**Homework 8**

**Instructions.** This homework is due at the beginning of lab, Wednesday, November 1st. When executing the commands of this homework on the hpc-class cluster, please use the salloc or sbatch commands. We may get in trouble if we just type the commands at the command prompt of the head node (the node you log onto). In addition, every command in this lab/homework is “real” bioinformatics, meaning they take real time and things can go wrong. (I myself have had to do every step about three times to get it to this point. Do not assume it should just work. Only thourougly pre-chewed, roughage-removed, watered-down homeworks just work, and this one is not one of those.)

1. Your first step is to download some data from the [Short Read Archive.](https://www.ncbi.nlm.nih.gov/sra)
   1. Navigate to the website and enter the experiment accession SRX026594. What machine was used to generate these sequence data? When were the data generated? Note the run accession(s) associated with this experiment.

**(Illumina Genome Analyzer II)**

* 1. It is no longer possible to download the data using the web interface. Instead, NCBI provides the [sratoolkit](https://www.ncbi.nlm.nih.gov/sra/docs/toolkitsoft/) command-line tools. These tools are installed on hpc-class. Make them available by issuing the command module load sratoolkit. Read this helpful blog about [paired-end reads,](http://thegenomefactory.blogspot.com/2013/08/paired-end-read-confusion-library.html) the help for the fastq-dump command, and explain what the --split-files command-line option does.

**--split-files indicates the number of files you want to divide the data into. Dump each read into separate file. Files will receive suffix corresponding to read number. Split-3 files would do the same and include a singleton file.**

* 1. Using the run accession you noted in Part a, issue a fastq-dump command that uses option --split-files and compresses the file using gzip technology. Since our hpc-class home directories are limited to 5Gb of storage and this is big data, you may encounter your disk quota while attempting this lab/homework. You can create a personal directory in /ptmp and store the files there. You will need to check the fastq-dump help to get it to dump the data in this directory. Please copy and paste the command you used below.

**fastq-dump /ptmp/bcbio444hw8/ fastq --gzip --split-files SRR065390**

* 1. Since these data are quite old, we should investigate the quality score encoding, which has changed over the years. The [Wikipedia](https://en.wikipedia.org/wiki/FASTQ_format#NCBI_Sequence_Read_Archive) entry claims that SRA has converted all quality scores to the Sanger standard, but there is no citation provided for this claim, and I have only found confirmatory rumors.
     1. On the same Wikipedia page, there is a helpful diagram of the different ASCII quality score encodings. Can you use this diagram to assess whether the quality scores are in standard Sanger format? Explain how you came to your conclusion.

**We can determine if it is in sanger format if it has ASCII characters before 64, since Illumina format only uses ASCII values from 59 (or 64 depending on the version) or greater. The data we saw had a lot of exclamation points (which are a quality score of 0 and only appear in the anger format), so it appears the data is in sanger format.**

* + 1. In the same **Encoding** section of the page, it states that Illumina, for some platforms, used Phred scores 0 through 2 to communicate something other than probability of error. Can you determine if these data are subject to this special encoding? Explain how you came to your conclusion.

**The data is not subject to the special encoding. If it were, bad data would be indicated by long chains of ASCII character B (which replaces chains of low scores). In the data, however, there are many chains of exclamation points, which would not only be replaced by B characters in Illumina 1.6, but does not occur in Illumina encoding**

1. The data you downloaded are reads of the roundworm *C. elegans*. [WormBase](http://caltech.wormbase.org/virtualworm/) is a website repository of everything *C. elegans*, including the reference genome. Navigate to the public ftp site and drill down to *C. elegans* genome for the reference strain Bristol N2. (PRJNA# accession numbers identify BioProjects, which you can look up and learn more about in [NCBI’s BioProject Database.](https://www.ncbi.nlm.nih.gov/bioproject)) Once you find the file you need, download it to hpc-class using the Linux command wget. How many chromosomes does *C. elegans* have? How long are each of the chromosomes in the current version of the reference genome? (**Hint**: remember BioPython)

**There were 6 chromosomes (I, II, III, IV, V, X), with lengths, 15072434, 15279421, 13783801, 17493829, 20924180, and 17718942 respectively. The program I used also found a chromosome called MtDNA with a length of 13794, but I assumed that this was data for some other structure that the program mistakenly grouped with the other 6 (given the difference in size and naming convention).**

1. We will align the reads to the reference genome using the software bwa, which becomes available if you type module load bwa.
   1. The first step is to build an index for the reference genome. You do this with the bwa index gzipped\_fasta\_file, where gzipped\_fasta\_file is the reference genome you downloaded in Part **??**. How much disk space does the index occupy compared to the gzipped fasta file?

