CBW HT-seq Module 2 - Genome Alignment

by Mathieu Bourgey, *Ph.D*

Introduction

This workshop will show you how to launch individual first steps of a DNA-Seq pipeline

We will be working on a 1000 genome sample, NA12878. You can find the whole raw data on the 1000 genome website: http://www.1000genomes.org/data

NA12878 is the child of the trio while NA12891 and NA12892 are her parents.

Mother	Father	Child
NA12892	NA12891	NA12878

If you finish early, feel free to perform the same steps on the other two individuals: NA12891 & NA12892.

For practical reasons we subsampled the reads from the sample because running the whole dataset would take way too much time and resources. We're going to focus on the reads extracted from a 300 kbp stretch of chromosome 1

Chromosome	Start	End
chr1	17704860	18004860

Original Setup

Software requirements

These are all already installed, but here are the original links.

- BVATools
- SAMTools
- BWA
- Genome Analysis Toolkit
- Picard

Environment setup

```
export ROOT_DIR=~/workspace/HTSeq_module2
export TRIMMOMATIC_JAR=$ROOT_DIR/tools/Trimmomatic-0.36/trimmomatic-0.36.jar
export PICARD_JAR=$ROOT_DIR/tools/picard-tools-1.141/picard.jar
export GATK_JAR=$ROOT_DIR/tools/GenomeAnalysisTK-3.5/GenomeAnalysisTK.jar
```

```
export BVATOOLS_JAR=$ROOT_DIR/tools/bvatools-1.6/bvatools-1.6-full.jar
export REF=$ROOT_DIR/reference/

rm -rf $ROOT_DIR
mkdir -p $ROOT_DIR
cd $ROOT_DIR
ln -s ~/CourseData/HT_data/Module2/* .
```

Data files

The initial structure of your folders should look like this:

```
<ROOT>
|-- raw_reads/              # fastqs from the center (down sampled)
    `-- NA12878/            # Child sample directory
    `-- NA12891/            # Father sample directory
    `-- NA12892/            # Mother sample directory
`-- reference/              # hg19 reference and indexes
`-- scripts/                # command lines scripts
`-- saved_results/          # precomputed final files
`-- tools/                  # Some tools for the analysis
```

Cheat sheets

- Unix comand line cheat sheet

First data glance

So you've just received an email saying that your data is ready for download from the sequencing center of your choice.

What should you do ?

Fastq files

Let's first explore the fastq file.

Try these commands

```
zless -S raw_reads/NA12878/NA12878_CBW_chr1_R1.fastq.gz
```

These are fastq file.

Could you descride the fastq format ?

```
zcat raw_reads/NA12878/NA12878_CBW_chr1_R1.fastq.gz | head -n4
zcat raw_reads/NA12878/NA12878_CBW_chr1_R2.fastq.gz | head -n4
```

What was special about the output and why was it like that?

You could also count the reads

```
zgrep -c "^@SN1114" raw_reads/NA12878/NA12878_CBW_chr1_R1.fastq.gz
```

We found 56512 reads

Why shouldn't you just do ?

```
zgrep -c "^@" raw_reads/NA12878/NA12878_CBW_chr1_R1.fastq.gz
```

Quality

We can't look at all the reads. Especially when working with whole genome 30x data. You could easilly have Billions of reads.

Tools like FastQC and BVATools readsqc can be used to plot many metrics from these data sets.

Let's look at the data:

```
mkdir -p originalQC/
java -Xmx1G -jar ${BVATOOLS_JAR} readsqc \
  --read1 raw_reads/NA12878/NA12878_CBW_chr1_R1.fastq.gz \
  --read2 raw_reads/NA12878/NA12878_CBW_chr1_R2.fastq.gz \
  --threads 2 --regionName ACTL8 --output originalQC/
```

open a web browser on your laptop, and navigate to `http://cbwXX.dyndns.info/`, where `XX` is the id of your node. You should be able to find there the directory hierarchy under `~/workspace/` on your node. open `originalQC` folder and open the images.

What stands out in the graphs ?

