NGS Sequence data

Jason Stajich

UC Riverside

jason.stajich[at]ucr.edu

twitter:hyphaltip stajichlab

Lecture available at http://github.com/hyphaltip/CSHL_NGS

NGS sequence data

- Quality control

- Alignment

- Variant calling

- SNPs

- Indels

Sequence data sources

- Sanger

- Long reads, high quality, expensive

- Illumina

- Short reads 50-150bp (HiSeq) and up to 250bp (MiSeq)

- Cheap and Dense read total (HiSeq 200-300M paired-reads for ~$2k)

- 454

- Longish reads 300-500 bp, some homopolymer seq problems,

- Expensive ($10k for 1M reads), recent chemistry problems

- Going away in 3 years

- PacBio

- Long reads, but small amount (10k)

- Low seq quality and not cheap

- Can help improve assemblies, probably not sufficient for an assembly alone (too

expensive to get deep enough coverage)

Sequence data source (cont)

- SOLiD

- Short reads, 30-50bp. Reasonably price-point for the density

- 1/5 as many reads as Illumina HiSeq

- Ion Torrent

- Cheaper machine, fast, 100bp reads and reported 100M

- Quality okay for some applications

Sequencer comparisons

Glenn TC, "Field guide to next-generation DNA sequencers" DOI:10.1111/

j.1755-0998.2011.03024.x

Table 2 Comparison of sequencing instruments, sorted by cost/Mb, with expected performance by mid 2011

Instrument	Run time[a]	Millions of reads/run	Bases/read[b]	Yield Mb/run	Reagent cost/run[c]	Reagent cost/Mb	Minimum unit cost (% run)[d]
3730xl (capillary)	2 h	0.000096	650	0.06	$96	$1500	$6 (1%)
Ion Torrent – '314'chip	2 h	0.10	100	>10	$500	<$50	~$750 (100%)
454 GS Jr. Titanium	10 h	0.10	400	50	$1100	$22	$1500 (100%)
Starlight*	†	~0.01	>1000	†	†	†	†
PacBio RS	0.5–2 h	0.01	860–1100	5–10	$110–900	$11–180	†
454 FLX Titanium	10 h	1	400	500	$6200	$12.4	$2000 (10%)
454 FLX+[e]	18–20 h	1	700	900	$6200	$7	$2000 (10%)
Ion Torrent – '316'chip*	2 h	1	>100	>100	$750	<$7.5	~$1000 (100%)
Helicos[f]	N/A	800	35	28 000	N/A	NA	$1100 (2%)
Ion Torrent – '318'chip*	2 h	4–8	>100	>1000	~$925	~$0.93	~$1200 (100%)
Illumina MiSeq*	26 h	3.4	150 + 150	1020	$750	$0.74	~$1000 (100%)
Illumina iScanSQ	8 days	250	100 + 100	50 000	$10 220	$0.20	$3000 (14%)
Illumina GAIIx	14 days	320	150 + 150	96 000	$11 524	$0.12	$3200 (14%)
SOLiD – 4	12 days	>840[g]	50 + 35	71 400	$8128	<$0.11	$2500 (12%)
Illumina HiSeq 1000	8 days	500	100 + 100	100 000	$10 220	$0.10	$3000 (12%)
Illumina HiSeq 2000	8 days	1000	100 + 100	200 000	$20 120[h]	$0.10	$3000 (6%)
SOLiD – 5500 (PI)*	8 days	>700[g]	75 + 35	77 000	$6101	<$0.08	$2000 (12%)
SOLiD – 5500xl (4hq)*	8 days	>1410[g]	75 + 35	155 100	$10 503[h]	<$0.07	$2000 (12%)
Illumina HiSeq 2000 – v3[i]*	10 days	≤3000	100 + 100	≤600 000	$23 470[h]	≥$0.04	~$3500 (6%)

File formats

FASTQ

```
@SRR527545.1 1 length=76
```

GTCGATGATGCCTGCTAAACTGCAGCTTGACGTACTGCGGACCCTGCAGTCCAGCGCTCGTCATGGAACGCAAACG

+

HHHHHHHHHHHHFGHHHHHHFHHGHHHGHGHEEHHHHHEFFHHHFHHHHBHHHEHFHAH?CEDCBFEFFFFAFDF9

FASTA format

```
>SRR527545.1 1 length=76
```

GTCGATGATGCCTGCTAAACTGCAGCTTGACGTACTGCGGACCCTGCAGTCCAGCGCTCGTCATGGAACGCAAACG

SFF - Standard Flowgram Format - binary format for 454 reads

Colorspace (SOLiD) - CSFASTQ

```
@0711.1 2_34_121_F3
```

```
T113323210022101310111313322000020001200002000010000
```

+

```
64;;9:;>+0*&:*.*1-.5:$2$3&$570*$575&$9966$5835'665
```

Quality Scores in FASTQ files

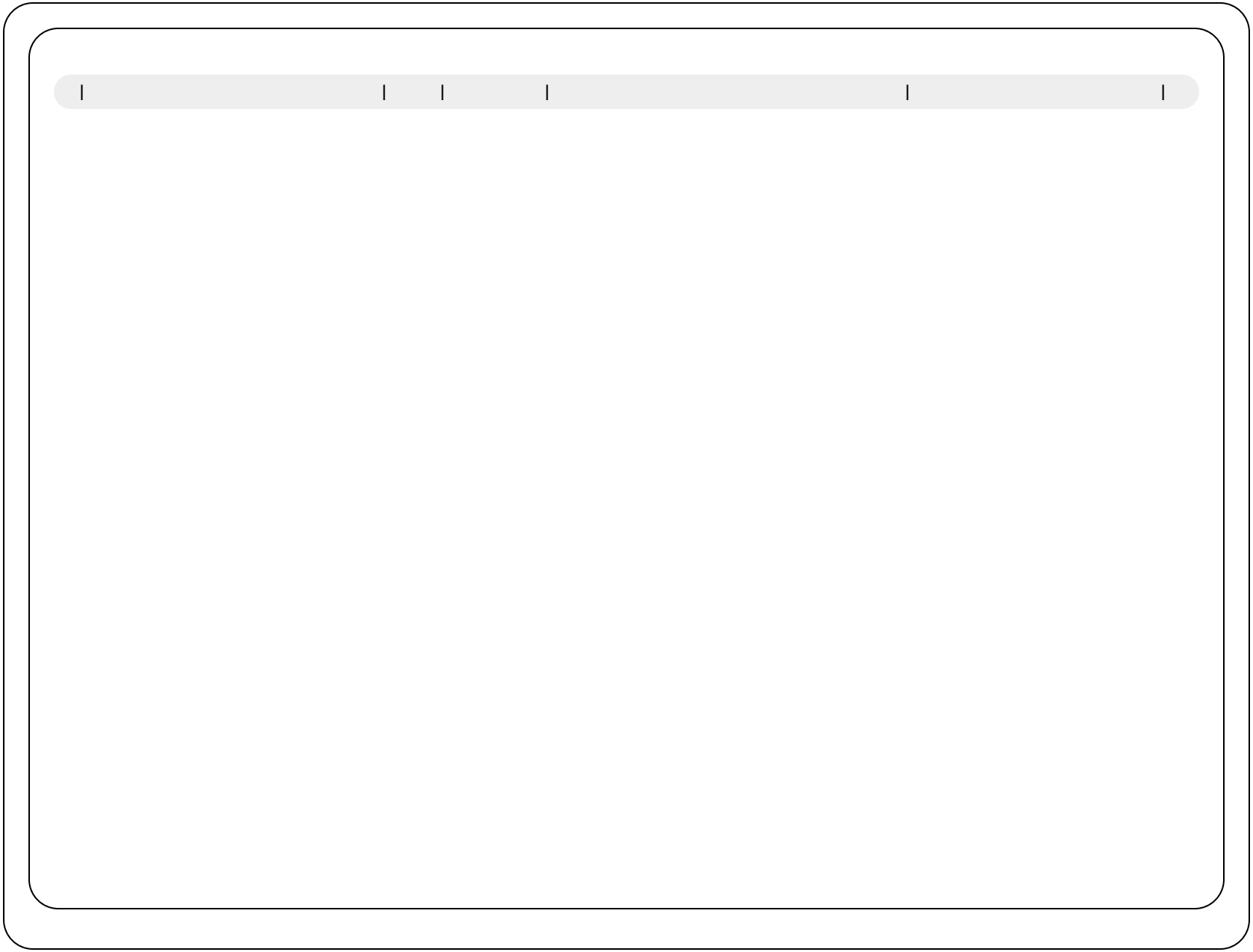
SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS...

.....xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx.....

.....JJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJ.....

LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL.....

```
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
```



```
33                          59    64        73                          104                 126
```

```
S - Sanger        Phred+33,  raw reads typically (0, 40)
```

```
X - Solexa        Solexa+64, raw reads typically (-5, 40)
```

```
I - Illumina 1.3+ Phred+64,  raw reads typically (0, 40)
```

```
J - Illumina 1.5+ Phred+64,  raw reads typically (3, 40)
```

```
    with 0=unused, 1=unused, 2=Read Segment Quality Control Indicator (bold)
```

```
    (Note: See discussion above).
```

Read naming

ID is usually the machine ID followed by flowcell number column, row, cell of the read.

Paired-End naming can exist because data are in two file, first read in file 1 is paired with first

read in file 2, etc. This is how data come from the sequence base calling pipeline. The trailing /1

and /2 indicate they are the read-pair 1 or 2.

In this case #CTTGTA indicates the barcode sequence since this was part of a multiplexed run.

File: Project1_lane6_1_sequence.txt

```
@HWI-ST397_0000:2:1:2248:2126#CTTGTA/1
```

TTGGATCTGAAAGATGAATGTGAGAGACACAATCCAAGTCATCTCTCATG

```
+HWI-ST397_0000:2:1:2248:2126#CTTGTA/1
```

```
eeee\dZddadddddddeeeeeeeedaed_ec_ab_\NSRNRcdddc[_c^d
```

File: Project1_lane6_2_sequence.txt

```
@HWI-ST397_0000:2:1:2248:2126#CTTGTA/2
```

CTGGCATTTTCACCCAAATTGCTTTTAACCCTTGGGATCGTGATTCACAA

```
+HWI-ST397_0000:2:1:2248:2126#CTTGTA/2
```

```
]YYY_\[[][da_da_aa_a_a_b_Y]Z]ZS[]L[\ddccbdYc\ecacX
```

Paired-end reads

These files can be interleaved, several simple tools exist, see velvet package for shuffleSequences

scripts which can interleave them for you.

Interleaved was requried for some assemblers, but now many support keeping them separate.

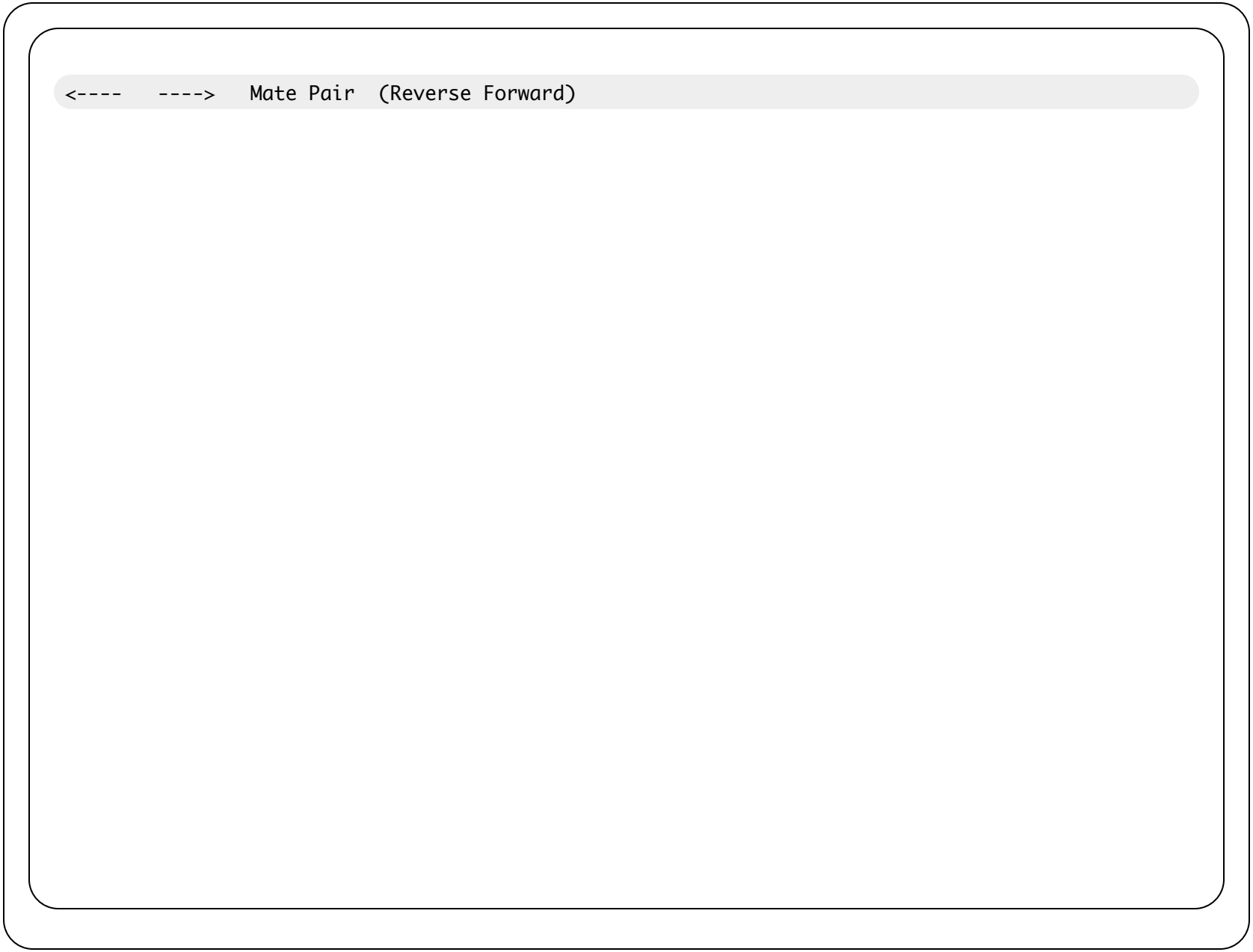
However the order of the reads must be the same for the pairing to work since many tools ignore

the IDs (since this requires additional memory to track these) and instead assume in same order

in both files.

Orientation of the reads depends on the library type. Whether they are

```
---->   <----   Paired End (Forward Reverse)
```


<----   ---->   Mate Pair (Reverse Forward)

Data QC

- Trimming

- Adaptive or a hard cutoff

- sickle, FASTX_toolkit, SeqPrep

- Additional considerations for Paired-end data

- Evaluating quality info with reports

FASTX toolkit

- Useful for trimming, converting and filtering FASTQ and FASTA data

- One gotcha – Illumina quality score changes from 64 to 33 offset

- Default offset is 64, so to read with offset 33 data you need to use -Q 33 option

- fastx_quality_trimmer

- fastx_splitter – to split out barcodes

- fastq_quality_formatter – reformat quality scores (from 33 to 64 or)

- fastq_to_fasta – to strip off quality and return a fasta file

- fastx_collapser - to collapse identical reads. Header includes count of number in the bin

FASTX - fastx_quality_trimmer

- Filter so that X% of the reads have quality of at least quality of N

- Trim reads by quality from the end so that low quality bases are removed (since that is where

errors tend to be)

- Typically we use Phred of 20 as a cutoff and 70% of the read, but you may want other settings

- This is adaptive trimming as it starts from end and removes bases

- Can also require a minimum length read after the trimming is complete

FASTX toolkit - fastx_trimmer

- Hard cutoff in length is sometimes better

- Sometimes genome assembly behaves better if last 10-15% of reads are trimmed off

- Adaptive quality trimming doesn't always pick up the low quality bases

- With MiSeq 250 bp reads, but last 25-30 often low quality and HiSeq with 150 bp often last

20-30 not good quality

- Removing this potential noise can help the assembler perform better

Trimming paired data

- When trimming and filtering data that is paired, we want the data to remain paired.

- This means when removing one sequence from a paired-file, store the other in a separate file

- When finished will have new File_1 and File_2 (filtered & trimmed) and a separate file

File_unpaired.

- Usually so much data, not a bad thing to have agressive filtering

Trimming adaptors

- A little more tricky, for smallRNA data will have an adaptor on 3' end (usually)

- To trim needs to be a matched against the adaptor library - some nuances to make this work

for all cases.

- What if adaptor has low quality base? Indel? Must be able to tolerate mismatch

- Important to get right as the length of the smallRNAs will be calculated from these data

- Similar approach to matching for vector sequence so a library of adaptors and vector could be

used to match against

- Sometimes will have adaptors in genomic NGS sequence if the library prep did not have a

tight size distribution.

Trimming adaptors - tools

- cutadapt – Too to matching with alignment. Can search with multiple adaptors but is

pipelining each one so will take 5X as long if you match for 5 adaptors.

