NGS3 DNA-seq: Pre-processing WGS data (Upstream analysis)

TABLE OF CONTENTS

1. Introduction
2. Raw Data Processing
    - Introduce to FastQ files
    - Sequencing quality control (FastQC)
        - Check FastQC
    - Read trimming and filtering (Trimmomatic)
        - Trimmomatic for single-end (SE) reads
        - Trimmomatic for paired-end (PE) reads
3. Alignment: Mapping reads to reference genome
    - Align with BWA mem
    - Sorting and Indexing
4. Mapped reads post-processing
    - Sorting and marking duplicate, inxdexing the BAM file
    - Base Quality Score Recalibration
5. Alignment Data: Quality Control
    - Coverage Analysis
    - Coverage Plot in R

~

1. Introduction

In DNA sequencing, upstream analysis refers to the bioinformatic analysis that is performed on the raw sequencing data to process, filter, and prepare the data for downstream analysis.

2. Raw Data Processing

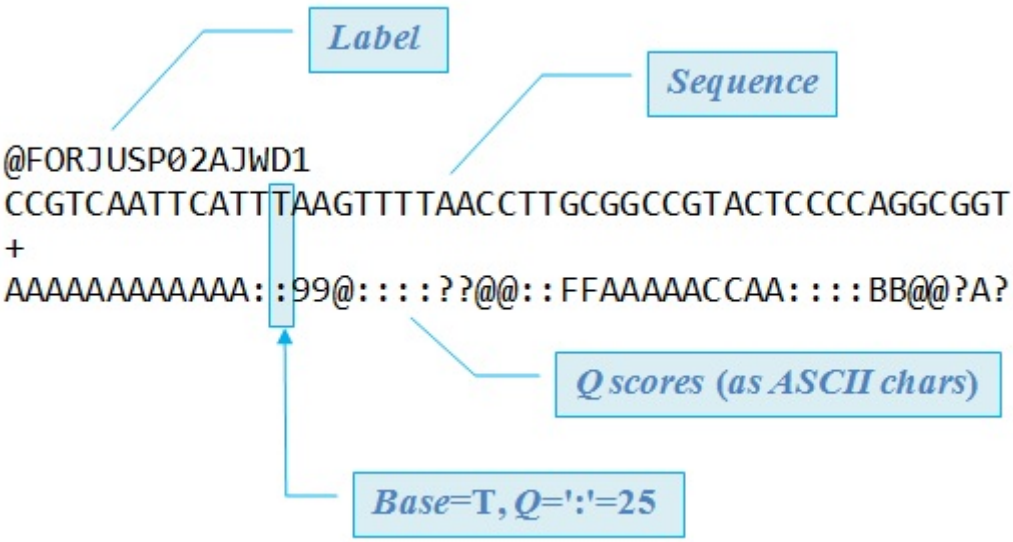
Create shortcuts to work dir

```
p_raw='/home/duydao/dnaseq_work/work/1_raw'
p_trim='/home/duydao/dnaseq_work/work/2_trim'
p_align='/home/duydao/dnaseq_work/work/3_align'
p_ref='/home/duydao/dnaseq_work/work/ref_genome'
```

Introduce to FastQ files

https://en.wikipedia.org/wiki/FASTQ_format

File format



Sequencing quality control (FastQC)

🌟 Hands-on: Check FastQC

Check the quality of raw fastq file using FastQC:

```
cd 0_raw

# Basic syntax
fastqc <file> -o <path/to/output>

# Do for all file
for i in $(ls *.fastq.gz); do fastqc $i; done
```

Hands-on: Check the quality of these files using FASTQC.

```
1. Check the quality score of sample1. Interpret the report.

2. Check the quality score of sample2. Interpret the report.
```

Read trimming and filtering (Trimmomatic)

http://www.usadellab.org/cms/uploads/supplementary/Trimmomatic/TrimmomaticManual_V0.32.pdf

Things to do in Trimming:

- Quality trimming
- Adapter trimming

🌟 Hands-on: Trimming raw data

Trimmomatic for single-end (SE) reads

```
trimmomatic SE \
-phred33 \
-threads 4 \
-trimlog LowQuality_Reads.log \
$p_raw/sample1/LowQuality_Reads.fastq.gz \
$p_trim/sample1/LowQuality_Reads_trimmed.fastq.gz \
SLIDINGWINDOW:4:30 \
MINLEN:20
```

