A Fun Introductory Command Line Exercise: Next Generation Sequencing Quality Analysis with Emoji!

*Unfortunately, this lesson only works on computers running Mac OSX or Linux 🙁 😱 🔀





*this activity was adapted from code and slides developed by Andrew Lonsdale (@LonsBio) at Melbourne University. <u>Here's a link</u> to a Lightning Talk that Andrew gave in 2017 about FASTQE.

Goals:

- Use basic command line programming to:
 - Introduce students to writing basic command line scripts
 - Analyze and assess the quality of FASTQ formatted short read NGS data
 - Trim/filter low quality reads in FASTQ files

The 1st step of any Next Generation Sequencing (NGS) analysis pipeline is checking the quality of the raw sequencing reads in each FASTQ formatted file. If the sequence quality is poor, then your resulting downstream analysis will be inaccurate and misleading. FastQC is a popular software used to provide an overview of basic quality metrics for NGS data. In this lesson, you will use an even more universal form of communication to analyze FASTQ files, THE EMOJI 🗱 👯 🚾.

Technical requirements/limitations:

- Unfortunately, Windows does not support the use of emoticons (2) (3) so this lesson only works on computers running Mac OSX or Linux
- Also, you need to have Anaconda installed on your computer (see pre class assignment https://bit.ly/2RxKApp). Anaconda is a Python-based data processing & scientific computing platform with built in third-party libraries. Installing Anaconda is equivalent to automatically installing Python and some commonly used libraries on your computer, including ones you will need for this activity
- Lastly, the FASTQE program is limited to short read NGS data of 500bp or less

Like FastQC, **FASTQE** can be used to analyze the quality of any type of FASTQ file whether it's from a genome sequencing project, an RNA-seq project, a ChIP-seq project, etc. Here's brief background on the in class metagenomics project that we're collecting sequencing data for. Garter snakes excrete sexually dimorphic pheromones to attract a mate. The hypothesis of our experiment is that male and female garter snakes host unique microbial communities in their mouths, cloacae and musk glands that contribute to sexually dimorphic bioengineering of these pheromone molecules. **Figure 1** provides an overview of our 16S metagenomics analysis pipeline. For this lesson though, all you need are the FASTQ files. Feel free to substitute your own favorite FASTQ files for this activity if you like.

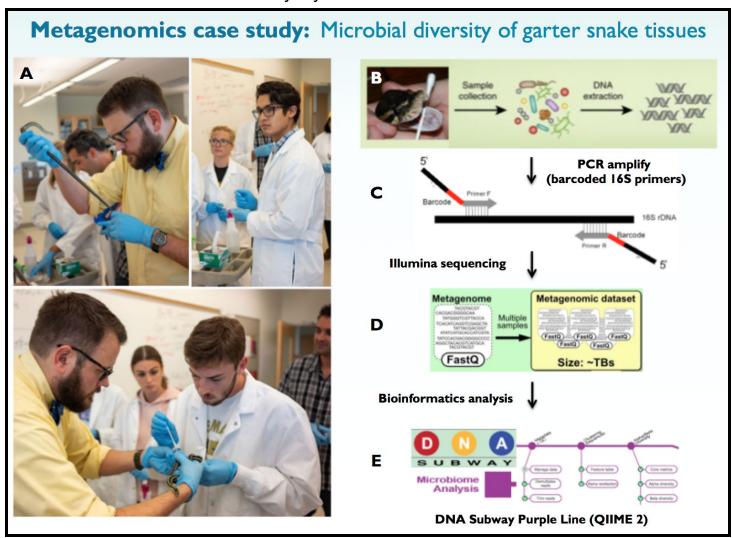


Figure 1. Overview of the in class metagenomics project. Using a saline swabbing technique, microbial samples were collected from garter snake tissues in class **(A)**. Swabs were placed in sterile tubes to release collected microbes and DNA was extracted for downstream analysis **(B)**. Barcoded primers were used to PCR amplify the microbial 16S ribosomal DNA repeat genes for each sample followed by Illumina sequencing of PCR amplicons **(C-D)**. The DNA Subway Purple Line web-based

software can be used to analyze FASTQ data files generated from Illumina sequencing to reveal the microbial population of our swabs **(E)**. Garter snakes were provided by Dr. Rocky Parker in the JMU Department of Biology (**A**; yellow shirt).