- SeqPrep - Preserves paired-end data and also quality filtering along with adaptor matching

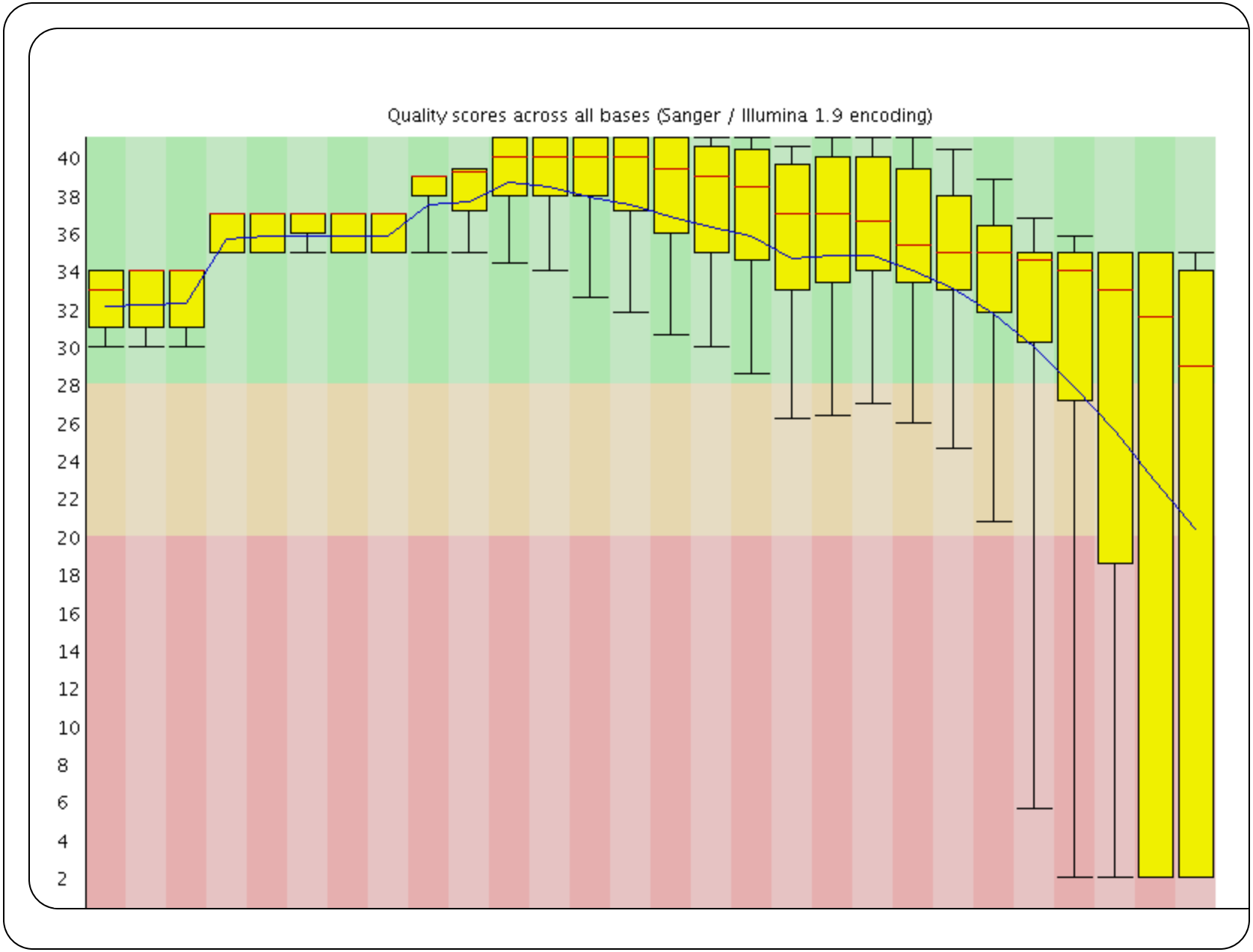
FASTQC for quality control

- Looking at distribution of quality scores across all sequences helpful to judge quality of run

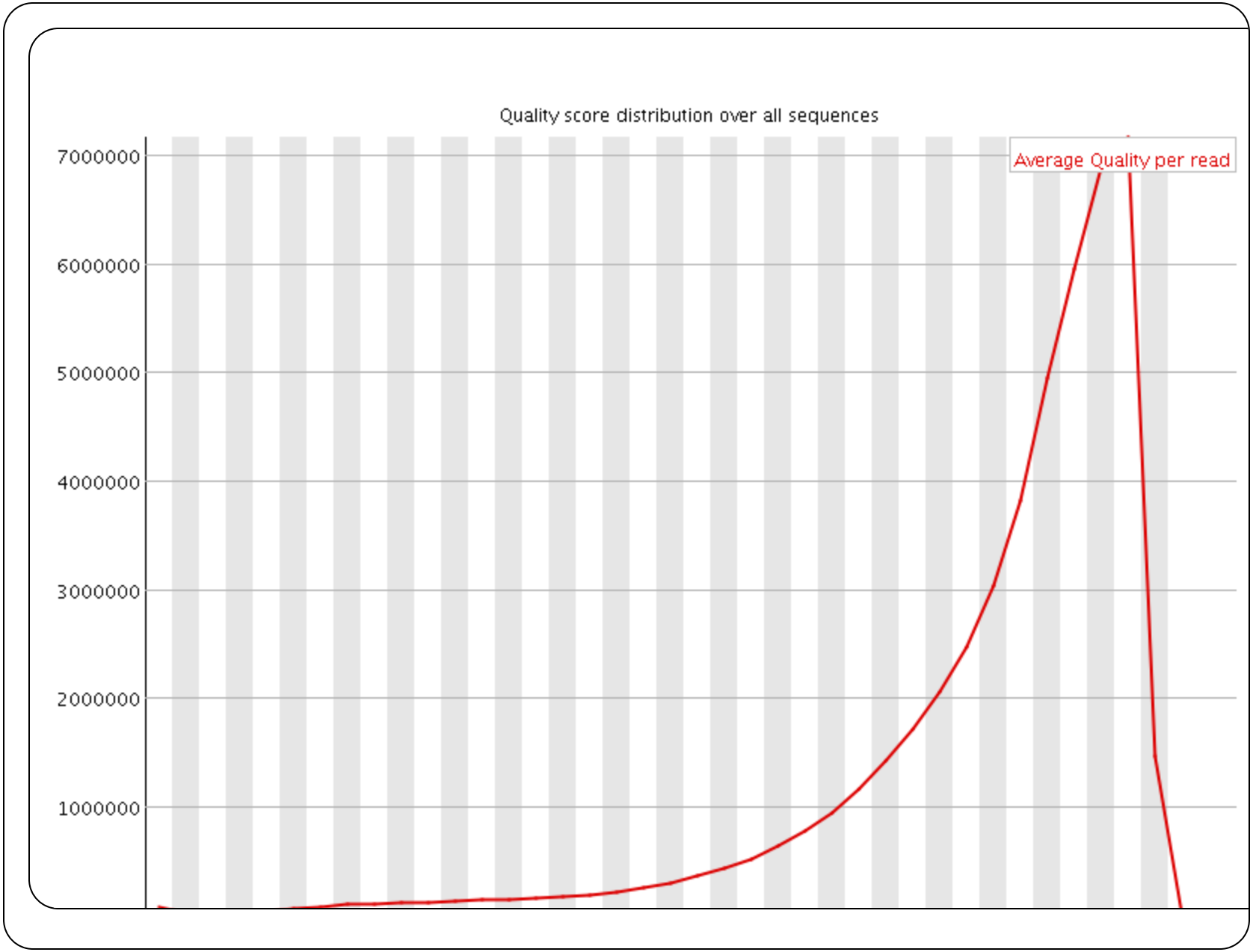
- Overrepresented Kmers also helpful to examine for bias in sequence