Trimmomatic for paired-end (PE) reads

The flow of reads when trimming with Trimmomatic Paired End mode

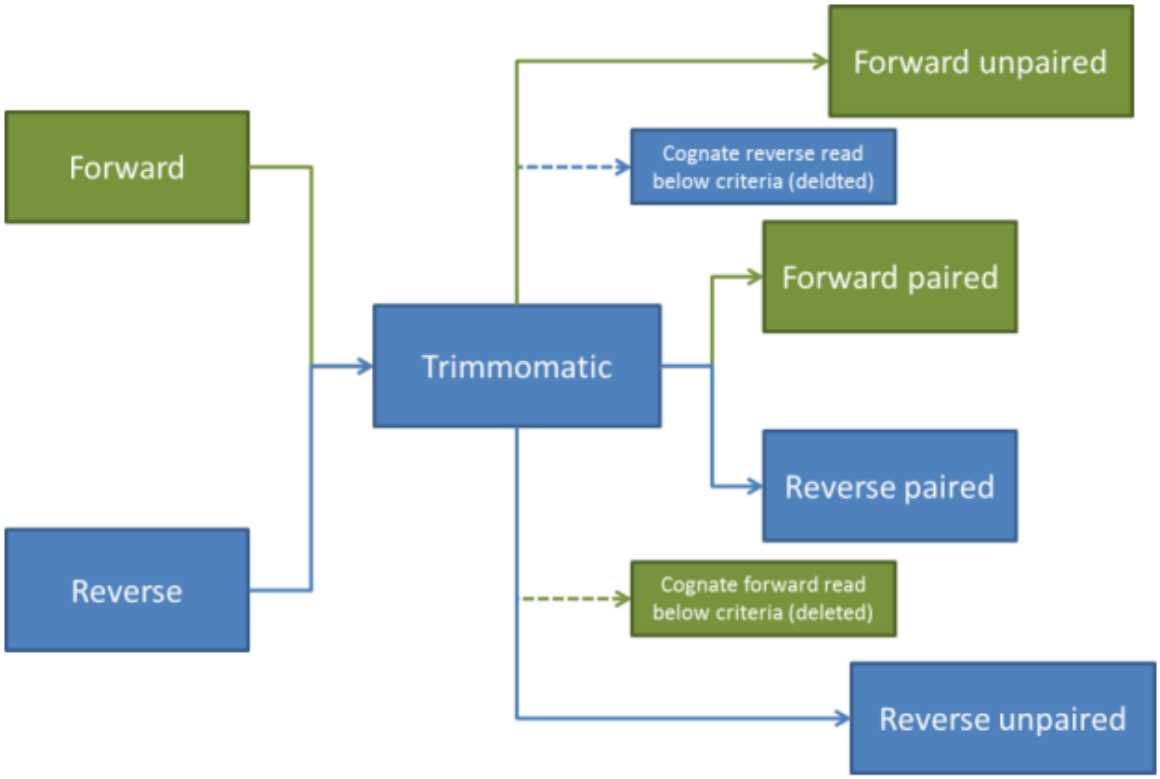


Figure 1: Flow of reads in Trimmomatic Paired End mode

```
# Trimming reads with Trimmomatic. Notes that the adapter's at ILLUMINACLIP
must be specified (try to figure out this by look into the tool's adapters
dir)
trimmomatic PE \
-phred33 \
-threads 4 \
-trimlog $p_trim/sample2/NIST7035.log \
$p_raw/sample2/NIST7035_TAAGGCGA_L001_R1_001.fastq.gz \
$p_raw/sample2/NIST7035_TAAGGCGA_L001_R2_001.fastq.gz \
$p_trim/sample2/NIST7035_TAAGGCGA_L001_R1_001_trimmed_paired.fastq.gz \
$p_trim/sample2/NIST7035_TAAGGCGA_L001_R1_001_trimmed_unpaired.fastq.gz \
$p_trim/sample2/NIST7035_TAAGGCGA_L001_R2_001_trimmed_paired.fastq.gz \
$p_trim/sample2/NIST7035_TAAGGCGA_L001_R2_001_trimmed_unpaired.fastq.gz \
ILLUMINACLIP:/home/duydao/dnaseq_work/tools/trimmomatic/share/trimmomatic-
0.39-2/adapters/NexteraPE-PE.fa:2:30:10:8:3:true \
HEADCROP:10 \
LEADING:3 \
TRAILING:10 \
SLIDINGWINDOW:4:30 \
MINLEN:36
```

Remove unpaired reads

```
cd $p_trim/sample2/

rm *_unpaired*
```

Check QC again after Trim:

```
fastqc NIST7035_TAAGGCGA_L001_R1_001_trimmed_paired.fastq.gz -o qc_checked/
fastqc NIST7035_TAAGGCGA_L001_R2_001_trimmed_paired.fastq.gz -o qc_checked/
```

Is trimming really necessary nowaday?