As previously discussed, FASTQE is a program that will analyze a FASTQ file and read out an emoji output as an indicator of each sequence's quality in the file. So a high quality read may look like this $\cupe{\cupebox{0.5}{look}}$, while this symbol $\cupebox{0.5{look}}$...well you get the idea.

• Open up your computer's **Terminal** or **command line**

When using command line code, lines that start with the # sign are text descriptors or instructions, not lines of code. For example:

- #This is the command used to navigate to your desktop using the change directory command (hit enter)
 - Then type: cd Desktop (hit enter)
- Here's what that looks like in Terminal

```
[Rays-Mac:~ rayenke$ #This is the command used to navigate to your desktop
[Rays-Mac:~ rayenke$ cd Desktop
Rays-Mac:Desktop rayenke$ ■
```

- The # character is used to indicate comments. A line starting with # line doesn't do anything, in this case it's just telling you what the next line of code does which is navigate to your computer's desktop
- We will stick with this convention throughout the activity
- Comments can occur anywhere in a line, anything after the # will be ignored
- #check to see if you are still on your desktop using the print working directory command
 - o Type: pwd
 - Question 1: If you've printed a path that doesn't make sense (i.e. the directory you navigated to is the incorrect directory) how would you go back to the previous directory? (hint, it includes the *change directory* command)
- Move the 2 compressed fastq files available with this lesson to your desktop (they have .gz extension and need to be unzipped; they also available here for download https://bit.ly/2FbODRS)
- #Unzip both files using the gunzip command

- #gunzip followed by a fastq file name will unzip and individual file
- #or gunzip followed by *.gz will unzip all files ending with .gz
 - o Type: gunzip *.fastq.gz
 - Question 2a: What's the purpose of using a zipped file?
 - Question 2b: Would it make sense to use the *.gz command if you only need a single zipped file in your directory?
 - Question 3: Would you use gunzip if you had a fastq file?
- #check to see if your files are unzipped using the *list* command
 - Type: ls
 - Attach a screenshot of your terminal after using the ls command
 - Question 4: Are there any files that you know you've installed in this directory missing when you used the ls command?
- #Install the fastqe software using the *install* command
 - Type: pip install fastqe
- Note: if you don't have Anaconda installed at this point you will probably get an error message here
 - Question 5: Why is Anaconda necessary for pip to run?
- #now you're ready to run fastqe; you can use individual .fastq file names or type
 *.fastq to run all .fastq files in your directory
 - o Type: fastqe *.fastq
 - Question 6: If you were to type fastqe *.fastqgz into the terminal, would you expect an error? Why or why not?
 - Question 7: What are the advantages and disadvantages to using the command fastqe *.fastq rather than fastqe sample.fastq?
- take a screenshot of your Emoji output for each file & paste it into an MS ppt file
 - It should look something like this:



Notice that 1 of your files (femal_oral2) seems to have lower quality than the other based on the Emoji readout. Let's use another program called **Fastp** to get a more conventional readout of the .fastq file data. **Fastp** is similar to the **FastQC** program we previously used, however, it also has a trimming tool to cut out or **filtering** the low quality sequences in our file.

- Type the command pwd to determine if the current directory is sufficient for your new directory
- #Create a new folder on the desktop called fastp using the make directory command
- Type: mkdir fastp
- Drag your 2 fastq files into this folder
- #navigate to the fastp folder
 - o Type: cd fastp
- #Install fastp
 - Type: conda install -c bioconda fastp
- #run fastp on the lower quality female_oral2.fastq file
 - o Type: fastp -i female_oral2.fastq -o
 out.female oral2.fastq