- Overrepresented sequences can often identify untrimmed primers/adaptors

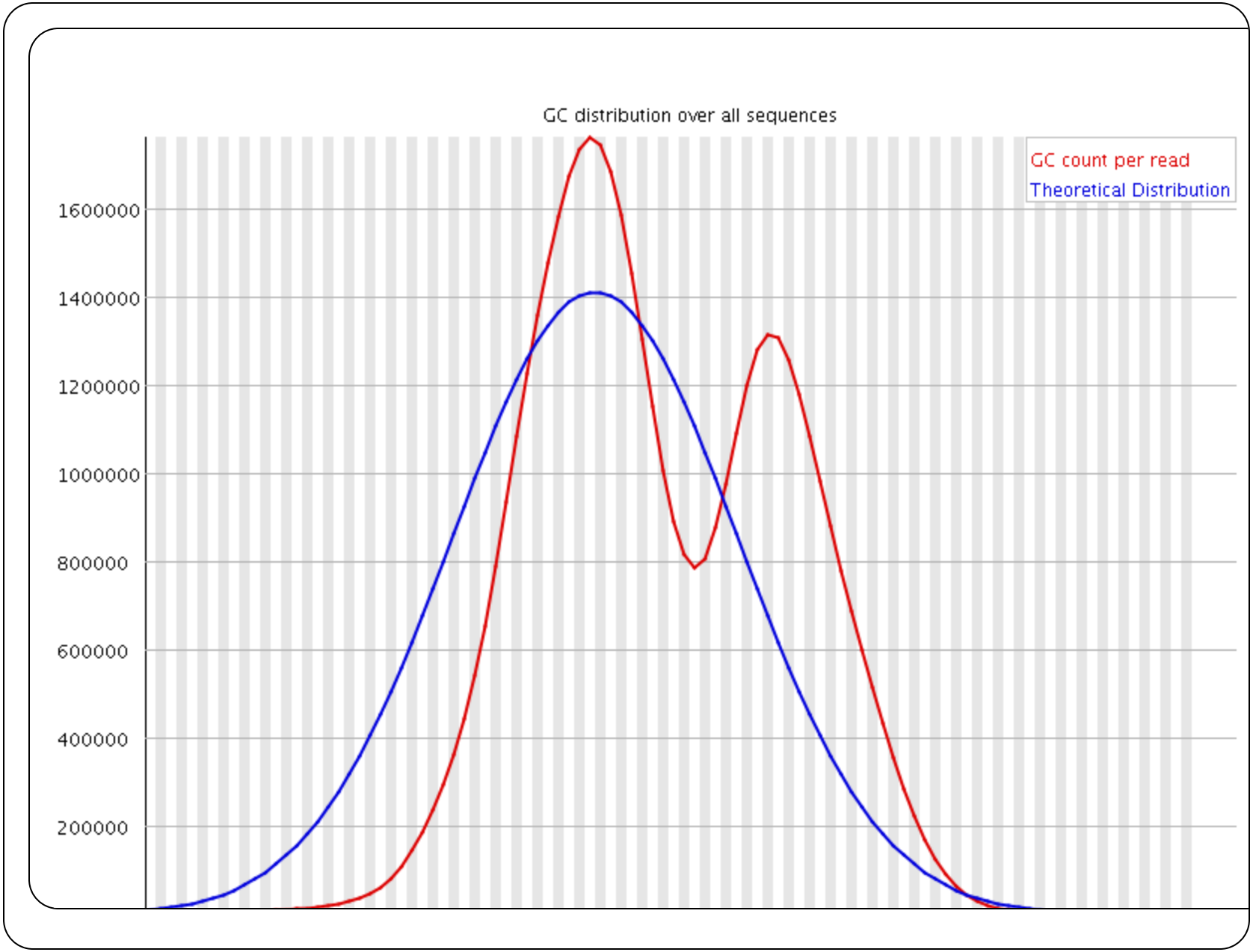
FASTQC - per base quality


Quality scores across all bases (Sanger / Illumina 1.9 encoding)
40
38
36
34
32
30
28
26
24
22
20
18
16
14
12
10
8
6
4
2

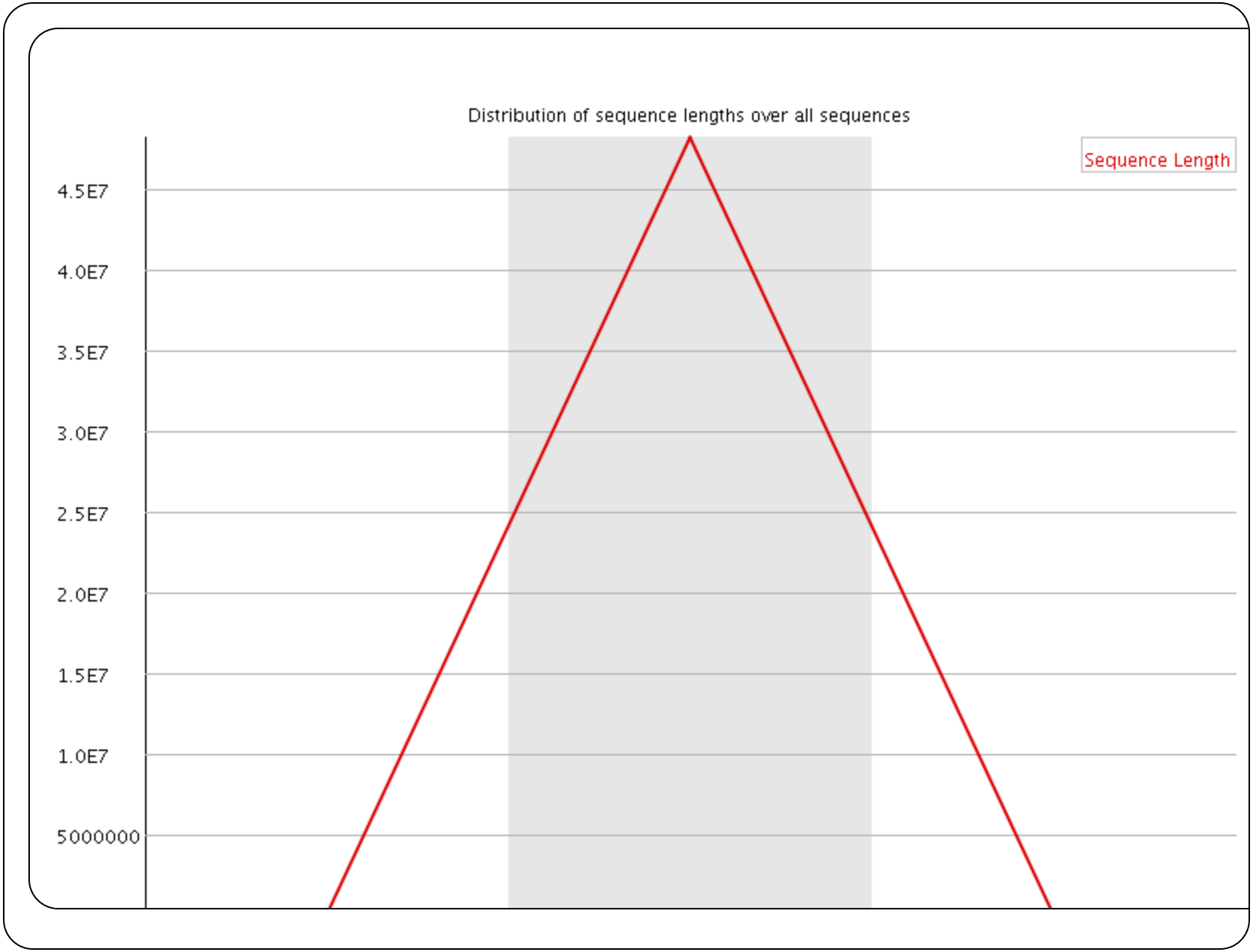
FASTQC - per seq quality


Quality score distribution over all sequences
Average Quality per read
7000000
6000000
5000000
4000000
3000000
2000000
1000000