3. Alignment: Mapping reads to reference genome

First, Index the reference genome (We have done before)

```
bwa index -a bwtsw hs38DH.fa
```

Introduce to SAM/BAM file

Align with BWA mem

```
# Align with BWA mem, using 8 threads
bwa mem -t 4 \
-R '@RG\tID:rg1\tSM:NA12878\tPL:illumina\tLB:lib1\tPU:H7AP8ADXX:1:TAAGGCGA'
\
$p_ref/hg38/hs38DH.fa \
$p_trim/sample2/NIST7035_TAAGGCGA_L001_R1_001_trimmed_paired.fastq.gz \
$p_trim/sample2/NIST7035_TAAGGCGA_L001_R1_001_trimmed_paired.fastq.gz >
$p_align/sample2/NIST7035_aln.sam
```

Convert SAM to BAM

BAM is a binary compressed of SAM file, which is less heavier than SAM.

```
samtools view -Sb NIST7035_aln.sam > NIST7035_aln.bam

# Now we have the compress bam file, we can remove unused sam file.
rm NIST7035_aln.sam
```

Samtools bitflags

The "bitflag" field is a 16-bit integer that encodes various properties of a mapped read.

Use this link to explore more about this: https://broadinstitute.github.io/picard/explain-flags.html

```
# not primary alignment (0x100) & supplementary alignment (0x800)
samtools view -F 0x900 NIST7035_aln.bam

# not primary alignment (0x100)
samtools view -f 0x100 NIST7035_aln.bam
```

Flagstat

The flagstat function of SAMtools provides a summary of the number of records corresponding to each of the bit flags.

```
samtools flagstat NIST7035_aln.bam

#
28137263 + 0 in total (QC-passed reads + QC-failed reads)
28133082 + 0 primary
0 + 0 secondary
4181 + 0 supplementary
0 + 0 duplicates
0 + 0 primary duplicates
28132795 + 0 mapped (99.98% : N/A)
28128614 + 0 primary mapped (99.98% : N/A)
28133082 + 0 paired in sequencing
14066541 + 0 read1
14066541 + 0 read2
0 + 0 properly paired (0.00% : N/A)
28128614 + 0 with itself and mate mapped
0 + 0 singletons (0.00% : N/A)
1382652 + 0 with mate mapped to a different chr
16 + 0 with mate mapped to a different chr (mapQ>=5)
```

4. Mapped reads post-processing

🌟 [Hands-on]: Use GATK to analyze data

Sorting and marking duplicate, inxdexing the BAM file

SortSam

```
gatk SortSam \
--INPUT $p_align/sample2/NIST7035_aln.bam \
```

```
--OUTPUT $p_align/sample2/NIST7035_sorted.bam \
--SORT_ORDER coordinate
```

MarkDuplicates

Detect duplicate using GATK.

```
gatk MarkDuplicates \
--INPUT $p_align/sample2/NIST7035_sorted.bam \
--OUTPUT $p_align/sample2/NIST7035_dedup.bam \
--METRICS_FILE $p_align/sample2/NIST7035.metrics
```

```
# Full BAM file (with duplicate reads are marked)
samtools view NIST7035_dedup.bam #27708663

# BAM file with duplicate reads only
samtools view -f 0x400 NIST7035_dedup.bam
samtools view -c -f 0x400 NIST7035_dedup.bam #Count: 6059946 lines
```

PCR duplication & Optical Duplication

```
# Marked the read as an optical duplicated (DT:Z:SQ) & PCR duplicated
(DT:Z:LB)
gatk MarkDuplicates \
--INPUT $p_align/sample2/NIST7035_sorted.bam \
--OUTPUT $p_align/sample2/NIST7035_dedup.bam \
--METRICS_FILE $p_align/sample2/NIST7035.metrics2 \
--TAGGING_POLICY All
#
samtools view -f 0x400 NIST7035_dedup.bam | grep DT:Z:SQ | less -S
```

