

NGS read alignment

Vivek Iyer
Human Genetics Informatics
Wellcome Sanger Institute
vvi@sanger.ac.uk

In the footsteps of Anthony Doran



NGS read alignment

The most important first step in understanding next-generation sequencing data is the initial alignment or assembly that determines whether an experiment has succeeded and provides a first glimpse into the results.

Flicek & Birney, 2009. *Nature Methods*

Sequence alignment in NGS is:

- *Process of determining the most likely source of the observed DNA sequencing read within the reference genome sequence.*

Principles and approaches to sequence alignment have not changed much since 80's



Overview

Intro - REFERENCES

Methods / Aligners

Alignment Outputs

Alignment Viewers

NGS Workflows, QC and BAM Improvement



The *reference* is key in all that follows

Sequence alignment in NGS is:

- *Process of determining the most likely source of*
 - *the observed DNA sequencing read*
 - *within*
 - *the reference genome sequence*



The reference is key in all that follows

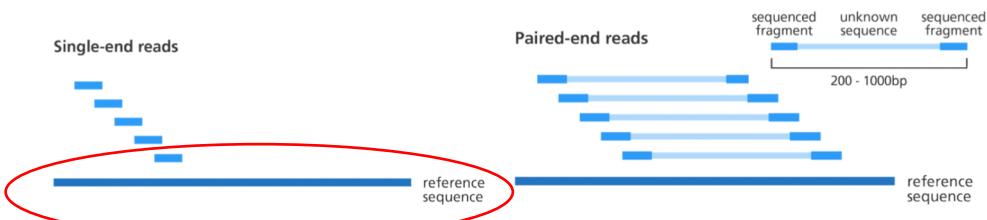


Image credit: Genome Research Limited

The view of alignment in *this* lecture is *wildly* asymmetric:

- short read sequences
- very long reference sequences. Genomes (~Gbp) or Transcriptomes (~100Mbp)

Where do references come from?

Organism => Consortium => Money => Sequencing => Assembly => Curation=> Dissemination => Curation, Dissemination => ...

Human, Mouse, Zebrafish, Chicken:

Lots of curation / dissemination was done at various centres and distributed by NCBI. NCBI still distributes!

Now the umbrella for curation / patching is: **Genome Reference Consortium**

The reference is key in all that follows

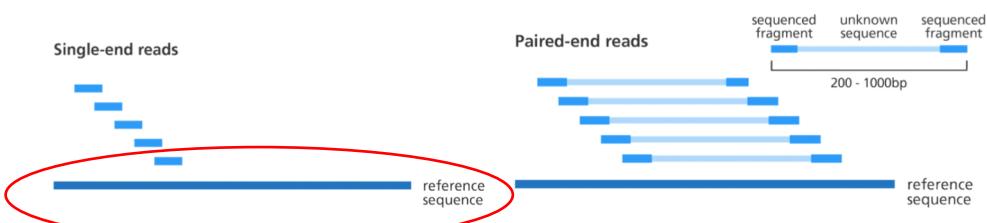


Image credit: Genome Research Limited

The view of alignment in *this* lecture is *wildly* asymmetric:

- short read sequences
- very long reference sequences. Genomes (~Gbp) or Transcriptomes (~100Mbp)

Where do references come from?

Organism => Consortium => Money => Sequencing => Assembly => Curation=> Dissemination => Curation, Dissemination => ...

Human, Mouse, Zebrafish, Chicken:

Lots of curation / dissemination was done at various centres and distributed by NCBI. NCBI still distributes!

Now the umbrella for curation / patching is: **Genome Reference Consortium**

TREE OF LIFE PROGRAM AT SANGER WILL CHANGE THIS FOREVER

The reference is key in all that follows

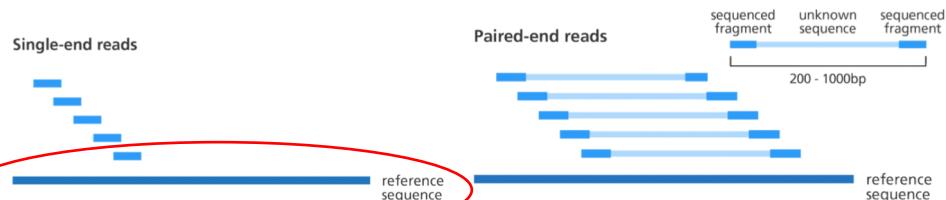


Image credit: Genome Research Limited

HUMAN reference sequences

Release name	Date of release	Equivalent UCSC version
GRCh38	Dec 2013	hg38
GRCh37	Feb 2009	hg19
NCBI Build 36.1	Mar 2006	hg18
NCBI Build 35	May 2004	hg17
NCBI Build 34	Jul 2003	hg16