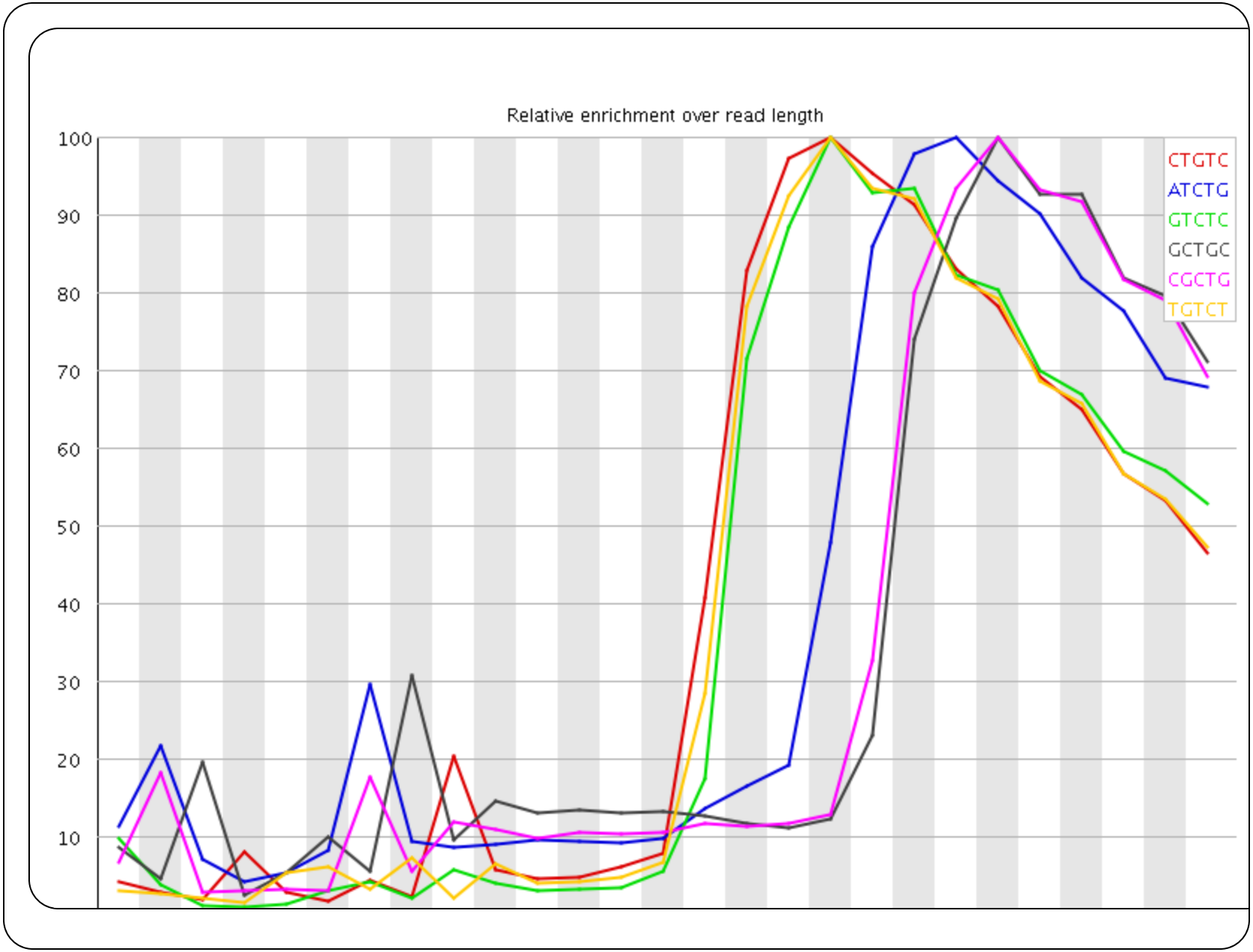
FASTQC - per seq GC content


GC distribution over all sequences
GC count per read
Theoretical Distribution
1600000
1400000
1200000
1000000
800000
600000
400000
200000

FASTQC - Sequence Length


Distribution of sequence lengths over all sequences
Sequence Length
4.5E7
4.0E7
3.5E7
3.0E7
2.5E7
2.0E7
1.5E7
1.0E7
5000000

FASTQC - kmer distribution


Relative enrichment over read length
100
90
80
70
60
50
40
30
20
10
CTGTC
ATCTG
GTCTC
GCTGC
CGCTG
TGTCT

FASTQC - kmer table

Sequence	Count	Obs/Exp Overall	Obs/Exp Max	Max Obs/Exp Position
CTGTC	33437120	7.1755667	14.170156	50-54
ATCTG	33814270	7.064167	15.138542	65-69
GTCTC	32389760	6.950804	14.348899	50-54
GCTGC	29340155	6.9267426	16.531528	70-74
CGCTG	29089270	6.8675127	16.455105	70-74
TGTCT	33183170	6.6351447	13.49372	50-54
CTCTT	33408740	6.5170074	13.125135	50-54
TCTCT	33224365	6.4810414	13.289863	50-54
GCCGA	26214755	6.4660773	16.517157	75-79
GACGC	26117475	6.442083	16.140318	65-69
ATACA	30984490	6.422781	13.738384	55-59
ACATC	30017510	6.3917494	14.464595	60-64
ACACA	28701480	6.3852487	14.690713	60-64
TGCCG	27026655	6.3805614	15.88847	70-74
TACAC	29248425	6.227985	13.874619	60-64
CATCT	30438105	6.203463	13.795571	60-64
TGACG	26974620	6.1994843	15.338736	65-69
CTGAC	27494840	6.1646304	15.06331	65-69
CACAT	28919350	6.1579137	14.247532	60-64

Getting ready to align sequence

Sequence aligners


Batmis
Segemehl
Passion
DynMap
RUM
Bismark
SHRiMP 2
Stampy
MapSplice
REAL
BS Seeker
Supersplat
SpliceMap
BRAT
BRAT-BW
BFAST
GNUMAP
GenomeMapper
mrFAST
mrsFAST
PerM
RNA-Mate
X-Mate
BSMAP
RazerS
MicroRazerS
SHRiMP
BWA
BWA-SW
CloudBurst
ProbeMatch
WHAM
TopHat
Bowtie
Bowtie2
MOM
PASS
Slider
Slider II
OPALMA
SOCS
MAQ
SeqMap
ZOOM
PatMaN
RMAP
SOAP
SOAP2
SOAPSplice
BWT-SW
GMAP
GSNAP
Exonerate
Mummer 3
Blat
SSAHA
2001
2003
2005
2007
2009
2011
2013

Short read aligners

Strategy requires faster searching than BLAST or FASTA approach. Some approaches have been

developed to make this fast enough for Millions of sequences. *maq* - *one of the first aligners*

Burrows-Wheeler Transform is a speed up that is accomplished through a transformation of the

data. Require indexing of the search database (typically the genome). BWA, Bowtie *? LASTZ* ?

BFAST

Workflow for variant detection

- Trim

- Check quality

- Re-trim if needed

- Align

- Possible realign around variants

- Call variants - SNPs or Indels

- Possibly calibrate or optimize with gold standard (possible in some species like Human)

NGS Alignment for DNA

- Short reads (30-200bp)

- Bowtie and BWA - implemented with the BWT algorithm, very easy to setup and run

- SSAHA also useful, uses fair amount of memory

- BFAST - also good for DNA, supports Bisulfide seq,color-space but more complicated to

run

- Longer reads (e.g. PacBio, 454, Sanger reads)

- BWA has A mode using does a Smith-Waterman to place reads. Can tolerate large indels

much better than standard BWA algorithm but slower. BWA-MEM is the currently

reccomended mode - BWA-SW was the earlier implementation and may be more tested,

BWA-MEM is the successor.

- LAST for long reads

BWA alignment choices

From BWA manual

On 350-1000bp reads, BWA-SW is several to tens of times faster than the existing programs. Its

accuracy is comparable to SSAHA2, more accurate than BLAT. Like BLAT, BWA-SW also finds

chimera which may pose a challenge to SSAHA2. On 10-100kbp queries where chimera detection

is important, BWA-SW is over 10X faster than BLAT while being more sensitive.

BWA-SW can also be used to align ~100bp reads, but it is slower than the short-read algorithm.

Its sensitivity and accuracy is lower than SSAHA2 especially when the sequencing error rate is

above 2%. This is the trade-off of the 30X speed up in comparison to SSAHA2's -454 mode.

When running BWA you will also need to choose an appropriate indexing method - read the

manual. This applies when your genome is very large with long chromosomes.

Colorspace alignment

- For SOLiD data, need to either convert sequences into FASTQ or run with colorspace aware

aligner

- BWA, SHRiMP, BFAST can do color-space alignment

Realignment for variant identification

- Typical aligners are optimized for speed, find best place for the read.

- For calling SNP and Indel positions, important to have optimal alignment

- Realignment around variable positions to insure best placement of read alignment

- Stampy applies this with fast BWA alignment followed by full Smith-Waterman alignment

around the variable position

- Picard + GATK employs a realignment approach which is only run for reads which span a

variable position. Increases accuracy reducing False positive SNPs.

Alignment data format

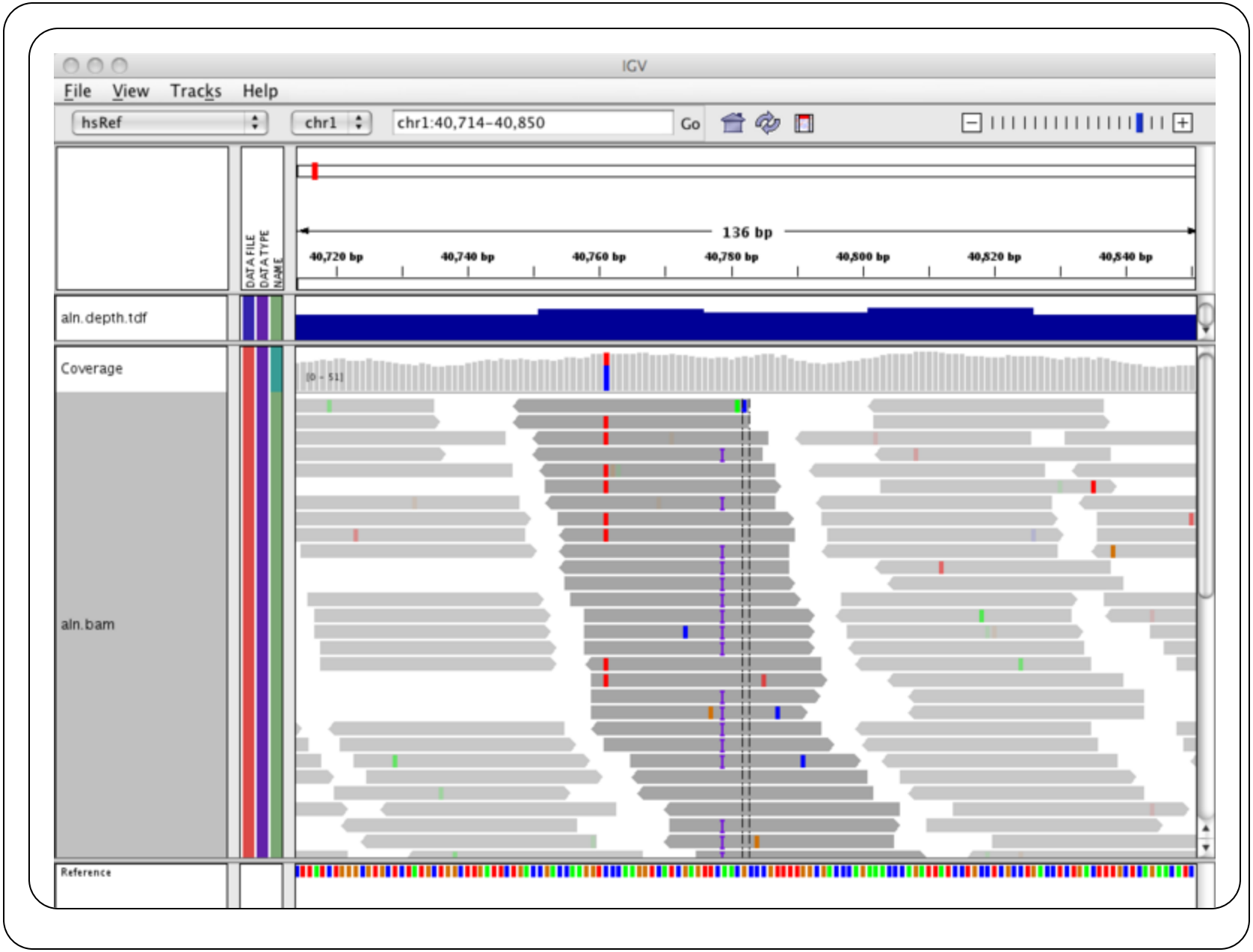
- SAM format and its Binary Brother, BAM

