BASH BASED PIPELINE FOR VARIANT CALLING AND ANNOTATION

BY OLUWASEUN AKINSULIRE – TEAM CRICK--PROJECT 5

A TECHNICAL REPORT SUBMITTED TO THE HACKBIO IN FULFILLMENT OF THE REQUIREMENTS FOR COMPLETION OF THE GENOMICS WORKSHOP PROGRAM (JULY 2022).

https://github.com/Seun90x/HackBio-Stage-Three-Final-Project– Link to Github repo.

Introduction

A pipeline is a collection of tools/softwares arranged in a sequential order of steps in order to achieve a specific task. The variant calling pipeline identifies single nucleotide variants present within the whole genome and exome data. The variants are identified by comparing the datasets of an individual with a reference sequence. Variant analysis is a crucial procedure for whole exome, targeted panels, and whole genome sequencing. The variant calling pipeline consists of a series of interlinked sequential steps which will be discussed further below. The variant annotation step aims to identify the function and effect of all identified SNPs using SNP annotation tools. In the annotation phase, the biological information is extracted. The functional information is assigned to DNA variants based on available information such as nucleic acid and protein sequences. SnPEff is an open source variant annotation tool. It predicts the effects of variants on genes by using a computational algorithm to detect deleterious variants.

Aim

The aim of this project was to develop a pipeline, which is a bash script with the commands for all the software merged within a for loop, that makes it easy to run routine variant callings anytime at the provision of a starting fastq file.

Methodology

Softwares used:

- FASTP
- BWA
- SAMtools
- Freebayes

They were all installed using the *sudo apt install* command.

Data Collection: Two sample datasets of *E. coli* were downloaded from NCBI (https://www.ncbi.nlm.nih.gov/guide/sitemap), a public biological database, using SRA explorer: A bash script containing the curl command alongside the URL to the SRA for each sequence was written and run to get the datasets into the pipeline for further computation. A txt file (samples.txt) containing the sequences was created to make them accessible, and it was called in every future command The reference genome was downloaded from https://datacarpentry.org using the wget command.

Computation: FASTP was used to trim adapters and remove sequences that do not meet quality standards. It was executed on all the sequences using a for loop.

- for SAMPLE in $(cat samples.txt) ; do

```
fastp \
  -i "raw_data/${SAMPLE}_1.fastq.gz" \
  -I "raw_data/${SAMPLE}_2.fastq.gz" \
  -o "trimmed_reads/${SAMPLE}_1.fastq.gz" \
  -O "trimmed_reads/${SAMPLE}_2.fastq.gz" \
  --html "trimmed_reads_/${SAMPLE}_fastp.html"
done
```

It should be noted that this experiment assumed that all sequences were untrimmed.

The reference genome was then indexed in order to align the data. A directory was created for the reference first, then it was indexed using the *bwa index* command.

- bwa index raw_data/ref_genome/ecoli_rel606.fasta

The trimmed reads were then aligned to the reference genome using the *bwa mem* command.

- for SAMPLE in $(cat samples.txt) ; do

```
bwa mem \
      raw_data/ref_genome/ecoli_rel606.fasta \
      trimmed_reads/${SAMPLE}_1.fastq.gz \
      trimmed_reads/${SAMPLE}_2.fastq.gz \
      > results/sam/${SAMPLE}.aligned.sam
```

done

The alignment output was in SAM format and was then converted to BAM using *samtools view*.

- for SAMPLE in $(cat samples.txt) ; do

```
samtools view \
    -S -b results/sam/${SAMPLE}.aligned.sam \
    > results/bam/${SAMPLE}.aligned.bam
done
```

The BAM file was then sorted by coordinates using *samtools sort* command.

- for SAMPLE in $(cat samples.txt) ; do

```
samtools sort \
    -o results/bam/${SAMPLE}.aligned.sorted.bam \
    results/bam/${SAMPLE}.aligned.bam

done
```

To be able to identify and call variants from the mapped samples, Freebayes was used.

- for SAMPLE in $(cat samples.txt) ; do

```
freebayes \
    -f raw_data/ref_genome/ecoli_rel606.fasta \
    results/bam/${SAMPLE}.aligned.bam \
    > results/vcf/${SAMPLE}.vcf

done
```

Results and Discussion

The outputs are VCF files of the sample datasets. VCF is a text file format (most likely stored in a compressed manner). It contains meta-information lines, a header line, and then data lines each

containing information about a position in the genome. The format also has the ability to contain genotype information on samples for each position. The files show the end result of the variant calling process.

Limitation/Implication

The pipeline will be able to make routine variant calling anywhere and on any device as long as there is a fastqc file to work with. It can take in multiple files at a time and can also manage directories preventing the “no such directory” error when running softwares on files.
However, the analysis assumed that all raw sequence data were of low quality as FASTQC was not carried out on them first to determine that. So, FASTP was run on possibly both low and high quality sequences, hence the pipeline will be incapable of running FASTQC first on datasets.

References

Sinha, A.U., 2020: Introduction to Variant Calling: QC, Alignment, Deduplication, Variant Annotation.

Cingolani, P., Platts, A., Wang, L. L., Coon, M., Nguyen, T., Wang, L., and Ruden, D. M. (2012). A program for annotating and predicting the effects of single nucleotide polymorphisms, SnpEff: SNPs in the genome of *Drosophila melanogaster* strain w1118; iso-2; iso-3. *Fly*, **6**(2), 80-92.