_____

_____

_____

All the generated graphics have their uses. This being said 2 of them are particularly useful to get an overal picture of how good or bad a run went.

These are the Quality box plots (Figure 1)

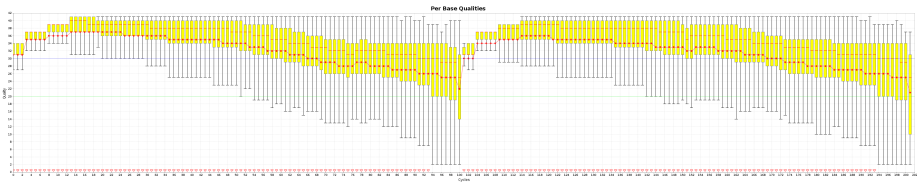

Figure 1: Quality box plots

and the nucleotide content graphs (Figure 2).

The Box plot shows the quality distribution of your data. The Graph goes > 100 because both ends are appended one after the other.

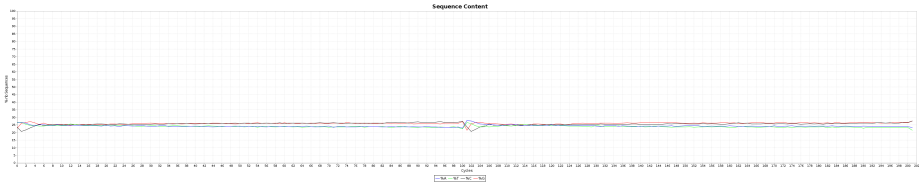

Figure 2: Nucleotide content

The quality of a base is computated using the Phread quality score (Figure 3).

$$Q_{\mathbf{sanger}} = -10 \log_{10} p$$

The formula outputs an integer that is encoded using an ASCII table (Figure 4).

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00	Null	32	20	Space	64	40	@	96	60	`
1	01	Start of heading	33	21	!	65	41	A	97	61	a
2	02	Start of text	34	22	"	66	42	B	98	62	b
3	03	End of text	35	23	#	67	43	C	99	63	c
4	04	End of transmit	36	24	$	68	44	D	100	64	d
5	05	Enquiry	37	25	%	69	45	E	101	65	e
6	06	Acknowledge	38	26	&	70	46	F	102	66	f
7	07	Audible bell	39	27	'	71	47	G	103	67	g
8	08	Backspace	40	28	(	72	48	H	104	68	h
9	09	Horizontal tab	41	29	)	73	49	I	105	69	i
10	0A	Line feed	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage return	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	47	2F	/	79	4F	O	111	6F	o
16	10	Data link escape	48	30	0	80	50	P	112	70	p
17	11	Device control 1	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	50	32	2	82	52	R	114	72	r
19	13	Device control 3	51	33	3	83	53	S	115	73	s
20	14	Device control 4	52	34	4	84	54	T	116	74	t
21	15	Neg. acknowledge	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	54	36	6	86	56	V	118	76	v
23	17	End trans. block	55	37	7	87	57	W	119	77	w
24	18	Cancel	56	38	8	88	58	X	120	78	x
25	19	End of medium	57	39	9	89	59	Y	121	79	y
26	1A	Substitution	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	63	3F	?	95	5F	_	127	7F	□

The way the lookup is done is by taking the the phred score adding 33 and using this number as a lookup in the table. The Wikipedia entry for the FASTQ format has a summary of the varying values.

Older illumina runs were using phred+64 instead of phred+33 to encode their fastq files.

Trimming

After this careful analysis of the raw data we see that - Some reads have bad 3' ends. - no read has adapter sequences in it.

Although nowadays this doesn't happen often, it does still happen. In some cases, miRNA, it is expected to have adapters. Since they are not part of the genome of interest they should be removed if enough reads have them.

To be able to remove adapters and low qualtity beses we will use Trimmomatic.

The adapter file is already in your reference folder.

We can look at the adapters

```
cat $REF/adapters.fa
```

Why are there 2 different ones?

