**Computer Assignment 8: Characterizing Genetic Variants from Next-generation Sequencing Data**

## For this assignment, please provide responses to all questions in a separate word document entitled “Ans8LastnameFirstname.docx” (or your preferred widely-supported office format), containing only your answers. Upload the file to Canvas by 11:59 p.m. CST the Friday following discussion.

The **1000 Genomes Project** is an international collaboration aimed at comprehensively detailing the extent of human genetic variation by providing full genome sequences of several thousand individuals of various ethnicities from around the world. Beyond cataloging human genetic variation, the data from this project has been used to infer human demography and evolutionary history, inform genome-wide association studies of risk-associated genetic variants, highlight patterns of linkage disequilibrium in the human genome, and improve the human reference genome sequence. For more information, please check

<https://www.internationalgenome.org>

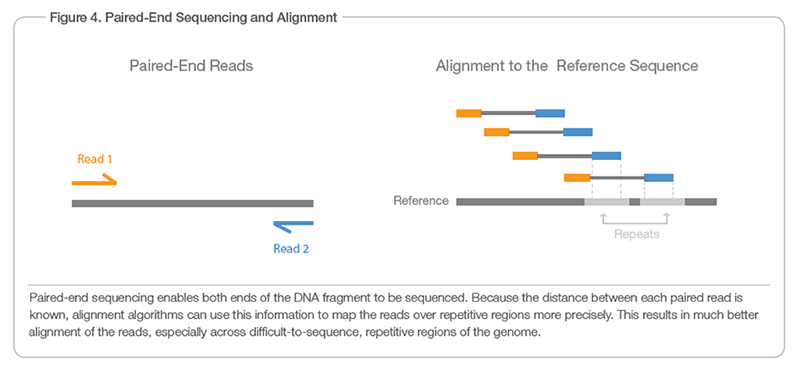
In today’s assignment, we are going to analyze a subset of next-generation sequencing (NGS) data from the 1000 Genomes Project. We will perform the analytical steps required to go from raw sequence reads of exome-captured DNA from an Illumina HiSeq 2000 instrument to annotation of identified genetic variants with potential roles in disease. These steps include:

* Understanding the contents of **raw NGS files**
* **Aligning** raw NGS reads to the human genome
* **Identifying** single nucleotide variant (**SNV**) and insertion-deletion (**indel**) mutations
* **Annotating** these variants with biologically pertinent information
* Exploring the annotated variants for their **potential roles in human disease**

**This assignment will serve as a primer for your final projects. Therefore, it is very important that you have sufficient understanding of the analytical process, both in terms of implementation as well as conceptual backdrop.**

**Part 1: The NGS data file**

We will begin our analysis with the list of raw sequence reads generated through the Illumina sequencing platform. In particular, we will analyze paired-end 75 bp sequence reads (shown below from [the Illumina website](https://www.illumina.com/science/technology/next-generation-sequencing/plan-experiments/paired-end-vs-single-read.html)) of an exon-enriched DNA library:



The raw sequence reads are already uploaded to our Midway projects directory. Just as you did last week, (1) login to Midway from either Terminal or Windows PowerShell and (2) check the contents of the project folder by using the following commands:

*ssh* [*CNET@midway2.rcc.uchicago.edu*](mailto:CNET@midway2.rcc.uchicago.edu)

The needed files are in following folder

*ls /project2/bios11141/CompAsn8*

Create a directory called Lab8 in your /scratch/midway2/CNET/ directory with the mkdir command and copy the following folders from the Lab8 directory to your directory:

data, geno-ready-bam, and scripts

**Make sure to name the directory exactly as it appears in this document**.To copy a folder and all of its contents, use the cp command we learned previously with the -r argument (for recursive, change CNET to your own CNET), for example

cp -r /project2/bios11141/CompAsn8/scripts /scratch/midway2/CNET/CompAsn8/

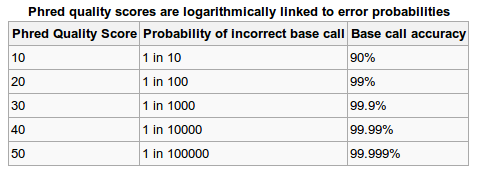
Repeat this command to copy the data and geno-ready-bam directories into your Lab8 directory.

