General instructions for data files and types iPyrad can run are found [here](https://ipyrad.readthedocs.io/en/master/4-data.html); parameters are found [here](https://ipyrad.readthedocs.io/en/master/6-params.html); and information about iPyrad’s seven steps can be found [here](https://ipyrad.readthedocs.io/en/master/7-outline.html). There’s also a great walkthrough for running iPyrad on a cluster [here](https://radcamp.github.io/AF-Biota/02_ipyrad_partI_CLI.html).

Log on to TACC.

Navigate to your scratch directory.

Download the Day3\_iPyrad\_worksheet\_files directly from Anne’s TACC to your scratch directory. This may take a few minutes as it’s quite large.

scp -r eac3496@frontera.tacc.utexas.edu:/scratch1/03123/eac3496/training\_course/Day3\_ipyrad\_worksheet\_files .

## Installing iPyrad software

**Anne will guide you through these steps**

Install iPyrad. First, follow the directions found under “Linux install instructions for Conda” [here](https://ipyrad.readthedocs.io/en/master/3-installation.html):

*# Fetch the miniconda installer with wget*

wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86\_64.sh

*# Install miniconda into $HOME/miniconda3*

*# \* Type 'yes' to agree to the license*

*# \* Press Enter to use the default install directory*

*# \* Type 'yes' to initialize the conda install*

bash Miniconda3-latest-Linux-x86\_64.sh

*# Refresh your terminal session to see conda*

bash

*# test that conda is installed. Will print info about your conda install.*

conda info

Now, follow the instructions under “Conda install”. This line of code is telling conda to retrieve iPyrad:

conda install ipyrad -c conda-forge -c bioconda

Check to see if iPyrad is working by typing the following:

ipyrad –h

If that doesn’t work, enter the following and then try the above line again.

module unload python3

What does the iPyrad help (ipyrad -h) output look like (this may take a few seconds to run)?

## Make a new project in iPyrad

Generate a new parameters file for your data with the project name ranaddrad.

Now, check your current directory to see if a params file was actually made. Look at that file’s contents now.

Modify the following parameters in your newly created parameters file in iPyrad:

[2]: make the raw fastq path the path to the two .fastq files (i.e., where you currently are)

[3] add the path to the barcodes file (with the name of the barcodes file) here.

[7]: the data are paired ddRAD data, but **we’re only going to use the forward reads** for the sake of saving time. If you wanted to use both reads, you’d put pairddrad here and iPyrad automatically detects R1 and R2 in the file names.

[8]: the restriction overhang is CATGC, (make sure you keep the comma).

[14]: clustering threshold should be set to 0.91.

[21]: should be set to 4 by default.

[27]: in addition to the default output file types, we also want iPyrad to also spit out a usnps file (u), a nexus file (n), a structure file (k), and a vcf file (v).

# Run steps 1 and 2 in a development node

Start a development node for two hours.

Before doing any of the steps in iPyrad, you’ll need to run the following line again. Do so now.

module unload python3

Using the sequence data, the barcodes file, and your newly created params file, within the idev node, run the first ***two*** steps of iPyrad.

Once this is complete, check your output files for this run. What do they look like? What formats do they have?

# Run steps 3–7 as a job

Steps 3 and 6 are both clustering steps and are very computationally intensive, so we can’t run these steps within an idev node but will have to submit them as a job.

Copy the skeletonslurm file you made on Day 2 into your current directory and save it as ranaddrad\_s3-7.slurm. Note that this file name specifies both the data that will be run and the steps that TACC will be running. This helps us stay organized.

In the appropriate part of this file, add the line of code you to unload python3 (module unload python3), followed by the line of code that you want TACC to run to specify running iPyrad steps 3–7 for your data and parameters file.

Now, before we submit this as a job, let’s just make sure that everything is in place to run.

Submit your slurm file as a job to TACC.

Check the status of the job. This will take a while to actually run in TACC.

So that you can work on this in your own time, download the relevant data files (and final output from step 7; **ranaddrad\_output\_files.zip**) from the **data** directory on GitHub.

# Once you’ve finished running iPyrad

TACC likely hasn’t had time to finish running through all of iPyrad, but you hopefully understand the general gist of it. To expedite our subsequent analyses, download the relevant output files from iPyrad (**Day3\_iPyrad\_output\_files.zip**), available in the **data** directory on GitHub. *These will also be the files you’ll use to build trees in RAxML-ng.* Take a look at the structure of this directory. Keep in mind that the stats files *do not* have accompanying interim data files; I’ve done this just to simplify things.

Move into the copied folder (Day3\_iPyrad\_output\_files).

Make a *new* folder called ranaddrad\_output\_files and move *all* files within your copied folder into this directory.

Copy your parameters file into the ranaddrad\_output\_files directory.

How many output files are there (excluding the params file)?

How do these output files correspond to parameter [27] in the iPyrad parameters file?

Take a look at the final stats file (ranaddrad\_stats.txt).

How many loci are there in the final assembly? (*two places in the stats file state this number)*

How many SNPs are there in the final assembly?

How many parsimony-informative sites (pis) are there in the final assembly?

Which sample has the highest number of ***consensus*** reads? How many are there? Is this the same sample as the one with the highest number of loci?

Which sample has the lowest number of ***consensus*** reads? How many are there? Is this the same sample as the one with the lowest number of loci?

Which stats file contains information on the depth of coverage for each of your samples?

Which stats file contains information on the heterozygosity estimates for your samples?