- Good to keep it sorted by chromosome position or by read name

- BAM format can be indexed allowing for fast random access

- e.g. give me the number of reads that overlap bases 3311 to 8006 on chr2

What are we trying to achieve?


IGV
File View Tracks Help
hsRef
chr1
chr1:40,714-40,850
Go
136 bp
40,720 bp
40,740 bp
40,760 bp
40,780 bp
40,800 bp
40,820 bp
40,840 bp
DATA FILE
DATA TYPE
NAME
aln.depth.tdf
Coverage
aln.bam
Reference

Manipulating SAM/BAM

- SAMtools

- One of the first tools written. C code with Perl bindings Bio::DB::Sam (Lincoln Stein FTW!)

with simple Perl and OO-BioPerl interface

- Convert SAM <-> BAM

- Generate Variant information, statistics about number of reads mapping

- Index BAM files and retrieve alignment slices of chromosome regions

- Picard – java library for manipulation of SAM/BAM files

- BEDTools - C tools for interval query in BED,GFF and many other format fiels

- Can generate per-base or per-window coverage from BAM files with GenomeGraph

- BAMTools C++ tools for BAM manipulation and statistics

Using BWA,SAMtools

```
# index genome before we can align (only need to do this once)
```

```
$ bwa index genome/Saccharomyces.fa
```

```
# -t # of threads
```

```
# -q quality trimming
```

```
# -f output file
```

```
# for each set of FASTQ files you want to process these are steps
```

```
$ bwa aln -q 20 -t 16 -f W303_1.sai Saccharomyces W303_1.fastq
```

```
$ bwa aln -q 20 -t 16 -f W303_2.sai Saccharomyces W303_2.fastq
```

```
# do Paired-End alignment and create SAM file
```

```
$ bwa sampe -f W303.sam genome/Saccharomyces.fa W303_1.sai W303_2.sai \
```

```
W303_1.fastq W303_2.fastq
```

```
# generate BAM file with samtools
```

```
$ samtools view -b -S W303.sam > W303.unsrt.bam
```

```
# will create W303.bam which is sorted (by chrom position)
```

```
$ samtools sort W303.unsrt.bam W303.sorted
```

```
# build index
```

```
$ samtools index W303.sorted.bam
```

New BWA options

Some recent improvements to bwa for 70-100bp reads is the `bwa mem` alignment algorithm. All in

one step now to create the sam file.

```
$ bwa mem -t 32 -M genome/Saccharomyces.fa W303_1.fastq W303_2.fastq > W303.sam
```

can even use samtools an pipe it to

bam on the fly

```
$ bwa mem -t 32 -M genome/Saccharomyces.fa W303_1.fastq \
```

```
W303_2.fastq |  samtools view -bS > W303.unsrt.bam
```

BAM using Picard tools

Can also convert and sort all in one go with Picard

```
$ java -Xmx2g -jar SortSam.jar IN=W303.sam OUT=W303.sorted.bam \
```

```
SORT_ORDER=coordinate VALIDATION_STRINGENCY=SILENT CREATE_INDEX=true
```

Or if you already created a bam file, but need to sort it, the input can also be a nam file

```
$ java -Xmx2g -jar SortSam.jar IN=W303.unsrt.bam OUT=W303.sorted.bam \
```

```
SORT_ORDER=coordinate VALIDATION_STRINGENCY=SILENT CREATE_INDEX=true
```

Lots of other resources for SAM/BAM manipulation in Picard documentation on the web

http://picard.sourceforge.net/command-line-overview.shtml.

View header from BAM file

```
$ samtools view -h W303.sorted.bam
```

```
samtools view -h W303.sorted.bam | more
```

```
@HD VN:1.0  GO:none SO:coordinate
```

```
@SQ SN:chrI LN:230218   UR:file:genome/Saccharomyces.fa M5:6681ac2f62509cfc220d78751b8dc524
```

```
@SQ SN:chrII    LN:813184   UR:file:genome/Saccharomyces.fa M5:97a317c689cbdd7e92a5c159acd290d2
```

```
$ samtools view -bS W303.sam > W303.unsrt.bam
```

```
$ samtools sort W303.unsrt.bam W303.sorted
```

```
# this will produce W303.sorted.bam
```

```
$ samtools index W303.sorted.bam
```

```
$ samtools view -h @SQ  SN:chrV LN:576874
```

```
@SQ SN:chrVI    LN:270161
```

```
@SQ SN:chrVII   LN:1090940
```

```
@SQ SN:chrVIII  LN:562643
```

```
@SQ SN:chrIX    LN:439888
```

```
@SQ SN:chrX LN:745751
```

```
@SQ SN:chrXI    LN:666816
```

```
@SQ SN:chrXII   LN:1078177
```

```
@SQ SN:chrXIII  LN:924431
```

```
@SQ SN:chrXIV   LN:784333
```

```
@SQ SN:chrXV    LN:1091291
```

```
@SQ SN:chrXVI   LN:948066
```

```
@SQ SN:chrMito  LN:85779
```

```
@PG ID:bwa  PN:bwa  VN:0.6.2-r131
```

```
SRR527547.1387762   163 chrI    1   17  3S25M1D11M1S    =   213 260
```

```
CACCCACACCACACCCACACACCCACACCCACACCACACC  IIIIIIIIIIIHIIIIHIIIGIIIHDDG8E?@:??DDDA@
```

```
XT:A:M    NM:i:1  SM:i:17 AM:i:17 XM:i:0  XO:i:1
```

SAM format

Col	Field	Type	Regexp/Range	Brief description
1	QNAME	String	`[!-?A-~]{1,255}`	Query template NAME
2	FLAG	Int	`[0,2^16-1]`	bitwise FLAG
3	RNAME	String	`\*|[!-()+-<>-~][!-~]*`	Reference sequence NAME
4	POS	Int	`[0,2^29-1]`	1-based leftmost mapping POSition
5	MAPQ	Int	`[0,2^8-1]`	MAPping Quality
6	CIGAR	String	`\*|([0-9]+[MIDNSHPX=])+`	CIGAR string
7	RNEXT	String	`\*|=|[!-()+-<>-~][!-~]*`	Ref. name of the mate/next segment
8	PNEXT	Int	`[0,2^29-1]`	Position of the mate/next segment
9	TLEN	Int	`[-2^29+1,2^29-1]`	observed Template LENgth
10	SEQ	String	`\*|[A-Za-z=.]+`	segment SEQuence
11	QUAL	String	`[!-~]+`	ASCII of Phred-scaled base QUALity+33

Read Groups

One component of SAM files is the idea of processing multiple files, but that these track back to

specific samples or replicates.

This can be coded in the header of the SAM file

```
@RG ID:Strain124 PL:Illumina PU:Genomic LB:Strain124 CN:Broad
```

It can also be encoded on a per-read basis so that multiple SAM files can be combined together

into a single SAM file and that the origin of the reads can still be preserved. This is really useful

when you want to call SNPs across multiple samples.