Navigate to your newly copied data directory and execute the command “ls -lh”. You should see two files with the “*fastq*” extension. These *fastq* files contain the sequence reads and their quality information. Notice that these files are very large!

Every four lines in these fastq files provide information for a single read. Look at the first four lines (i.e. one read) of one of the *fastq* files by executing

head -4 <name of fastq file>

**1.1** Why are there two *fastq* files, one with the \_1.fastq extension and another with \_2.fastq? (Hint: this has to do with the way that reads are made in the NGS technique used here).

The first line provides information about the sequencing run and the specific read that follows. The second line contains the nucleotide sequence of the sequencing read. The fourth line has one character for each of the nucleotides in the sequence. It may not look like it, **but these characters are quality scores assigned to each base call.** They are not particularly human readable because they are encoded in a native computer format. This encoding scheme is used to calculate the “*Phred*” quality scores, which corresponds to the base calling error probabilities.

**1.2** Phred scores above 30 are generally considered trustworthy. However, 99.9% accuracy across millions of base pair calls still results in thousands of sequencing errors. How can we accurately identify variations in our genome (compared to the refence genome) and distinguish these from simple sequencing errors? (Hint: remember the highly parallelized approach of NGS.)

**Part 2: Alignment**

The first step we must take to utilize these sequencing data is to **align each individual read to the reference human genome**. This serves to identify which genomic region the sequencing read corresponds to; from this alignment, we can determine where deviations from the human reference genome occur. As we have seen with other alignment applications, alignment is computationally expensive. This is particularly important in genomic analyses, as we try to align millions of short nucleotide sequences to the 3 billion possible positions in the human genome. The **Burrows Wheeler alignment (BWA)** algorithm is the most commonly employed algorithm for mapping short sequence reads to genomic databases because of its consistency and speed. We will use the BWA algorithm to align our paired end sequencing reads to the human genome. This will proceed through two steps:

1. The bwa aln program aligns the reads from each paired end file to the human reference genome. This will produce two distinct files in the ***‘suffix array index’ (.sai)*** format; a way of representing reference alignments using an indexing strategy that is computationally efficient but unintelligible to us. Each ***.sai*** file contains distributions of possible alignment positions for individual ends of the paired end reads.
2. The bwa sampe and samtools programs merge the two ***.sai*** files to determine the best alignment position for each pair of paired end reads based on the genomic locations of the possible alignment positions from each individual read. This step produces a ***‘binary alignment map’ (.bam)*** that contains our final alignment for downstream analyses.

Finally, in order to facilitate genotyping and variant calling, we must sort our aligned sequences according to the genomic coordinates of the human genome to which the sequences are aligned. We will be using samtools sort to sort the ***.bam*** file.

We have provided you with the script: Bwa\_alignment\_paired-end.sh that will execute these three steps: alignment, merging and conversion to ***.bam*** format, and sorting. View the contents of this script with the *less* command, and browse the annotated comments delineated with the ‘#’ symbol. While we don’t expect you to understand all of the intricacies of the code, you should be able to intuit much of what is going on (to exit *less* type *q)*.

**2.1** What is the purpose of all of the ‘echo’ commands in the script? If you’re unsure, then consider the results of the command in your terminal: echo Hello World >> log.txt

**2.2** There are two flags specified for the bwa aln commands, and one flag for the bwa sampe command. Check the [BWA manual](http://www.htslib.org/doc/samtools.html) and describe what these parameters are specifying. (A quick Google search can often turn up more useful explanations than program documentation).

Many Unix commands have optional flags (sometimes called options) that control the general behavior of the command. You’ve already used flags before! For example, cp -r command copies files recursively. The flag in this case is ‘-r’.