Let's try removing them and see what happens.

```
mkdir -p reads/NA12878/
```

```
java -Xmx2G -cp $TRIMMOMATIC_JAR org.usadellab.trimmomatic.TrimmomaticPE -threads 2 -phred33 \
  raw_reads/NA12878/NA12878_CBW_chr1_R1.fastq.gz \
  raw_reads/NA12878/NA12878_CBW_chr1_R2.fastq.gz \
  reads/NA12878/NA12878_CBW_chr1_R1.t20l32.fastq.gz \
  reads/NA12878/NA12878_CBW_chr1_S1.t20l32.fastq.gz \
  reads/NA12878/NA12878_CBW_chr1_R2.t20l32.fastq.gz \
  reads/NA12878/NA12878_CBW_chr1_S2.t20l32.fastq.gz \
  ILLUMINACLIP:${REF}/adapters.fa:2:30:15 TRAILING:20 MINLEN:32 \
  2> reads/NA12878/NA12878.trim.out
```

```
cat reads/NA12878/NA12878.trim.out
```

What does Trimmomatic says it did ?

Let's look at the graphs now

```
mkdir -p postTrimQC/
java -Xmx1G -jar ${BVATOOLS_JAR} readsqc \
  --read1 reads/NA12878/NA12878_CBW_chr1_R1.t20l32.fastq.gz \
  --read2 reads/NA12878/NA12878_CBW_chr1_R2.t20l32.fastq.gz \
  --threads 2 --regionName ACTL8 --output postTrimQC/
```

How does it look now ?

Could we have done a better job ?

Alignment

The raw reads are now cleaned up of artefacts we can align the read to the reference.

In case you have multiple readsets or library you should align them separatly !

Why should this be done separatly ?

```
mkdir -p alignment/NA12878/

bwa mem -M -t 2 \
 -R '@RG\tID:NA12878\tSM:NA12878\tLB:NA12878\tPU:runNA12878_1\tCN:Broad Institute\tPL:ILLUMI
  ${REF}/hg19.fa \
  reads/NA12878/NA12878_CBW_chr1_R1.t20l32.fastq.gz \
  reads/NA12878/NA12878_CBW_chr1_R2.t20l32.fastq.gz \
  | java -Xmx2G -jar ${PICARD_JAR} SortSam \
  INPUT=/dev/stdin \
  OUTPUT=alignment/NA12878/NA12878.sorted.bam \
 CREATE_INDEX=true VALIDATION_STRINGENCY=SILENT SORT_ORDER=coordinate MAX_RECORDS_IN_RAM=500
```

Why is it important to set Read Group information ?

The details of the fields can be found in the SAM/BAM specifications Here For most cases, only the sample name, platform unit and library one are important.

Why did we pipe the output of one to the other? Could we have done it differently ?

Lane merging (optional)

In case ywe generate multiple lane of sequencing or mutliple library. It is not practical to keep the data splited and all the reads should be merge into one massive file.

Since we identified the reads in the BAM with read groups, even after the merging, we can still identify the origin of each read.

SAM/BAM

Let's spend some time to explore bam files.

try

```
samtools view alignment/NA12878/NA12878.sorted.bam | head -n4
```

Here you have examples of alignment results. A full description of the flags can be found in the SAM specification

Try using picards explain flag site to understand what is going on with your reads

The flag is the 2nd column.

What do the flags of the first 4 reads mean ?

Let's take the 3nd one and find it's pair.

try

```
samtools view alignment/NA12878/NA12878.sorted.bam | grep "1313:19317:61840"
```

Why did searching one name find both reads ?

You can use samtools to filter reads as well.

```
# Say you want to count the *un-aligned* reads, you can use
samtools view -c -f4 alignment/NA12878/NA12878.sorted.bam

# Or you want to count the *aligned* reads you, can use
samtools view -c -F4 alignment/NA12878/NA12878.sorted.bam
```

How many reads mapped and unmapped were there ?

Another useful bit of information in the SAM is the CIGAR string. It's the 6th column in the file. This column explains how the alignment was achieved. M == base aligns *but doesn't have to be a match*. A SNP will have an M even if it disagrees with the reference. I == Insertion D == Deletion S == soft-clips. These are handy to find un removed adapters, viral insertions, etc.

