = force quit without saving

Run script via “sbatch scriptname”

Will’s old stitching notes below, not sure about modification for nomenclature

1) ImageJ script “StitchAndPointwiseLSMsPreWarping.ijm.” Should end up with .nrrd files for each channel per fish. (Note: I updated the original macro to streamline nomenclature, and retitled the file).

2) Upload stitched files onto server for next steps.

Note: to avoid nomenclature problems down the line, I modified the ImageJ script to append “<desired prefix>\_”

**Warping Image Files**

1) Make a directory called “images” (mkdir images). Registration software does not accept other names.

2) Move stitched nrrd files and “filetodirectory\_newstitch.py” into “images” (mv \*nrrd etc.).

3) From within “images”, run “filetodirectory\_newstitch.py” by typing ./filetodirectory\_newstitch.py

If not executable, try typing chmod +x filetodirectory.py

4) Open “fastqc.slurm” in vi mode. Change the line with the "--array" option to = whatever number of directories are in the images directory you have (ie, 1-80 if you are doing 80 fish)

5) From directory just above “images,” run fastqc.slurm file by typing “sbatch fastqc.slurm”. This may take a while. Check status of jobs using the “squeue” command (ex: squeue –l –u williamjoo). You should end up with directories called “reformatted.#” for each fish. (reformatted.X/X/.nrrd).

Troubleshooting: 1) Are you in the right directory 2) Nomenclature: If your directories included a prefix like “fish,” you might have to edit the filetodirectory script to remove it. For example, see filetodirectory\_fish in the schierfs1 data/example directory.

6) Some files may not process correctly on the first run. In this case, delete the Registration.# files for any of your missing numbers, update the fastqc.slurm file “array” line, and rerun fastqc. You do not need to delete the folders by hand if you run the script withpqueuecheck\_betterupdatednorange.py (but you must wait until all jobs are done running to use this, as it doesn’t deal with running jobs anymore). The script will output a list of the numbers of directories for the images that failed registration without completion, make a new fastqc.slurm file where you replace the value after the --array option with this list of numbers (replace 1-80 with the new list).

Note: Be careful using this script – I would open it up and comment on the following line by adding a # sign before it (that’s a comment in python), make sure it gives expected results, and then uncomment line and rerun. This is for safety in case there is something off with your file naming and then the script would delete all the registered data, not just the ones that didn’t finish.

7) Move all the warped .nrrd files into a new folder for smoothing. (mv ./reformatted\*/\*/\*nrrd “new directory”). For all analyses that you want to also do warping for, please name this directory “smoothedtiffs”, so that it is compatible with downstream scripts that will make your life easier.

**Smoothing Image Files**

This is very similar to stitching files, in that you need to deal with the \_p0 situation and run the script 3x. Go into the directory smoothedtiffs and then modify and run the following script fastqc\_smoothing.slurm. Once finished, make a directory called onlysmoothedtiffs and move just the smoothed images to this directory.

**Making MAP Maps**

1. Get a list of your genotyping data, either by using the splitgenotypes\_doubleoption\_addvalue.py on a matrix of data in a 96-well plate format, as in the example matrix file or by making your own list. Label the matrix file as follows, with the gene name you want, underscore, then word “matrix” (snap91hethomxhomhet\_matrix). We generate the matrix from a google sheet and just copy and paste it into the directory, so I would do something like that to be sure the formatting is the same. You must fill empty wells in the 96-well plate with something like “e”. This matrix method is a life-saver for duplicated genes, because I built in a method to take two side-by-side matrices and generate the duplicated genotypes. We generate the file by just typing in the hom and het designations will looking at the gel, makes it less work than figuring out the numbers of each well. The addvalue line is for when you run multiple genes in the same microscopy run. Put the last number of the gene before into that line (ie put 0 for gene 1 if it runs 1-48, and then put 48 for gene 2). In the end, your list of genotypes needs to be in a file called “genotyping”. In this file there are lists of genotypes. You can generate those lists by hand and just start from step 2 skipping the matrix generation of the list, or you can run the script and it will automatically generate the file called genotyping for you.
2. Decide what genotypes you want to compare and add an asterisk to the beginning of these lines in your genotyping file. Just do this for one gene at a time in the file. The matrix method does not generate genotypes like “hetandwt”, so you can combine the “het” and “wt” lists if you want to use “hetandwt” as well. Please use the designation “hetandwt” exactly, as these genotype designations are used later for renaming files to go on the website. Mapmapping produces both comparisons, but one comparison has the colors in the right direction and “hetandwt”, “hom”, “wt”, etc are used to determine which comparison you want automatically. If you want a genotyping designation other than the following list, you will need to add it to the script for moving files to website (wt, wtwt, hetwt, wthet, wtandhet, hetandwt, hethet, het, homwt, wthom, hethom, homhet, hom, homhom). This script is called renamefilesforwebsite.py.
3. Run the script splitfilesbygenotypecomma\_setupfiles\_bothlsmandczi.py
4. Make sure it did what you wanted, and if it did, just type ./jobsubmission.sh and all jobs will be submitted. Make sure you have the “MaketheMAPMap” files in this directory because jobs won’t run without them.