BuildBamIndex

```
gatk BuildBamIndex \
--INPUT $p_align/sample2/NIST7035_dedup.bam
```

Using GATK modules

```
#index fasta
samtools faidx $p_ref/hg38/hs38DH.fa
```

Create Sequence Dictionary

```
gatk CreateSequenceDictionary \
--R $p_ref/hg38/hs38DH.fa \
--O $p_ref/hg38/hs38DH.dict
```

Base Quality Score Recalibration

Download data: https://ftp.ncbi.nlm.nih.gov/snp/organisms/human_9606/VCF/

This file is heavy. It contains all the knowns variants sites.

```
wget https://ftp.ncbi.nlm.nih.gov/snp/organisms/human_9606/VCF/00-
All.vcf.gz

wget https://ftp.ncbi.nlm.nih.gov/snp/organisms/human_9606/VCF/00-
All.vcf.gz.tbi
```

Here, we will do Base Quality Score Recalibariation step.

```
# Index vcf
tabix -p vcf $p_ref/hg38/All_20180418.vcf.gz

#
gatk BaseRecalibrator \
-R $p_ref/hg38/hs38DH.fa \
-I $p_align/sample2/NIST7035_dedup.bam \
-known-sites /mnt/portable_drive/Homo_sapiens_assembly38.dbsnp138.vcf \
-O $p_align/sample2/NIST7035-recal.table
#
gatk ApplyBQSR \
-R $p_ref/hg38/hs38DH.fa \
-I $p_align/sample2/NIST7035_dedup.bam \
-bqsr $p_align/sample2/NIST7035-recal.table \
-O $p_align/sample2/NIST7035_recal.bam
#
gatk BaseRecalibrator \
-R $p_ref/hg38/hs38DH.fa \
-I $p_align/sample2/NIST7035_recal.bam \
-known-sites /mnt/portable_drive/Homo_sapiens_assembly38.dbsnp138.vcf \
-O $p_align/sample2/NIST7035-after_recal.table
```

Analyze covatiates before and after ApplyBQSR

```
gatk AnalyzeCovariates \
-before $p_align/sample2/NIST7035-recal.table \
```

```
-after $p_align/sample2/NIST7035-after_recal.table \
-plots $p_align/sample2/NIST7035_recalibration_plots.pdf
```

5. Alignment Data: Quality Control

Coverage Analysis

Remove duplicate

```
gatk MarkDuplicates \
--INPUT $p_align/sample2/NIST7035_sorted.bam \
--OUTPUT $p_align/sample2/NIST7035_remove_dup.bam \
--METRICS_FILE $p_align/sample2/NIST7035_remove_dup.metrics2 \
--REMOVE_DUPLICATES true
```

To check the coverage of exome regions, we must have their location which stored in .bed file.

Download the .bed file that contains Exome locations.
https://sapac.support.illumina.com/downloads/Illumina-dna-prep-exome-20-bed-files.html

"Illumina Exome 2.0 Plus Panel HG38 BED File"

```
# move the file to workspace
mv ~/Downloads/hg38_Twist_ILMN_Exome_2.0_Plus_Panel_annotated.BED ./

cat hg38_Twist_ILMN_Exome_2.0_Plus_Panel_annotated.BED | awk ' {print
$1,$2,$3} ' > hg38_exome.bed
```

Calculate coverage using BED tools

```
bedtools coverage \
-hist \
-a $p_ref/hg38_exome.bed \
-b $p_align/sample2/NIST7035_remove_dup.bam > NIST.bed.cov
#
grep ^all NIST.bed.cov > NIST.all.cov
```

A coverage plot in R

Finally, plot the coverage.

```
cover <- read.table("NIST.all.cov")
cov_cumul <- 1-cumsum(cover[,5])
plot(cover[1:200, 2], cov_cumul[1:200], type='l',
xlab="Depth",
ylab="Fraction of capture target bases >= depth",
ylim=c(0,1.0),
main="Target Region Coverage")
abline(v = 20, col = "gray60")
abline(v = 50, col = "gray60")
abline(v = 80, col = "gray60")
abline(v = 100, col = "gray60")
abline(h = 0.50, col = "gray60")
abline(h = 0.90, col = "gray60")
axis(1, at=c(20,50,80), labels=c(20,50,80))
axis(2, at=c(0.90), labels=c(0.90))
axis(2, at=c(0.50), labels=c(0.50))
```