- You should now have 3 new files in your fastp folder
 - 1. A .html file (this is your QC report)
 - 2. A .json file (ignore this for now)
 - 3. A trimmed fastq file (out.female_oral2.fastq)
- Open your html QC report & collect some before and after filtering data
 - Question 8: How many reads are in this FASTQ file before and after filtering?
 - Copy/paste before and after Quality plots into your ppt file
 - Question 9: How do they compare?
 - This is the per base quality plot with quality (or Phred score) on the y and base position on the x axis
 - Lastly, let's rerun fastqe to get an emoji output on our trimmed file
- #use the fastp output .fastq file to rerun fastqe
- Type: fastqe out.female_oral2.fastq
 - Question 10: How does your before and after filtering emoji outputs look for this file?
 - Question 11: If we were to trim the data a certain way to get a good quality output, would that mean that we would have to trim all data in the study with the same parameters? Why or why not?
 - Question 12: Which tool (fastqe or fastp) did you find easier to use?
 - Question 13: Which tool (fastqe or fastp) do you think is more a more reliable research grade tool?

To sum up, you just looked at Illumina FASTQ data quality using Emoji output. You then filtered the low quality sequences in your FASTQ file and output before and after QC plots. Following filtering, you created a new Emoji output of your filter FASTQ file.

You did all of that on the command line, congrats!

Here's the script only version of the above exercise:

#download the 2 fastq.gz files from Canvas onto your desktop #navigate to your desktop directory

cd Desktop

#Unzip both fastq files using the gunzip command

gunzip *.fastq.gz

#check to see if your files are unzipped (.fastq extension)

ls

#Install the fastge software

pip install fastge

#run fastge on all .fastg files on the desktop

fastqe *.fastq

#Create a new desktop folder called fastp using the mkdir command

mkdir fastp

#Drag your 2 fastq files into this folder

#navigate to the fastp folder

cd fastp

#Install fastp

conda install -c bioconda fastp

#run fastp on the lower quality female_oral2.fastq file

fastp -i female_oral2.fastq -o out.female_oral2.fastq

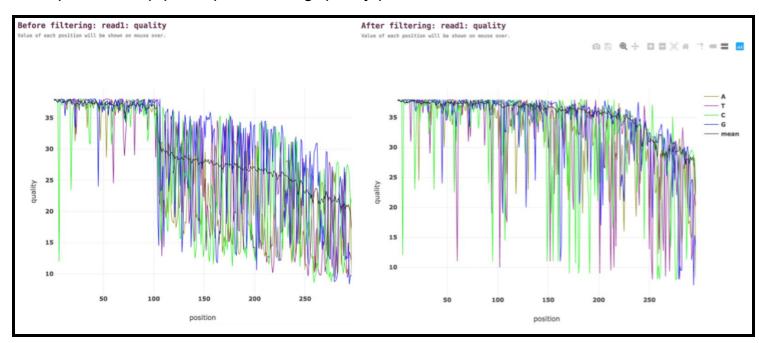
#use the fastp output .fastq file to rerun fastqe

fastqe out.female_oral2.fastq

Examples of fastp pre & post filtering metrics:

Before filtering	
total reads:	244
total bases:	72.224000 K
Q20 bases:	53.889000 K (74.613702%)
Q30 bases:	49.129000 K (68.023095%)
GC content:	47.599136%
GC content: After filtering	47.599136%
	150
After filtering	
After filtering total reads:	150
After filtering total reads: total bases:	150 44.400000 K

Examples of fastp pre & post filtering quality plots:



Optional Extended Learning Question #1:

Bioinformatics is a science just like any other field in biology. This activity has taught you how to use these bioinformatic tools to analyze genomic data, but if you just work through the text and save all of the files on your computer, you may find that you forget what these files are for later on! We suggest that you do this question to help you learn how to document your work using command line tools.

BQ- use the command line to make a text file that documents all of your work. i.e.

- What programs were used
- Why those programs were used
- What files were generated
- What the files contained
- A brief discussion about the work done

Hints- Remember to save your file regularly while working, as hiccups happen!

- Make sure all of these files are in the same folder
- If your files are not all in the same folder, make a folder with a descriptive name and move all of the files related to this activity to the same folder.