**To analyze morphology by comparing amount of warping to standard image:**

Keep reformatted.x folders from before.

Keep your Registration folders!

1. Unzip all the registration.gc files:

gunzip /n/regal/schier\_lab/WJ\_Mapping/Registration.\*/warp/\*/\*/registration.gz

2. Run warpingfastqc.slurm. Change array, folder paths to match your reformatted + registration files. Then move all jacobian files to a smoothing folder. I call this directory “warpingsmoothed”.

usually: mv reformatted.\*/jacobian\* <destination folder>

Note: if you accidentally deleted the reformatted.x files, re-create them beforehand using mkdir name.{1..x}

3. Smooth using warpingfastqc\_smoothing.slurm (has different smoothing parameters than those used for pERK signal). As before, fastqc must be copied into smoothing folder first. As before, move smoothed files to a new “onlysmoothedtiffs” directory.

4. If you have run the regular pErk mapmapping as above with the script generating the folders and used the suggested directory names, you can run morphology analysis very easily. Just run the script newduplicatedirectoriesandscripts.py and then you should have copied over all the tErk images you need and identified all comparisons you want to make and all genotyping information. Then you just need to run the new jobsubmission.sh file generated from this script.

**Putting maps on website:**

1. Make a directory where you want the comparisons you are interested in to go. Then go into the onlysmoothedtiffs directory that contains the mapmapping data you want to put on the website.
2. Modify the two lines with a hardcoded directory in the renamefilesforwebsite.py (right now it is) /n/schierfs2/projects/ImageRegistration/data/sthyme/mapfilesforwebsite/
3. Run the script renamefilesforwebsite.py as follows ./renamefilesforwebsite.py -run 1st -xtype hethomfxhomhetm -gene ep300 where the “gene” flag input should match the beginning of your output directories containing the mapmapped data. The xtype is what I use for extra information, but you could change that to some other type of information if you wanted. For information like a stimulus, I just typically use a different gene name for the entire run and use a - to designate (ie, mir137-heat is what I use for the matrix file and that gets propagated by the scripts). The “run” is so that I know if this is my repeat 1 or so on for that gene, but it could also be used for different information in the naming, if you wanted.
4. Then you need to resize the image. This can be done on your own computer with Fiji or with my script. My scripts have hardcoded directories in them sometimes, so you will maybe want to copy your own and hardcode your own directories. Run it on an interactive node as follows: /n/schierfs2/projects/ImageRegistration/newFiji.app/ImageJ-linux64 -macro /n/schierfs2/projects/ImageRegistration/newFiji.app/scripts/ttransformjrgb621x140.ijm
5. Then convert files to RGB (sadly cannot do both these things at once in a cluster script). Again, there are hardcoded directories here. xvfb-run -a /n/schierfs2/projects/ImageRegistration/newFiji.app/ImageJ-linux64 -macro /n/schierfs2/projects/ImageRegistration/newFiji.app/scripts/rgbtransformjrgb621x140.ijm
6. You might have to remove the stub file first, but Fiji will let you know.
7. Then run opencv to remove background of map with the script convert\_tiff\_map\_to\_pngs.py. This script has a hardcoded directory called “maps”, but you can also change that in the code. To use opencv, you must load a module for it by typing module load opencv/3.0.0-fasrc04 in your interactive session. This script generates directories of png files that will go onto the schier lab webserver.
8. Move directories of png files into the webserver. /n/schierfs2/schiermedia/sthyme\_schizophreniamicroscopy/img/maps If you use a different directory it will be more annoying to put the files into the website admin because you’ll need to change the base server name, so I’d just put them here.
9. Change permissions on your maps (you could try doing it before moving probably, so that it’s easier to designate them all with a \*, haven’t tried it, but not a big deal if you just have a few). Type chmod -R 755 directroynameyouwanttochange
10. Add maps to your admin on the website. Fairly self-explanatory on the website itself.

