Small Group Exercise #6

Mapping

Part 1 --- Mapping reads to a reference with Bowtie2.

1. Return to the ASC documentation system https://hpcdocs.asc.edu/ and explore the README info for the mapping software **Bowtie2**. This file includes various problems the system administrators have noticed and multiple examples of how to run the software.

    i. Go to https://hpcdocs.asc.edu/ and log in with your ASC credentials.
    ii. Click on "Sequence alignment", then "Bowtie2".

2. In the README module, you can see the queue script normally required to run a real Bowtie2 job. An example script named `Bowtie2_example.sh` is located at: `/home/shared/biobootcamp/data/example_ASC_queue_scripts`.

3. Make a copy of the script to your home directory.

    i. `cd` (to take you to the top level of your home directory)
    ii. `mkdir mapping_working_example` (create a directory to work in)
    iii. `cd mapping_working_example` (to change into the directory)
    iv. `cp /home/shared/biobootcamp/data/example_ASC_queue_scripts/Bowtie2_example.sh .` (to copy the script to your working directory; note the "." at the end of this command, specifying the current directory)

4. Now use a text editor to open/view the script:

    i. e.g. `nano Bowtie2_example.sh`
    ii. Note that both the indexing and mapping steps are included and that comments in the script indicate where you will be adding your commands once you decide what you need to do.

5. You will need raw FASTQ files of paired-end Illumina reads copied to your working directory. Copy them to your working directory from `/home/shared/biobootcamp/data/Lamellibrachia_luymesi_sequence_reads_for_assembly/`. The two files are named `Lamellibrachia_luymesi_genomic_L001_R1_001.fastq` and `Lamellibrachia_luymesi_genomic_L001_R2_001.fastq`. Can you think of an easy way to check if these files represent a properly paired set of reads, using basic command line utilities?

6. Next, you will need a reference sequence to map the reads to. For this exercise, you will be using the potentially complete mitochondrial genome from *Lamellibrachia luymesi* that you identified in Exercise #3 on Tuesday morning. Make a copy of that file to your current working directory.

7. You now should have everything you need in the current working directory: the `Bowtie2_example.sh` queue script, two `*.fastq` files of reads derived from genomic DNA and the `*.fasta file` containing the potentially complete mitochondrial genome to serve as a reference. Check to be sure by "long listing" the current directory.

8. Now it's time to add the required information to the queue script. You will need to include the names of the `*.fasta` and `*.fastq` files as well as what you want the "basename" for the index files and output SAM file to be. A suggestion for the basename is: `potential_mito_genome_Lamellibrachia_luymesi`. Another suggestion is to write down the names of each file on paper for reference when you are adding them to the appropriate place in the `Bowtie2_example.sh` queue script. Once this is done, save the script, note the run conditions for the ASC queue at the bottom of the script and exit from nano.

9. Now submit the job to the ASC queue system and monitor the job using `squeue`.

10. Once complete (it should take 5-10 min), list the current working directory and note the many new files that have been created. All files ending in `*.bt2` represent the index created by Bowtie2-build and all are required for the index to be valid and functional. Keep this in mind if you construct Bowtie2 (or other mapper indices) in one location and then move it/them to another. You should also see your output SAM file. Take note of the size of this file by running:

    `l –h *.sam` (SAM files can be huge, hence the need for a compressed, binary version: BAM files).

11. Before we move on, the last thing to do is determine just how many reads mapped to our reference. This information can be found by tailing the log report from the run using: `tail -29 Bowtie2example.*`. The overall alignment rate will be in the middle of the output, just above the first line of hashes (i.e., ####).
    Check with other members of your group regarding their overall mapping rates.
    How do people feel about the overall mapping rate? Why might it be what it is?

Part 2 --- Mapping reads to a reference with BWA.

1. For fun, we will compare our Bowtie2 results to that of another popular aligner, the Burrows-Wheeler Aligner (BWA), from the makers of Samtools.

2. Explore the documentation for BWA on ASC.

3. From the ascdoc above, there is no need to create a queue script for BWA. However, take note that you will need to load it into your workspace by using the "module load" command. Back in your terminal session, load the BWA module.

4. To see the usage information for BWA and begin utilizing the software, run `bwa` after loading the module.
   Note from the output that, similar to genome tools (gt, which you used for assembly assessment), or Samtools, the usage of BWA is `bwa <subcommand> [options]` ; so to see the options for creating an index run: `bwa index` . You can see that the minimal information to provide bwa is the name of the FASTA file to serve as the reference. Thus, now run `bwa index` with the name of the file that you will use for the *reference* (i.e., the same one used previously for Bowtie2).

5. Once the above is complete, do a `ls -alt` and note that there are five new files in the current directory with 2-3 letters as file extensions. All of these represent the index created by BWA and are required for the index to be valid and functional (just like BLAST+ and Bowtie2). **INFO**: BWA and Bowtie2 use their own index formats that are NOT interchangeable.

6. Now let's see how we can use BWA to map against this index.

   i. Run `bwa` without any arguments. From reading the end of the usage message, we are suggested to try `bwa mem` as a starting point. Let's see what options are available for that.
   ii. From reading the "Usage:" statement, we will minimally have to supply the name of the index to use as well as the two (2) `*.fastq` files of reads to be mapped. Also note that since there is no explicit option regarding the output, we should make the assumption that it would be directed to the screen (which we don't want). So a redirect (i.e., >) will need to be appended to the end of the command to a file we name with a .sam extension (keep track of this). Lastly, the `-t` can be added to use more than one CPU. So add a `-t 2` to your command to take advantage of two CPUs. Draft out your command on paper or text editor and when ready, enter it into Terminal and execute.

7. Once the mapping completes, note the size of the resulting SAM file and compare it to the one you got from Bowtie2. Are they similar in size? Why do you think so?

8. Unlike Bowtie2, BWA does not provide a summary of the mapping at the end of a run. But this information can be obtained using a separate program called **BAM**. Note that like BWA, "bam" expects a [options] structure. Since we want basic stats for the SAM file we generated from BWA, the command would be: `bam stats --basic --in <YOUR_FILE_FROM_BWA>.sam` . From the output, identify your MappingRate(%) and compare that to what you obtained from

Bowtie2. How different are the values and what might be the reason for this? Discuss among the group.

On your own ... There are many additional uses for your mapping data, Samtools and other software. Try filtering your map file (during BAM creation step) according to particular values of the `-f` and `-F` flags. For example, could you send all of your unmapped reads to a separate file for analysis? What would be the point of that? Also take a look at the powerful utilities available with Picard, GATK (demo'd later this week), and Bamtools.