Lab 4: Practical Next Generation Resequencing

1. Extract data from SRA

1.1 Download SRA source file from our website

wget http://cbb.sjtu.edu.cn/course/bi462/data/SRR069216.sra

wget http://cbb.sjtu.edu.cn/course/bi462/data/SRR341963.sra

(1) Find the two records in SRA (NCBI), tell what they are, respectively?

1.2 Download and install SRA-toolkit

wget http://cbb.sjtu.edu.cn/course/bi462/soft/sratoolkit.tar.gz

1.3 Run the SRA conversion tool – fastq-dump to dump the files (see options for SE/PE reads)

```
fastq-dump -help
fastq-dump --gzip SRR341963.sra
fastq-dump -gzip -split-3 SRR069216.sra
```

1.4 Modify one of the fastQ file (due to wrong annotation in SRA)

wget http://cbb.sjtu.edu.cn/course/bi462/soft/seqtk.tar.gz

...

```
seqtk trimfq -e 75 SRR341963.fastq.gz | gzip > SRR341963_1.fastq.gz
seqtk trimfq -b 75 SRR341963.fastq.gz | gzip > SRR341963_2.fastq.gz
```

(1) What does this processing do?

1.5 Check the validity of the fastQ files

```
for file in *.fastq.gz; do echo $file; gzip -cd $file | head -8; echo ""; done
gzip -cd *.fastq.gz | tail -8
zgrep -c "^@SRR" *.fastq.gz # how many reads?
```

(1) How many files are generated?

(2) How many reads are there in each of these files, repectively?

(3) What does it look like for each record (read)?

(4) Can you tell which quality system these fastQ files use?

2. Check data quality (fastqc)

2.1 Download fastQC

wget http://cbb.sjtu.edu.cn/course/bi462/soft/fastqc.zip

unzip fastqc.zip

2.2 Run fastQC on the fastQ files to generate the quality report

fastqc *.fastq.gz

2.3 Use Explorer (e.g., Firefox) to view the report.

(1) Compare per-base quality for the three fastQ files.

(2) Compare per-base sequence content for the three fastQ files.

(3) Compare k-mer content for the three fastQ files.

3. Adapter trimming

3.1 Download AdapterRemoval from
http://cbb.sjtu.edu.cn/course/bi462/soft/adapterremoval.tar.gz

3.2 Install AdapterRemoval

3.3 Run AdapterRemoval

(1) AdapterRemoval --help

(2) AdapterRemoval –file1 < (gzip –cd SRR341963_1.fastq.gz) \
--file2 <(gzip –cd SRR341963_2.fastq.gz) \
--trimns --trimqualities --minlength 25 --qualitybase 33 \
--collapse --mm 3 --settings SRR341963.settings \
--discarded >(gzip >SRR341963.discard.fastq.gz) \
--outputcollapsed >(gzip > SRR341963.collapsed.fastq.gz) \
--outputcollapsedtruncated >(gzip > SRR341963.collapsed.trunc.fastq.gz) \
--singleton > (gzip > SRR341963.singleton.trunc.fastq.gz) \
--output1 > (gzip > SRR341963.pair1.fastq.gz) \
--output2 > (gzip > SRR341963.pair2.fastq.gz)

(3) gzip –cd SRR341963.pair1.fastq.gz | grep –A 3 "@SRR341963.7$" > untrim.txt

(4) vim –d untrim.txt trim.txt

(5) cat SRR341963_1.settings

(6) AdapterRemovel –file1 <(gzip –cd SRR069216_1.fastq.gz) –file2 <(gzip –cd SRR069216_2.fastq.gz) –basename SRR069216 --collapse --trimns --trimqualities --minlength 25 --qualitybase 33 --pcr1 aatgatacggcgaccaccgagatctacactctttccctacacgacgctcttccgatct --pcr2 caagcagaagacggcatacgagatnnnnnngtgactggagttcagacgtgtgctcttccgatct --mm 3 --output1 >(gzip > SRR069216.pair1.trim.fastq.gz) --output2 >(gzip > SRR069216.pair2.trim.fastq.gz) --discarded >(gzip > SRR069216.trash.fastq.gz) --singleton >(gzip > SRR069216.singleton.fastq.gz) --minquality 4

(7) cat SRR069216.settings

(8) gzip –cd SRR069216.pair1.trim.fastq.gz | head –8

(9) gzip –cd SRR069216.pair2.trim.fastq.gz | head –8

(10) gzip –cd SRR069216.trash.fastq.gz | head

(11) gzip –cd SRR069216.singleton.fastq.gz | head

(1) What can AdapterRemovel do for you?

(2) What did the above commands do? Tell the contents in .settings files.

(3) How many reads are there in each file produced by above commands? That is, how many reads are paired? How many are trash which should be removed? And how many cannot find its pair?

4. Check trimmed reads quality

4.1 Use fastqc again to check the trimmed reads quality

fastqc SRR341963_1.trim.fastq.gz

fastqc SRR069216.pair1.trim.fastq.gz

fastqc SRR069216.pair2.trim.fastq.gz

(1) Read the quality report, and compare per-base sequence quality, per-base sequence content, k-mer content to the ones recovered in previous quality checking.