In order to run Bwa\_alignment\_paired-end.sh on Midway’s compute node, we will have to modify and submit a job-submit script, just as we did for our R simulation last week. Open up Bwa\_alignment\_paired-end\_SUBMIT.sh with nano program and make the following modifications:

* Provide a reasonable job-name
* Provide the absolute path to your .*fastq* sequencing files (use *pwd* to get the path*)*
* Provide the absolute path to your Bwa\_alignment\_paired-end.sh script (use *pwd*)
* Make sure to change ‘CNET’ for each of the three times it appears

Now, create (*mkdir*) and change into (*cd*) a directory called *alignment* **within your CompAsn8 subdirectory**. We will submit our alignment submission script from this directory, and the scheduler will know to save our output files here as well. Submit your job with *sbatch* command as we did previously:

*sbatch ../scripts/Bwa\_alignment\_paired-end\_SUBMIT.sh*

Once this command is executed, be sure that it is either running or at least queued to run. Remember, you accomplished this in computer assignment 6 by using:

*squeue -u CNET*

If the job is already indicated as Complete, check the logfile (SRS003660.align.log) and the output files from the job submit script, as an error was likely encountered. The job will take about 20 minutes to finish running. You can proceed to the next section of the assignment, but when the run is completed, be sure to return to answer the following questions.

**2.3** Look into SRS003660.align.log with *less* command. About how long did the alignment of each *fastq* file take? How long did the *bwa sampe* process take to complete?

**2.4** List the names the files generated by this run. Which is the largest, and how large is this file? *(-lh* option of *ls* converts bytes to larger metric units).

As the alignment will take some time, the resulting sorted *.bam* file has been provided in the ***geno-ready-bam*** directory that you copied earlier. Before moving on, we want to assess the quality of our alignment. However, our alignment is encoded in an unintelligible format. (Check with the *head* command if you want to try to make sense of it yourself.) Let’s use a program from the ***samtools*** package, a suite of programs for interacting with high-throughput sequencing files. The ***flagstat*** output of ***samtools*** provides a useful summary of how well our sequencing reads aligned to our genome database. Execute the following command to assess the quality of your alignment, which load the ***samtools*** software installed in our project directory, tells the ***samtools*** package we want to use the ***flagstat*** program, and provides the sorted ***.bam*** file that we want alignment information on. Be patient, as the results might take a minute or so to appear.Please note that this is a one-line command with two separate parts.

*/project2/bios11141/CompAsn8/software/samtools/samtools flagstat /scratch/midway2/CNET/CompAsn8/geno-ready-bam/SRS003660.sorted.bam*

**2.5** How many sequences were processed in total from both paired-end reads? How many of these reads were rejected by quality control (QC)?

**2.6** What percentage of these reads mapped to the reference genome in your alignment process?

**2.7** Each short read in our NGS data has a mate that is taken from the same segment of DNA but in the opposite direction. We could reasonably expect, then, that any given read and its pair will map to the same chromosome. How many paired-end reads in total (irrespective of mapQ) did map to different chromosomes? Why might this occur?

**Part 3: Genotyping**

The next step is to identify single nucleotide polymorphisms (SNPs) and insertion-deletion events (indels) in our individual's genomic sequence reads, as compared to the reference human genome. This process is referred to as **variant calling**. The intuition behind this step is to leverage the repeated reads over identical nucleotide bases to determine the genotype of the individual at this position.

**3.1** For example, let’s say we have 36 reads that map to nucleotide position 104,204 of chromosome 10. 28 of these reads place an “A” base at this position, and 8 reads have a “T”.

**a)** Intuitively, would you conclude that the individual has a heterozygous AT genotype? Or would you conclude that this individual has a homozygous AA genotype, and the eight “T” reads are sequencing or alignment errors?

**b)** How would the knowledge that the sequencing or alignment quality scores on the reads that produced the eight “T” calls are very low affect your confidence in your conclusion?

**c)** If the individual is truly AT heterozygous, how many of the reads (on average) would you expect to indicate a T base at this position?