Optional Extended Learning Question #2:

Commands such as the ones we have used today (gunzip, pwd, fastp, fastqe etc) can often be used in multiple ways by adding additional arguments (ie additional lines of code). The **documentation** for each program will usually explain more about what it does as well as additional argument commands that can be used. A typical convention in Unix/Linux is for programs to provide this documentation from the program itself with a -h or --help option (for example, by typing fastqe -h to view fastqe documentation)

A. View the documentation for both fastqe and fastp. Which program has more options?

B. Given that fastqe normally calculates the mean phred score for each base position in a fastq file and outputs a corresponding emoji, explain what your output represents when you add the additional argument fastqe female_oral2.fastq --min to your command. Also provide a screenshot of your emoji output with this command.

C. Finally, we've been looking at emoji outputs for DNA sequence quality, but what exactly do and mean? Add the --scale argument to determine the Phred score associated with each emoji output in the female musk2.fastq file.

Answer Key:

Command line cheat sheet: https://www.git-tower.com/blog/command-line-cheat-sheet/

Question 1: If you have printed a path that doesn't make sense, i.e. the directory you navigated to is the wrong directory, how would you go back to the previous directory? (hint, it includes the *change directory* command)

Answer 1: the command cd .. is used to go back to the previous directory, or up a folder

Question 2a: What is the purpose of using a zipped vs. an unzipped file?

Answer 2: NGS data is very large, using a zipped data file allows for easy transfer of data between machines and relatively small usage of disk storage space.

Question 2b: If your computer doesn't have a lot of room for file storage, which file type would you need?

Answer 2b: To save space on your computer, keep your files that you aren't using as a.gz, or zipped file

Question 2c: Would it make sense to use *.gz if you only need a single .gz file in your directory?

Answer 2: Potentially. If you are going to use the other files that will be unzipped, then yes, it does make sense to unzip them all. However, if you only need one file to be unzipped, it would make more sense to unzip that single file.

Question 3: Would you use gunzip if you had a fastq file?

Answer 3: No, as that fastq file is not zipped. It is ready to use.

Question 4: Are there any files that you know you have installed in this directory missing when you used the *Is* command?

Answer 4: This is individual. Some students will have "hidden files" that they **know** are in that directory. If they run into this problem, have them use the code ls -a, which shows hidden files.

Question 5: Why is Anaconda necessary for pip to run?

Answer 5: Anaconda imports many different packages, including python3 and pip. If you tried to run pip without it installed (using Anaconda) you would get an error.

Question 6: If you were to type fastqe *.fastgz into the terminal, would you expect an error?

Answer 6: Yes, Fastqe takes fastq files almost exclusively.

Question 7: What are the advantages and disadvantages to using the command fastqe *.fastq rather than fastqe sample.fastq?

Answer 7: Fastqe can't examine every single fastq file, it needs to examine one file at a time

Question 8: How many reads are in this FASTQ file before and after filtering?

Answer 8: Answer varies depending on the files used, however, each student should have the same answer at the end of the assignment, and the "after" filtering should be less than the before.

Question 9: How do they compare?

Answer 9: Same as above, answer should be the same if all students are using the same sample files. However the answers should be less than the original file, but the average read quality should be higher across the read

Question 10: How does your before and after filtering emoji outputs look for this file? Answer 10: Each student should have the same answer, but the answers are unique to the files used for processing. There should be significantly less poo emojis present in the output after filtering.

Question 11: If we were to trim the data a certain way to get a good quality output, would that mean that we would have to trim all data in the study with the same parameters? Why or why not?

Answer 11: This is a question that is up for debate! There are those who think that for the sake of reproducibility, that each file should be treated the same way (same trimming and filtering parameters) regardless of outcome. There are also people who think that the trimming and filtering parameters should be unique to each file, to produce the "best" output. This question is not really meant to be "right" or "wrong", but to get the students thinking about the scientific process and how to apply that to their research.

Question 12: Which tool (fastge or fastp) is more efficient to use?

Answer 12: For people who have never touched a command line before, both programs are relatively easy to use. However, your students might think that fastqe is easier to interact with. This may be due to the data output being "easy" to interpret. However, this may not make the tool more "useful" in a scientific sense.