NOTES FOR CZI FILES. Hopefully never needed again . . . .

As of 2/24/17: New system and software. Autosave doesn’t work for large number of samples? Must use Streaming function, but can only save as czi files.

Numbering convention: Streaming spits out files with numbers attached in name(#).czi format. There will be one file (the first one) without a number. Name this name(0).czi.

In fastqc\_newczistiching.slurm: rename output file in last line as appropriate. I copied this into my folder with all the czi files and run from inside there.

Set range as only # of fish (# of head files if split by position + tile). Ex: files (0)-(209), set range as 0-105 because 105 fish.

Can use newczi\_stitching\_queuecheck.py if files drop out during stitching step.

First time I ran this: lots of drop outs! Had to re-run many times to get all the files.

2) Make a directory called “images” (mkdir images). Registration software does not accept other names.

3) Move stitched nrrd files and “newczifiletodirectory.py” into “images” (mv \*nrrd etc.).

4) From within “images”, run “newczi\_filetodirectory.py”

This next step is same as before?

5) Open “fastqc.slurm” in vi mode. Change the line with the "--array" option to = whatever number of directories are in the images directory you have (ie, 1-80 if you are doing 80 fish)

Note: Useful commands for vi mode

vi <name of file or script > : enter vi mode

To edit text, hit “I”

To exit text-edit mode, hit “esc”

Type “:” to move cursor to bottom and execute commands

w = save

q = quit

q! = force quit without saving

6) Warping: From directory just above “images,” run fastqc.slurm file by typing “sbatch fastqc.slurm”. This may take a while. Check status of jobs using the “squeue” command (ex: squeue –l –u williamjoo). You should end up with directories called “reformatted.#” for each fish. (reformatted.X/X/.nrrd).

6) Warping: From directory just above “images,” run fastqc.slurm file by typing “sbatch fastqc.slurm”. This may take a while. Check status of jobs using the “squeue” command (ex: squeue –l –u williamjoo). You should end up with directories called “reformatted.#” for each fish. (reformatted.X/X/.nrrd).

Can use newczi\_regqueuecheck?

fullset = range(1,213)

to be whatever the number of images you have + 1 (ie, if you had 212 files, then make it 213)

Doesn’t work now?

Moved all the warped files to a new folder for smoothing. (mv ./reformatted\*/\*/\*nrrd “new directory”)

Smoothing step: used fastqc\_smoothing\_newczi.slurm.

This next step is the same:

1) Make folders for each comparison you want to make with MAPMapping (ie, het, hom, hetandwt) and copy your smoothed files into it using the script splitfilesbygenotype.py (you need to copy this script into the smoothed folder first, and run it from within that directory).

./splitfilesbygenotypecomma.py -ids "a,b,c,d" -dir "folder,” where a,b,c,d = individual fish, and “folder” = condition or genotype folder you created. If IDs are separated by spaces instead of commas, use “splitfilesbygenotype.py”

SAVE THESE NUMBER LISTS FOR JACOBIAN ANALYSES LATER

Troubleshooting: Are your IDs named properly? If the smoothed tiff files have a prefix before the number, like fishX, include this when you type in the IDs.

Notes: I need to come up with a way to automatically generate the IDs. Manually typing them gets tedious.

2) Make output directories for your mapping results.