Statistically, this intuition is leveraged in a likelihood framework. We can calculate the likelihood of a genotype by calculating the probability of observing a given combination of nucleotide reads at a particular site under every possible genotype. To apply a [**Bayesian**](https://en.wikipedia.org/wiki/Bayesian_statistics) statistics spin to this, we can further weight our calculated likelihoods by things such as known [**allele frequencies**](https://en.wikipedia.org/wiki/Allele_frequency) of previously described SNPs, or the probabilities of incorrectly called bases from sequencing and alignment scores. Through this process, we can enumerate the genotype of our individual at every position where it deviates from the human reference genome. **Finally, we will filter these results by various quality control (QC) parameters to ensure that variants called from well-aligned regions are retained for downstream analysis.**

This process of genotyping and filtering is automated in the ***mpileup\_genotyping.sh*** script in your scripts folder. If you look at this script, you will see that there are two distinct steps. The first step creates a raw set of genotype calls using the ***samtools mpileup***program, and directs this output into a ***‘variant call format’ (.vcf)* file**. The second step filters the genotypes according to read depth. To save computational time, we are going to limit our variant calling to genotypes on chromosome 5.

**3.2** Which parameter should you change to limit the genotyping to chromosome 5? Open ***mpileup\_genotyping.sh*** with nano and change this parameter accordingly (change it to chr5). Save your modified script. Hint: consult the ***mpileup*** section of the [*samtools*](https://www.htslib.org/doc/samtools.html) manual

**3.3** One crafty student suggests that to save further computational time, we could have limited our **alignment step** to align our reads solely to chromosome 6 rather than the entire genome. Why might this not be a good idea?

In order to run the ***mpileup\_genotyping.sh*** on Midway’s compute nodes, we will have to modify and submit a job submit script. Open up ***mpileup\_genotyping\_SUBMIT.sh*** with nano and make the following modifications (similar to what you did in part two):

* Provide a reasonable job-name
* Provide the absolute path to your copied sorted bam file from geno-ready-bam directory (use the *pwd* command)
* Provide the absolute path to your **mpileup\_genotyping.sh** script (use *pwd*)

Now, create (*mkdir*) and change into (*cd*) a new directory called *genotyping* within your CompAsn8 subdirectory. Submit your job from here with sbatch command as you did previously:

*sbatch ../scripts/mpileup\_genotyping\_SUBMIT.sh*

Once this command is executed, be sure that it is either running or at least queued to run. This analysis only takes about a minute to run, so do not be alarmed if the status is complete soon after submitting.

**3.4** Look into SRS003660.genotype.log with *less*. Which step of this process took the longest to complete?

Let’s look briefly at the contents of the created *vcf* file. Check that the first six columns of your output is similar as that given below:

*tail -4 SRS003660.flt.vcf*

chr5 181243433 . T G 81 .

chr5 181259957 . C T 55 .

chr5 181260211 . C T 85 .

chr5 181260427 . CTCTTCTTCTTCTTC CTCTTCTTCTTC 66 .

Each row of this output corresponds to a single variant call. This output contains a lot of information in it. For now, focus only on the following columns:

Column 1 = chromosome number

Column 2 = chromosomal coordinates of variant base

Column 4 = reference allele

**Column 5 = variant allele (your mutation!)**

Column 6 = Phred quality score reflecting probability of variant call. While in practice it is not this simple, we will interpret variants with Phred scores greater than 30 as high confidence variants.

**3.5** Output the last 8 lines of your vcf file with the *tail* command. List the mutations from your data (reference nucleotide, variant nucleotide, and chromosomal coordinates) that pass this Phred quality cutoff.

**3.6** Approximately how many variants are in our ***.vcf*** file output? Use the *wc* command with the required parameter to give the number of lines (check *man wc*) in the ***.vcf*** file. Header lines start with “#” and are not variants (you can use *grep “#” SRS003660.flt.vcf | wc* with the proper parameter to find the number of header lines).

At this point, let’s remove those variants that have Phred quality scores lower than 30. First, let’s extract all of the header lines that start with “#” and direct the output to a new file:

*grep "#" SRS003660.flt.vcf > SRS003660.flt.hq.vcf*

Next, use the *awk* search tool to find all rows where the sixth tab-delimited column has a value greater than 30. Pipe the output to a second *grep* command that confirms the row comes from chromosome 5 (a few header lines sneak through the *awk* command by itself). Redirect the output to append to the file we created above:

*awk '$6 > 30' SRS003660.flt.vcf | grep "chr5" >> SRS003660.flt.hq.vcf*

This new file with the ***.hq.vcf*** (high quality) extension will be used for the final stage.