The AddOrReplaceReadGroups.jar command set in Picard is really useful for manipulating these.

samtools flagstat

```
4505078 + 0 in total (QC-passed reads + QC-failed reads)
```

```
0 + 0 duplicates
```

```
4103621 + 0 mapped (91.09%:-nan%)
```

```
4505078 + 0 paired in sequencing
```

```
2252539 + 0 read1
```

```
2252539 + 0 read2
```

```
3774290 + 0 properly paired (83.78%:-nan%)
```

```
4055725 + 0 with itself and mate mapped
```

```
47896 + 0 singletons (1.06%:-nan%)
```

```
17769 + 0 with mate mapped to a different chr
```

```
6069 + 0 with mate mapped to a different chr (mapQ>=5)
```

Realigning around Indels and SNPs

To insure high quality Indelcalls, the reads need to realigned after placed by BWA or other aligner.

This can be done with PicardTools and GATK. Note that `-jar` `GATK` and `picard-tools` folders

need to refer to the whole path where they are located (unless are in your current directory)

Need to Deduplicate reads

```
$ java -Xmx3g -jar picard-tools/MarkDuplicates.jar INPUT=W303.sorted.bam \
```

```
  OUTPUT=W303.dedup.bam METRICS_FILE=W303.dedup.metrics \
```

```
CREATE_INDEX=true VALIDATION_STRINGENCY=SILENT
```

Fixing Read-Groups

I am using W303 since it is the strain name for this sequencing record. We'd do this for each file

RGLB would be processed as a bam file then later combine them. For now we will just treat it all

like one sample

```
$ java -Xmx3g -jar $PICARD/AddOrReplaceReadGroups.jar INPUT=W303.dedup.bam \
```

```
  OUTPUT=W303.readgroup.bam SORT_ORDER=coordinate CREATE_INDEX=True \
```

```
  RGID=W303 RGLB=SRR527545 RGPL=Illumina RGPU=Genomic RGSM=W303 \
```

```
VALIDATION_STRINGENCY=SILENT
```

Then identify Intervals around

variants

```
$ java -Xmx3g -jar GATK/GenomeAnalysisTK.jar -T RealignerTargetCreator \
```

```
-R genome/Saccharomyces.fa \
```

```
-o W303.intervals -I W303.readgroup.bam
```

Then realign based on these intervals

```
$ java -Xmx3g -jar GATK/GenomeAnalysisTK.jar -T IndelRealigner \
```

```
-R genome/Saccharomyces.fa \
```

```
-targetIntervals W303.intervals -I W303.readgroup.bam -o W303.realign.bam
```

SAMtools and VCFtools to call SNPs

```
$ samtools mpileup -D -S -gu -f genome/Saccharomyces.fa ABC.bam | \
```

```
bcftools view -bvcg - > ABC.raw.bcf
```

```
$ bcftools view ABC.raw.bcf | vcfutils.pl varFilter -D100 > ABC.filter.vcf
```

GATK to call SNPs

```
# run GATK with 4 threads (-nt)
```

```
# call SNPs only (-glm, would specific INDEL for Indels or can ask for BOTH)
```

```
$ java -Xmx3g -jar GenomeAnalysisTK.jar -T UnifiedGenotyper \
```

```
  -glm SNP -I W303.realign.bam -R genome/Saccharomyces.fa \
```

```
-o W303.GATK.vcf -nt 4
```

GATK to call INDELs

```
# run GATK with 4 threads (-nt)
```

```
# call SNPs only (-glm, would specific INDEL for Indels or can ask for BOTH)
```

```
$ java -jar GenomeAnalysisTK.jar -T UnifiedGenotyper\
```

```
  -glm INDEL -I W303.realign.bam \
```

```
  -R genome/Saccharomyces.fa -o W303.GATK_INDEL.vcf -nt 4
```

VCF Files

Variant Call Format - A standardized format for representing variations. Tab delimited but with specific

ways to encode more information in each column.

```
##FORMAT=<ID=AD,Number=.,Type=Integer,Description="Allelic depths for the ref and alt alleles in the c
```

```
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Approximate read depth (reads with MQ=255 or with b
```

```
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality">
```

```
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
```

```
##FORMAT=<ID=PL,Number=G,Type=Integer,Description="Normalized, Phred-scaled likelihoods for genotypes
```

```
##INFO=<ID=AC,Number=A,Type=Integer,Description="Allele count in genotypes, for each ALT allele, in th
```

```
##INFO=<ID=AF,Number=A,Type=Float,Description="Allele Frequency, for each ALT allele, in the same orde
```

```
#CHROM  POS ID  REF ALT QUAL    FILTER  INFO    FORMAT  W303
```

```
chrI    141 .   C   T   47.01   .   AC=1;AF=0.500;AN=2;BaseQRankSum=-0.203;DP=23;Dels=0.00;
```

```
FS=5.679;HaplotypeScore=3.4127;MLEAC=1;MLEAF=0.500;MQ=53.10;MQ0=0;MQRankSum=-2.474;QD=2.04;ReadPosRank
```

```
SB=-2.201e+01   GT:AD:DP:GQ:PL  0/1:19,4:23:77:77,0,565
```

```
chrI    286 .   A   T   47.01   .   AC=1;AF=0.500;AN=2;BaseQRankSum=-0.883;DP=35;Dels=0.00;
```

```
FS=5.750;HaplotypeScore=0.0000;MLEAC=1;MLEAF=0.500;MQ=46.14;MQ0=0;MQRankSum=-5.017;QD=1.34;ReadPosRank
```

```
SB=-6.519e-03   GT:AD:DP:GQ:PL  0/1:20,15:35:77:77,0,713
```

Filtering Variants

GATK best Practices http://www.broadinstitute.org/gatk/guide/topic?name=best-practices

emphasizes need to filter variants after they have been called to removed biased regions.

These refer to many combinations of information. Mapping quality (MQ), Homopolymer run length

(HRun), Quality Score of variant, strand bias (too many reads from only one strand), etc.

```
-T VariantFiltration -o STRAINS.filtered.vcf
```

```
--variant W303.raw.vcf \
```

```
--clusterWindowSize 10  -filter "QD<8.0" -filterName QualByDepth \
```

```
-filter "MQ>=30.0" -filterName MapQual \
```

```
-filter "HRun>=4" -filterName HomopolymerRun \
```

```
-filter "QUAL<100" -filterName QScore \
```

```
-filter "MQ0>=10 && ((MQ0 / (1.0 * DP)) > 0.1)" -filterName MapQualRatio \
```

```
-filter "FS>60.0" -filterName FisherStrandBias \
```

```
-filter "HaplotypeScore > 13.0" -filterName HaplotypeScore \
```

```
-filter "MQRankSum < -12.5" -filterName MQRankSum  \
```

```
-filter "ReadPosRankSum < -8.0" -filterName ReadPosRankSum  >& output.filter.log
```

VCFtools

A useful tool to JUST get SNPs back out from a VCF file is vcf-to-tab (part of vcftools).

```
$ vcf-to-tab < INPUT.vcf > OUTPUT.tab
```

```
#CHROM  POS REF W303
```

```
chrI    141 C   C/T
```

```
chrI    286 A   A/T
```

```
chrI    305 C   C/G
```

```
chrI    384 C   C/T
```

```
chrI    396 C   C/G
```

```
chrI    476 G   G/T
```

```
chrI    485 T   T/C
```

```
chrI    509 G   G/A
```

```
chrI    537 T   T/C
```

```
chrI    610 G   G/A
```

```
chrI    627 C   C/T
```

VCFtools to evaluate and manipulate

```
$ vcftools --vcf W303.GATK.vcf --diff W303.filter.vcf
```

```
N_combined_individuals: 1
```

```
N_individuals_common_to_both_files: 1
```

```
N_individuals_unique_to_file1:  0
```

```
N_individuals_unique_to_file2:  0
```

```
Comparing sites in VCF files...
```

```
Non-matching REF at chrI:126880 C/CTTTTTTTTTTTTTT. Diff results may be unreliable.
```

```
Non-matching REF at chrI:206129 A/AAC. Diff results may be unreliable.
```

```
Non-matching REF at chrIV:164943 C/CTTTTTTTTTTTT. Diff results may be unreliable.
```

```
Non-matching REF at chrIV:390546 A/ATTGTTGTTGTTGT. Diff results may be unreliable.
```

```
Non-matching REF at chrXII:196750 A/ATTTTTTTTTTTTTT. Diff results may be unreliable.
```

```
Found 8604 SNPs common to both files.
```

```
Found 1281 SNPs only in main file.
```

```
Found 968 SNPs only in second file.
```