5. Read mapping/filtering

5.1 Copy the genome sequence of virus to your directory

wget http://cbb.sjtu.edu.cn/course/bi462/ngs/Yersinia_pestisC092chr.fasta

5.2 Create index

bwa index Yersinia_pestisC092chr.fasta -a is

bwa index Yersinia_pestisCO92pPCP1.fasta -a is

5.3 Map the reads to genome, remove unmapped reads and sort by reference coordinates

(1) bwa aln -l 1024 Yersinia_pestisC092chr.fasta SRR341963.pair1.trim.fastq.gz > SRR341963.pair1.trim.chr.fastq.sai

(2) bwa aln -l 1024 Yersinia_pestisCO92pPCP1.fasta SRR341963.pair1.trim.fastq.gz > SRR341963.pair1.trim.pPCP1.fastq.sai

5.4 Generate SAM/BAM output and view with samtools

(1) bwa samse Yersinia_pestisCO92pPCP1.fasta SRR341963.pair1.fastq.sai SRR341963.pair1.trim.gz | samtools view -bSh -q 30 -F0x4 - > SRR341963.pair1.trim.q30.pPCP1.fastq.bam

(2) samtools view -c SRR341963.pair1.trim.q30.pPCP1.fastq.bam

(3) samtools sort SRR341963.pair1.trim.q30.pPCP1.fastq.bam SRR341963.pair1.trim.q30.pPCP1.fastq.sort

(4) bwa samse Yersinia_pestisCO92chr.fasta SRR341963.pair1.trim.chr.fastq.sai SRR341963.pair1.trim.fastq.gz | samtools view -bSh -q 30 -F0x4 - > SRR341963.pair1.trim.q30.chr.fastq.bam

(5) samtools view -c SRR341963.pair1.trim.q30.chr.fastq.bam

(6) samtools sort SRR341963.pair1.trim.q30.chr.fastq.bam SRR341963.pair1.trim.q30.chr.fastq.sort

5.5 Download and install PicardTools

5.6 Run PicardTools and then use samtools to view the results

(1) java -jar MarkDuplicates.jar I=SRR341963.pair1.trim.q30.chr.fastq.sort.bam O=SRR341963.pair1.trim.q30.chr.fastq.Mkdup.bam AS=TRUE M=/dev/null

(2) java -jar MarkDuplicates.jar I=SRR341963.pair1.trim.q30.pPCP1.fastq.sort.bam O=SRR341963.pair1.trim.q30.pPCP1.fastq.Mkdup.bam AS=TRUE M=/dev/null

(3) samtools view -F1024 -c SRR341963.pair1.trim.q30.pPCP1.fastq.Mkdup.bam

(4) samtools view -F1024 -c SRR341963.pair1.trim.q30.chr.fastq.Mkdup.bam

6. Analyzing base composition and nucleotide misincorporation patterns

7. Visualizing alignment

8. Recovering sequence variants

samtools mpileup output a per position alignment. Format is described: http://samtools.sourceforge.net/mpileup.shtml

(1) samtools mpileup

(2) samtools mpileup -f Yersinia_pestisCO92chr.fasta -B SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.bam > SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.all.pileup

(3) head SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.all.pileup

Now output a bcf (binary version of Variant Call Format), and output only variants

(1) samtools mpileup –f Yersinia_pestisCO92chr.fasta –B –u SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.bam > SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.bcf

(2) bcftools view –c –g SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.bcf > SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.varB

(3) wc –l SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.varB

(4) grep –c INDEL SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.varB

(5) grep INDEL SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.varB | head

(6) bcftools view –c –g –v SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.bcf > SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.varBV

(7) wc –l SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.varBV

9. samtools *mpileup -E -A*

All reads were pairwise aligned, mpileup can sort this out using Base Alignment Quality that downscale qualities when they are likely to be misaligned. Check http://samtools.sourceforge.net/mpileup.shtml

(1) samtools mpileup –f Yersinia_pestisCO92chr.fasta –E –A –u SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.bam > SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.EA.bcf

(2) bcftools view –c –g –v SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.EA.bcf > SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.varEA

(3) wc –l SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.varEA

(4) grep –c INDEL SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.varEA

(5) grep INDEL SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.varEA | head

(6) tail SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.varEA

10. vcfutils.*pl*

SNPs could be further filtered for depth thresholds, proximity with indels

(1) vcfutils.pl varFilter

(2) vcfutils.pl varFilter –d 5 –D 100 –Q 30 SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.varEA | awk '{if ($6 > 30){print $0}}' | wc –l

In the paper they claimed for 97 variants, how many could you find here? and how many would you think they are false positives?

(3) grep "2552616" SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.all.pileup

(4) grep "1385780" SRR341963_1.trim.q30.chr.fastq.Mkdup.NoClon.all.pileup

References

Bos, KI et al. A draft genome of Yersinia pestis from victims of the Black Death. 2011. Nature 478:506-510.

Schuenemann, VJ et al. Targeted enrichment of ancient pathogens yielding the pPCP1 plasmid of Yersinia pestis from victims of the Black Death. 2011. PNAS 108:E746-752.

Schubert, M et al. Improving ancient DNA read mapping against modern reference genomes. 2012. BMC genomics 13:178.

Ginolhac, A et al. mapDamage: testing for damage patterns in ancient DNA sequences. 2011.

Bioinformatics 27:2153-2155.

Li, H, and Durbin, R. Fast and accurate long-read alignment with Burrows-Wheeler transform. 2009. Bioinformatics 26:589-595.

Li, H et al. The sequence alignment/map format and SAMtools. 2009. Bioinformatics 25:2078-2079.