**3.7** Approximately how many high-quality variants are in your ***.hq.vcf*** output file?

**Part 4: Annotation**

As you can see, there are many variants that occur in an individual’s genome. At first glance, it can be overwhelming to decide what is worth investigating further. However, there are many attributes of a genetic variant that might warrant follow-up investigation. For example, we might expect that **nonsynonymous** protein-coding changes be of more interest than synonymous codon mutations. Similarly, if a SNP has previously been implicated in a genetic disease, it might be worth investigating if it has the same effect in this individual. Information of this sort is contained in many different curated databases. We can automatically compare the genomic coordinates of our variants to these various databases to layer this type of information on our data.

The automated process of layering pertinent biological information on our genetic data is termed **annotation**. We will use the tool ANNOVAR to annotate the variants in your ***.vcf*** file with information on:

* Genomic context (exonic, intronic, intergenic, etc.)
* If coding, the mutation type (synonymous, non-synonymous, frameshift etc.)
* RefGene ID of the affected gene
* If the variant has been discovered previously (has a dbSNP rs ID)
* If the variant has been implicated in any pathologies via genome-wide association studies
* Various metrics to gauge how deleterious and/or functional the variant may be

The code for the ANNOVAR analysis is located in the script ***annovar.sh***. This task is computationally inexpensive, so we will simply run this script on the head node rather than through a job submission script. The *--protocol* argument of the *table\_annovar.pl* command determines which databases we will use for annotation.

We have selected couple of databases: **refGene** to annotate with the RefGene name of the affected gene (or nearest genes, for intergenic mutations), **avsnp150** to annotate with the **rs ID** (SNP ID) of previously described SNPs in the NCBI dbSNP database, **clinvar\_20170905** to indicate if the genetic locus near the given SNP has been associated with any pathologies in previous genome-wide association studies, and **dbnsfp33a**, which applies annotations from the dbNSFP database that compiles many different functional prediction metrics to give a more global prediction of how deleterious a particular nucleotide mutation is likely to be. For more information, see below

refGene: <http://varianttools.sourceforge.net/Annotation/RefGene>

dbSNP: <https://www.ncbi.nlm.nih.gov/snp/>

clinvar: <https://www.ncbi.nlm.nih.gov/clinvar/docs/details/>

First, create (*mkdir*) and change into (*cd*) a new directory called *annotation* within your CompAsn8 subdirectory. We will execute ***annovar.sh*** from this directory, and our output files will be saved here as well. The *annovar.sh* script expects two arguments, provided in a specific order: the ***hq.vcf*** file that is to be annotated, and a prefix that will be used to name the output files. I would execute the following from my annotation directory to run the analysis. Change the relevant pathnames and execute the same analysis. Information about the ANNOVAR run will be printed to your terminal screen.

*../scripts/annovar.sh ../genotyping/SRS003660.flt.hq.vcf SRS003660.chr5*

**4.1** From this output, how many SNPs and how many indels were present in your ***hq.vcf*** file?

**4.2** Does ANNOVAR report anything concerning of which we should be mindful with regards to our ***hq.vcf*** input?

The ANNOVAR analysis produces many output files. The information we are most interested in is contained in the file *SRS003660.chr5.annovar.hg38\_multianno.txt* (**take a look with the *less* command**). While we will be interested in all of the contents of this file, we might also want to parse the data according to certain qualifiers to facilitate analysis and interpretation. For example, despite using an exon-enriched sequencing strategy, many of our reads and variants are derived from non-exonic positions.

**4.3** What are four types (choose any four) of non-exonic functional regions from which we see annotated variants in our results file (this information is in the sixth column)?