An in depth explanation of the CIGAR can be found here The exact details of the cigar string can be found in the SAM spec as well. Another good site

Cleaning up alignments

We started by cleaning up the raw reads. Now we need to fix and clean some alignments.

Indel realignment

The first step for this is to realign around indels and snp dense regions. The Genome Analysis toolkit has a tool for this called IndelRealigner.

It basically runs in 2 steps 1- Find the targets 2- Realign them.

```
java -Xmx2G  -jar ${GATK_JAR} \
  -T RealignerTargetCreator \
  -R ${REF}/hg19.fa \
  -o alignment/NA12878/realign.intervals \
  -I alignment/NA12878/NA12878.sorted.bam \
  -L chr1

java -Xmx2G -jar ${GATK_JAR} \
  -T IndelRealigner \
  -R ${REF}/hg19.fa \
  -targetIntervals alignment/NA12878/realign.intervals \
  -o alignment/NA12878/NA12878.realigned.sorted.bam \
  -I alignment/NA12878/NA12878.sorted.bam
```

How could we make this go faster ?

How many regions did it think needed cleaning ?

FixMates (optional)

This step shouldn't be necessary...But it is some time.

This goes through the BAM file and find entries which don't have their mate information written properly.

This used to be a problem in the GATKs realigner, but they fixed it. It shouldn't be a problem with aligners like BWA, but there are always corner cases that create one-off corrdinates and such.

This happened a lot with bwa backtrack. This happens less with bwa mem, but it still happens none the less.

```
java -Xmx2G -jar ${PICARD_JAR} FixMateInformation \
 VALIDATION_STRINGENCY=SILENT CREATE_INDEX=true SORT_ORDER=coordinate MAX_RECORDS_IN_RAM=500
  INPUT=alignment/NA12878/NA12878.realigned.sorted.bam \
  OUTPUT=alignment/NA12878/NA12878.matefixed.sorted.bam
```

Mark duplicates

As the step says, this is to mark duplicate reads.

What are duplicate reads? What are they caused by ?

What are the ways to detect them ?

Here we will use picards approach:

```
java -Xmx2G -jar ${PICARD_JAR} MarkDuplicates \
 REMOVE_DUPLICATES=false VALIDATION_STRINGENCY=SILENT CREATE_INDEX=true \
  INPUT=alignment/NA12878/NA12878.matefixed.sorted.bam \
  OUTPUT=alignment/NA12878/NA12878.sorted.dup.bam \
  METRICS_FILE=alignment/NA12878/NA12878.sorted.dup.metrics
```

We can look in the metrics output to see what happened.

```
less alignment/NA12878/NA12878.sorted.dup.metrics
```

We can see that it computed seperate measures for each library.

Why is this important to do and not combine everything ?

How many duplicates were there ?

This is very low, we expect in general <2%.

Recalibration

This is the last BAM cleaning up step.

The goal for this step is to try to recalibrate base quality scores. The vendors tend to inflate the values of the bases in the reads. Also, this step tries to lower the scores of some biased motifs for some technologies.

It runs in 2 steps, 1- Build covariates based on context and known snp sites 2- Correct the reads based on these metrics

```
java -Xmx2G -jar ${GATK_JAR} \
  -T BaseRecalibrator \
  -nct 2 \
  -R ${REF}/hg19.fa \
  -knownSites ${REF}/dbSNP_135_chr1.vcf.gz \
  -L chr1:17700000-18100000 \
 -o alignment/NA12878/NA12878.sorted.dup.recalibration_report.grp \
  -I alignment/NA12878/NA12878.sorted.dup.bam

java -Xmx2G -jar ${GATK_JAR} \
  -T PrintReads \
  -nct 2 \
  -R ${REF}/hg19.fa \
 -BQSR alignment/NA12878/NA12878.sorted.dup.recalibration_report.grp \
  -o alignment/NA12878/NA12878.sorted.dup.recal.bam \
  -I alignment/NA12878/NA12878.sorted.dup.bam
```

Extract Metrics

Once your whole bam is generated, it's always a good thing to check the data again to see if everything makes sens.