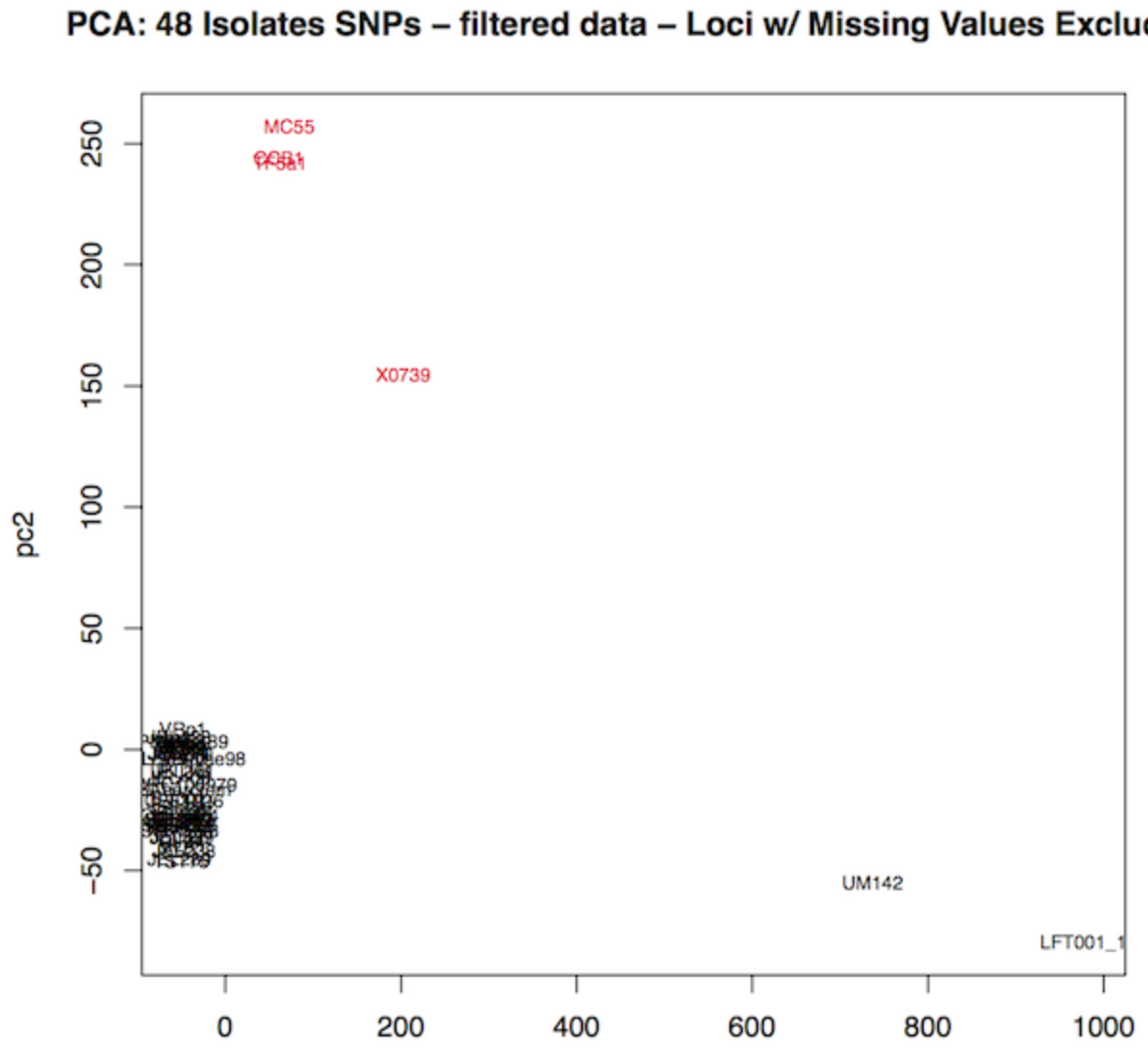
```
# calculate Tajima's D in binsizes of 1000 bp [if you have multiple individuals]
```

```
$ vcftools --vcf Sacch_strains.vcf --TajimaD 1000
```

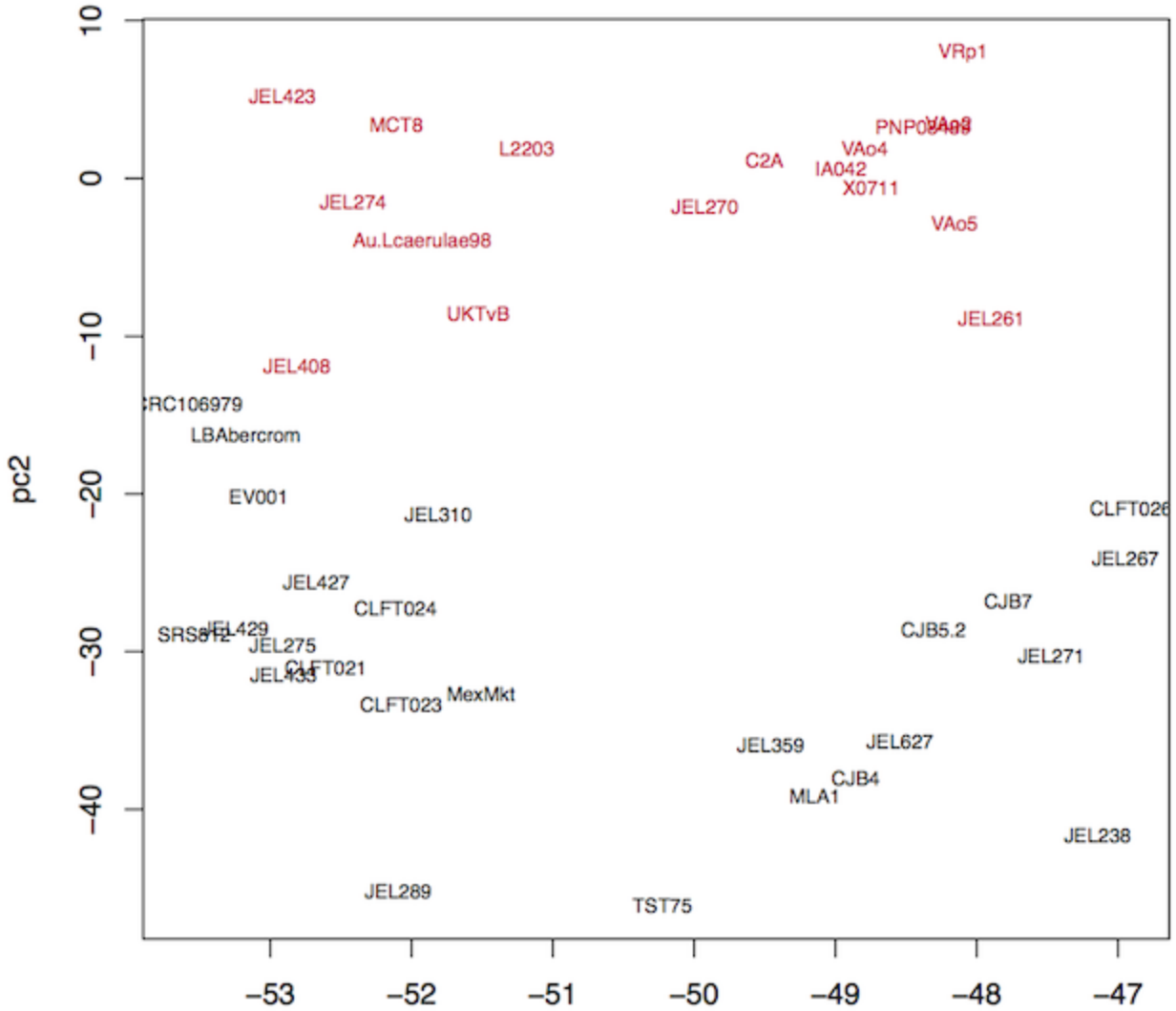
Can compare strains in other ways

PCA plot of strains from the SNPs converted to 0,1,2 for homozygous Ref, Homozygous Alt allele,

or heterozygous (done in R)


PCA: 48 Isolates SNPs – filtered data – Loci w/ Missing Values Excluded
MC55
X0739
UM142
LFT001_1
pc2
250
200
150
100
50
0
−50
0
200
400
600
800
1000

Zoomed PCA plot


pc2
10
0
−10
−20
−30
−40
−53
−52
−51
−50
−49
−48
−47
VRp1
JEL423
MCT8
L2203
C2A
VAo4
IA042
X0711
JEL274
JEL270
VAo5
Au.Lcaerulae98
UKTvB
JEL261
JEL408
LBAbercrom
EV001
JEL310
JEL267
JEL427
CLFT024
CJB7
JEL429
JEL275
CJB5.2
JEL271
CLFT021
CLFT023
MexMkt
JEL359
JEL627
CJB4
MLA1
JEL238
JEL289
TST75

Summary

- Reads should be trimmed, quality controlled before use. Preserving Paired-End info is

important

- Alignment of reads with several tools possible, BWA outlined here

- SAMTools and Picard to manipulate SAM/BAM files

- Genotyping with SAMtools and GATK

- Summarizing and manipulating VCF files with VCFtools