Answer 13: fastp of course would be the more reliable program in terms of project scalability and producing publishable data output. **How much credibility could you give a paper with a poo emojis in their results section?** Have fun with this one.

Optional Extended Learning Question #1 Answer:

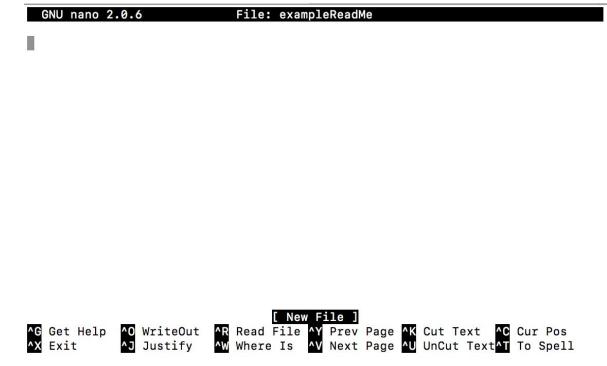
 If some of your students didn't put all of their files in the same directory, have them run the command

```
mkdir practiceFastqeQuestion
```

- Of course, they can name this folder anything that they would like to. I
 would recommend that they use camelCase, just to get into a habit of
 using it (a lot of bioinformaticians and computer programmers use
 either camelCase or snake_case)
- Once the folder is made, they can move files from their current directory to the new folder by using the command

```
mv forgotToMoveThisFile.docx
practiceFastqeQuestion
```

- Replace "forgotToMoveThisFile.docx" with the appropriate file names and "practiceFastqeQuestion" with the appropriate folder name
- Have your students navigate to the practiceFastqeQuestion folder
 by typing the command cd practiceFastqeQuestion
- Once in this folder, they can type the command nano exampleReadMe
- This uses the command line text editor nano, and works much like any other text editor. The potential actions that can be taken in the editor are located at the bottom of the screen



- I can type anything I want to in this editor box. Have your students use this as a "lab notebook" for their bioinformatic experiments. **Due to the nature of this project, what you might find to be necessary for a lab notebook may be different from what we would consider necessary.** We have provided some guidance in the original question. Of course, if you find that this is simply not enough information, you can ask your students to write more.
- To save the file as is, simply hit control O

GNU nano 2.0.6 File: exampleReadMe Modified

I have written a small amount of text to show how the editor works.

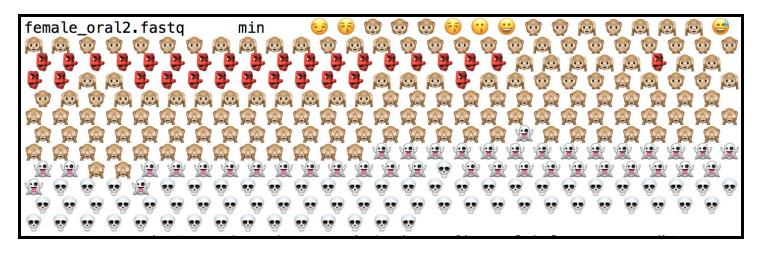
If you would like to know more about what you can do with the text editor nano, look at the bottom of the screen for options. I will now hit control O to save the text.



- nano gives you one last chance to edit the file name. If you are happy with the file name, hit enter.
- To exit the text editor, hit control X
- Once this question/assignment is complete, have them send you a screenshot of their directory (ls), and then a screenshot of their text editor
 - o Or, alternatively, have them send you the file

Optional Extended Learning Question #2 Answer:

- A. fastp --help will return way more options than fastqe
- B. the —min argument outputs show the emoji associated with the minimum quality base per position in the fastq file



C. Phred=10; 90% accurate; A Phred=20; 99% accurate; Phred=30; 99.9% accurate

Time of lesson: 30-45 minutes

Potential Tags:

- Bioinformatics
- Genomics
- Command line
- Terminal

- Linux
- Mac
- Fastqe
- Fastp
- FastQC
- Microbiome
- FASTQ
- Next Generation Sequencing (NGS)
- Emoji
- NIBLSE
- Undergraduates
- Illumina
- Phred Score
- Trimming
- Quality control
- Short read
- Introductory