Compute coverage

If you have data from a capture kit, you should see how well your targets worked

Both GATK and BVATools have depth of coverage tools. We wrote our own in BVAtools because - GATK was deprecating theirs, but they changed their mind - GATK's is very slow - We were missing some output that we wanted from the GATK's one (GC per interval, valid pairs, etc)

Here we'll use the GATK one

```
java  -Xmx2G -jar ${GATK_JAR} \
  -T DepthOfCoverage \
  --omitDepthOutputAtEachBase \
  --summaryCoverageThreshold 10 \
  --summaryCoverageThreshold 25 \
  --summaryCoverageThreshold 50 \
  --summaryCoverageThreshold 100 \
  --start 1 --stop 500 --nBins 499 -dt NONE \
  -R ${REF}/hg19.fa \
  -o alignment/NA12878/NA12878.sorted.dup.recal.coverage \
  -I alignment/NA12878/NA12878.sorted.dup.recal.bam \
  -L  chr1:17700000-18100000

### Look at the coverage
less -S alignment/NA12878/NA12878.sorted.dup.recal.coverage.sample_interval_summary
```

Coverage is the expected ~30x.

summaryCoverageThreshold is a usefull function to see if your coverage is uniform. Another way is to compare the mean to the median. If both are almost equal, your coverage is pretty flat. If both are quite different, that means something is wrong in your coverage. A mix of WGS and WES would show very different mean and median values.

Insert Size

```
java -Xmx2G -jar ${PICARD_JAR} CollectInsertSizeMetrics \
  VALIDATION_STRINGENCY=SILENT \
  REFERENCE_SEQUENCE=${REF}/hg19.fa \
  INPUT=alignment/NA12878/NA12878.sorted.dup.recal.bam \
 OUTPUT=alignment/NA12878/NA12878.sorted.dup.recal.metric.insertSize.tsv \
 HISTOGRAM_FILE=alignment/NA12878/NA12878.sorted.dup.recal.metric.insertSize.histo.pdf \
  METRIC_ACCUMULATION_LEVEL=LIBRARY

#look at the output
less -S alignment/NA12878/NA12878.sorted.dup.recal.metric.insertSize.tsv
```

What is the insert size and the corresponding standard deviation ?

Is the insert-size important ?

Alignment metrics

For the alignment metrics, we used to use `samtools flagstat` but with bwa mem since some reads get broken into pieces, the numbers are a bit confusing. You can try it if you want.

We prefer the Picard way of computing metrics

```
java -Xmx2G -jar ${PICARD_JAR} CollectAlignmentSummaryMetrics \
  VALIDATION_STRINGENCY=SILENT \
  REFERENCE_SEQUENCE=${REF}/hg19.fa \
  INPUT=alignment/NA12878/NA12878.sorted.dup.recal.bam \
 OUTPUT=alignment/NA12878/NA12878.sorted.dup.recal.metric.alignment.tsv \
  METRIC_ACCUMULATION_LEVEL=LIBRARY

### explore the results
less -S alignment/NA12878/NA12878.sorted.dup.recal.metric.alignment.tsv
```

What is the percent of aligned reads ?

Summary

In this lab, we aligned reads from the sample NA12878 to the reference genome `hg19`:

```
We became familiar with FASTQ and SAM/BAM formats.

We checked read QC with BVAtools.

We trimmed unreliable bases from the read ends using Trimmomatic.

We aligned the reads to the reference using BWA.

We sorted the alignments by chromosome position using PICARD.

We realigned short indels using GATK.

We fixed mate issues using PICARD.

We recalibratie the Base Quality using GATK.

We generate alignment metrics using GATK and PICARD.
```

Aknowledgments

I would like to thank and acknowledge Louis Letourneau for this help and for sharing his material. The format of the tutorial has been inspired from Mar Gonzalez Porta of Embl-EBI.