The following two commands can be used to create a second file that contains annotations solely for exonic variants. The first command captures the header line and shuttles the output to a new file; the second command pulls out all lines with the word “exonic” and appends these lines to the same file. (Note: in the first command, head -1 is the number one, not the letter L)

*head -1 SRS003660.chr5.annovar.hg38\_multianno.txt > SRS003660.chr5.annovar.hg38\_multianno.exonic.txt*

*grep "exonic" SRS003660.chr5.annovar.hg38\_multianno.txt >> SRS003660.chr5.annovar.hg38\_multianno.exonic.txt*

**4.4** How many of our annotated variants are exonic? (Remember to exclude the header line from the line count!). Why are we mainly interested in exonic variants?

**4.5** Construct another set of commands to create a third file that contains all nonsynonymous exonic variants. Paste the commands you used in your lab text. How many of the exonic variants are nonsynonymous amino acid substitutions? (NOTE: a query for “synonymous” would pick up all lines that contain synonymous, even if that word is nested within a larger word such as nonsynonymous. **Be careful of this fact and always verify that your *grep* search is picking up exclusively those lines you intend!**

If we simply wanted to know the number of lines that fit a certain *grep* search query, but we have no need to construct a new file, we can pipe the output of the *grep* command to the *wc -l* command:

grep "frameshift" SRS003660.chr5.annovar.hg38\_multianno.exonic.txt | wc -l

**44.6** How many of the exonic variants are insertions? How many are deletions?

Hint: if you search for ‘insertions”, “deletions” or “frameshift” using grep and the output you get is 0 then open up the output file using Excel and look at the columns that correspond to the nucleotides. Think about what the corresponding mutation should be when there is a gap '-' in different columns and you can count the number of insertions and deletions by counting the number of gaps in each column.

**Part 5: Interpretation**

We are done performing analyses at this point; the easiest way to look through our results is to download our results files using *scp* command (executed from a new Terminal window on your local computer) or *WinSCP*. Open the exonic results file as a spreadsheet in Excel.

The Excel file should look something like the screenshot below (note that this table contain analysis for chromosome 6). Take some time to get familiar with what each of the columns represent. Although all the columns are important, **the boxed ones are necessary for the completion of this assignment**.

A screenshot of a computer

Description automatically generated

Hint: you can download all three files at the same time using the wildcard character \*. Open a new terminal tab and *without* logging into midway and type (all in on-line):

*scp CNET@midway2.rcc.uchicago.edu:/scratch/midway2/CNET/CompAsn8/annotation/SRS003660.chr5.annovar.hg38\_multianno\* ./*

**5.1** There should be a few exonic variant loci you discovered that are associated with a pathology through genome-wide association studies (GWAS) in the ClinVar database. Identify a non-synonymous mutation that is associated with a specified condition. Which gene does this variant occur in, and with what phenotypic trait does it correlate? Hint: search the variant in the NCBI data base: <https://www.ncbi.nlm.nih.gov>

**5.2** What are the relevant nucleotide and amino acid substitutions for this non-synonymous mutation (hint: look at columns D, E, and J)?

**5.3** This allele has a ***dbSNP*** reference ID, listed in the ***avsnp150*** column. This means that this variant has been discovered previously. Navigate to the NCBI ***dbSNP*** (<http://www.ncbi.nlm.nih.gov/SNP/>) database and search for information on this SNP. What is the ancestral allele in human populations at this site? Which is the minor allele, and what is its allele frequency (minor allele frequency: MAF) for the different projects listed?

The page should look something like below; you might find the boxed regions helpful

A screenshot of a social media post

Description automatically generated

**5.4** Navigate to the population diversity section. Are there any population groups that noticeably differ in their genotype frequencies at this position according to this data? What is the range of MAF across these populations? You can find the data in the frequency tab boxed in the screenshot below.

A screenshot of a social media post

Description automatically generated

**5.5** There are many different metrics used by dbNSFP

<http://varianttools.sourceforge.net/Annotation/dbNSFP>

to try to predict the effects of our non-synonymous variants, such as SIFT, PolyPhen-2, etc. Pick two metrics to research and briefly summarize what kinds of criteria are considered in these metrics’ scoring schemes. For your chosen variant, describe what these metrics predict about any functional ramifications.

The different metrics can be found here:

A screenshot of a computer

Description automatically generated