National and Kapodistrian University of Athens
Department of Informatics and Telecommunications
Master of Science – "Data Science and Information Technologies"
Introduction to Bioinformatics
Academic year 2019-20

4th Lab Exercise

Familiarizing with CLI tools

All the essential files of the specific exercise are inside the compressed file Intro2Bio6.tar.gz found on eclass.

For uncompressing the tar file, type the command:
`tar xzvf Intro2Bio6.tar.gz`

The given files are:

A.vcf	A VCF file with positions in chromosome 20
aligned.sam	An aligned sam file, produced as the output of aligning the read files sample_1.fastq and sample_1.fastq on the human stored in human_g1k_v37_chr20.fasta genome (only chromosome 20), using the bowtie2 aligner
B.vcf	A VCF file with positions in chromosome 20
human_g1k_v37.genome	File containing the size of each chromosome
human_g1k_v37_chr20.fasta	Fasta file containing only chromosome 20 of the human genome (human_g1k_v37 genome build)
sample_1.fastq	Raw reads (forward strand)
sample_2.fastq	Raw reads (reverse strand)
TargetRegion.bed	BED file with 10 positions

Using these files, you are asked to:

1. Count the number of reads inside files sample_1.fastq and sample_2.fastq.
2. Count the number of reads inside the file aligned.sam. How many are mapped and how many are unmapped? Use the command `samtools view` and for locating the necessary flags see the web page http://broadinstitute.github.io/picard/explain-flags.html
3. Convert the file aligned.sam to a bam file, named sample.bam
4. Sort the file sample.bam and store the result into a new file, called sorted.bam. Do you notice any difference in the size of the two files? Why might that be?
5. Create an index for sorted.bam. Is it possible to also repeat for sample.bam? Why not?
6. Using bedtools calculate the coverage of sorted.bam on the genome file human_g1k_v37_chr20.
7. Repeat step 6 but this time produce the output in BEDGRAPH format.

8. Split the file sample.bam into 2 fastq files, one for each strand.
9. The file TargetRegion.bed contains 10 positions, each of length 1 bp. Increase their length by adding 100 bp to the end position and store the result in a file named TargetRegion.100bp.bed. For the execution of the command `bedtools slop` will come handy.
10. Merge any overlapping locations of TargetRegion.100bp.bed, using the command `bedtools merge` and store the result in the file TargetRegion.100bp.merged. bed.
11. Keep from sorted.bam only the reads that are located inside the coordinated designated by TargetRegion.100bp.merged.bed. How many reads exist?
12. Compare the files A.vcf and B.vcf using the command `vcf-compare` of the vcftools suite. How many locations are common in both and how many are unique in A and in B? Prior to using `vcf-compare`, the vcf files need to be compressed using bgzip (i.e. `bgzip A.vcf`) and then indexed using the `tabix` command (i.e. `tabix -p vcf A.vcf.gz`).