**The bwt (burrows-wheeler transform) file that it generates is 96 Mb, while the original fa.gz file is 28 Mb (though when unzipped, on my local machine, the fasta file is about 100 Mb).**

* 1. The next step is to align the reads to the reference genome. This step can take a long time, but the hpc-class cluster has 16 processor cores per node that can be utilized to speed up bwa. We will use the bwa mem command to align the reads. Read the help (just type bwa mem) to find out how to request 16 threads and provide the index and fastq files to bwa. Also, you should use the -M flag in order for the results to be compatible with other steps in this Lab/Homework. Note, the output is a sam file that is delivered to stdout: you should redirect it to a file ending in sam. Read about the [SAM format](https://genome.sph.umich.edu/wiki/SAM) and [hard vs. soft-clipping.](https://www.biostars.org/p/109333/) Record the command you used below, randomly select one aligned entry in the SAM file with at least one mismatch and show its local alignment with the reference genome below.

**salloc bwa mem -M -t 16 c\_elegans.PRJNA13758.current.genomic.fa.gz.bwt SRR065390\_1.fastq.gz SRR065390\_2.fastq.gz > aln.sam**

TGAANACCTCGAAACTTTTTCAGCGGNNTCNTTNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN

ATNNNCCNNNTTCACNNCTGCTNGCCNANNNNNNNNATAATTTTTTTACTTGGAAATTGTTTTAGCATCTGCAAAAAGTATTTATTCANNNNNNNNNNNN

* 1. Now we need to convert the SAM file to a binary BAM file. Issue these commands (please note some can be threaded):

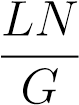
samtools view -b samfile > bamfile # convert SAM to BAM samtools sort -o sorted\_bamfile bamfile # sort BAM file (along genome) samtools faidx reference\_fasta\_file # create an index of the reference genome samtools index sorted\_bamfile # create an index of the BAM file

To view the result, you can use samtools tview sorted\_bam\_file. (Note: You may prefer using the [Integrative Genomics Viewer (IGV),](http://software.broadinstitute.org/software/igv/) but this graphical interface is not available on the hpc-class cluster. You may be able to install it on your local computer. To use it, you need all the same files that samtools tview needs.) Wander around until you find evidence of a sequencing error. Take a screenshot of the evidence and explain why you believe it to be a sequencing error.



**Of the sequences listed, this is the first to have nucleotides in the 14th through 19th positions.**

1. This question is about read depth.
   1. The coverage of a sequencing experiment is the expected number of reads covering each genome position under the assumption of random fragmentation. Ignoring the effects of chromosome ends, explain why coverage can be estimated as

*,*

where *L* is the read length, *N* is the number of reads, and *G* is the genome length. What is this estimated value for the *C. elegans* data? What is the predicted probability that a particular base is not covered by any read? (Show your work.)

**Suppose we have a genome of size 6, with 3 reads, each of length 4. The chance of reading any position in the genome is 4/6 for each read. With three reads, the chance of reading any position becomes 3 \* 4/6 = 3.**

**In general, for any read, the chance of reading any position is L/G. multiplying by the number of reads, N, gives us the expected value of LN/G.**

* 1. The actual coverage will be lower because of read errors, clipping, and alignment problems. We can estimate the average coverage per base using samtools with the command samtools depth sorted\_bam\_file combined with awk. Report the result.

**I used the command:**

**salloc samtools depth sorted\_align.bam | awk '{sum += $3; n++} END {print sum/n;}'**

**the result was 63.0468**

1. This question is about variant discovery.
   1. In this section, we will look for Single Nucleotide Variants (SNVs).
      1. Typically during library preparation, there are amplification steps (before the bridge amplification that occurs on the Illumina flow cell) where single molecules are copied many times. As a result, the same original molecule can be represented by multiple reads. Many pipelines remove the duplicated reads. Why are datasets deduplicated (dedupped)? Why would it *not* be a good idea to remove duplicate reads in amplicon sequencing of a metagenomic sample? ii. You can deduplicate with the samtools rmdup. How many reads were removed during deduplication?

iii. Use samtools mpileup to create a .bcf file, which is a binary version of the

[Variant Call Format (vcf).](https://en.wikipedia.org/wiki/Variant_Call_Format) You can convert to .vcf format using bcftools view (available with module load bcftools) to see the data in text format. Select one interesting variant and explain why it is listed as a variant. iv. Finally, you can call SNVs that are likely to be true variation in the sequenced individual relative to the reference genome using bcftools call. Output the results in .vcf format, so you can easily work with the results. How many SNVs do you call? Which calling method did you use? Why is it appropriate?

1. In this final part, you will compare the SNV calls between two methods.
   1. Repeat Part iv with the other calling method. **Bonus version I.** Repeat the analysis using the [GATK pipeline,](https://software.broadinstitute.org/gatk/best-practices/bp_3step.php?case=GermShortWGS) through the Variant Discovery step only. You can access the GATK tools with the command module load gatk. (**Hint:** You probably need to locate the necessary .jar files in the filesystem and explicitly name them in the GATK pipeline commands.) **Bonus version II.** Repeat the analysis after filtering the reads for lower quality (see module fastx-toolkit and consider that quality scores less than 30 are often considered of too low quality). **Bonus version III.** Repeat the analysis after using the bowtie2 aligner (module load bowtie2).
   2. The bedtools suite of software, available after you issue module load bedtool2, provides methods to compare .vcf files. Use bedtools intersect and bedtools subtract to prepare a Venn diagram showing how many SNVs the two methods both found and how many each uniquely found.