MOUSE reference sequences

Release name	Date of release	Equivalent UCSC version
GRCm38	Dec 2011	mm10
NCBI Build 37	Jul 2007	mm9
NCBI Build 36	Feb 2006	mm8
NCBI Build 35	Aug 2005	mm7
NCBI Build 34	Mar 2005	mm6

The actual reference is just a (big) sequence (fasta) file: \$ ls -h GRCh38.fa
> 3.1G GRCh38.fa

Why align?

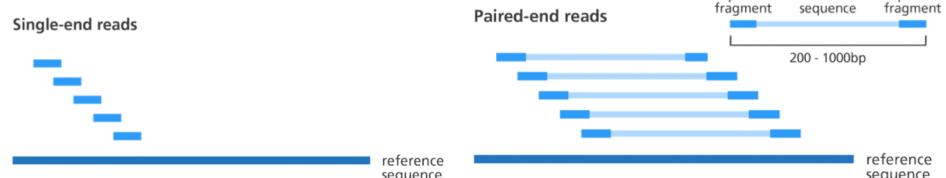
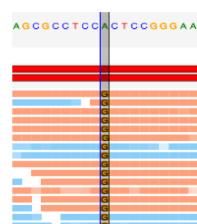
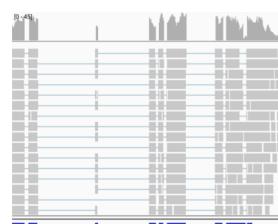


Image credit: Genome Research Limited

Align DNA: Identify variation



ALIGN RNA: Transcript abundance



Overview

Intro

Methods / Aligners

Alignment Outputs

Alignment Viewers

NGS Workflows, QC and BAM Improvement



Local alignment: Smith-Waterman algorithm (1981)

Algorithm for generating the optimal pairwise alignment between two sequences

Optimal alignment: the alignment which exhibits the most correspondences and the least differences, i.e. the alignment with the highest mapping score

Time consuming to carry out for every read - impractical

- Aligner will use it to refine a quick approximate placement
- Variant caller might use it to correctly re-align reads with insertions/deletions

EVEN SO, We will now get a feel for this

Local alignment: Smith-Waterman algorithm (1981)

We get a feel for this by aligning things by eye, then doing it using S-W:

QUERY SEQUENCES:
ATG

REFERENCE SEQUENCE:
ACTG

ALIGNMENT:
AC-G
|| |
ACTG

QUERY SEQUENCES:
ACCG

REFERENCE SEQUENCE:
ACTG

ALIGNMENT:
ACTG
|| |
ACTG

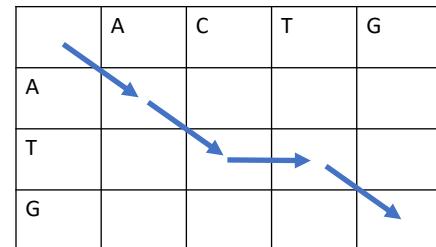
Local alignment: Smith-Waterman algorithm (1981)

We get a feel for this by aligning things by eye, then doing it using S-W:

QUERY SEQUENCES:
ATG

REFERENCE SEQUENCE:
ACTG

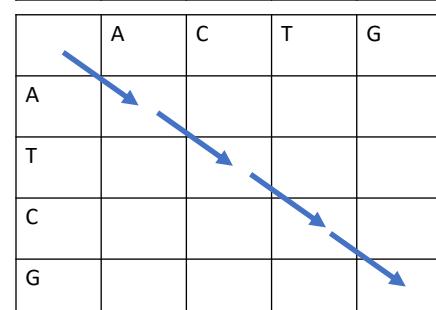
ALIGNMENT:
AC-G
|| |
ACTG



QUERY SEQUENCES:
ACCG

REFERENCE SEQUENCE:
ACTG

ALIGNMENT:
ACTG
|| |
ACTG



Local alignment: Smith-Waterman algorithm (1981)

Algorithm for generating the optimal pairwise alignment between two sequences:

Score each cell from top=>bottom, left=>right

Compare:

LEFT score = previous cell + gap penalty

TOP score = previous cell + gap penalty

DIAGONAL score = previous cell + match-mismatch

Match: +1

Mismatch: -1

Gap: -1

Each cell has three possible cumulative scores.

Put "top" cumulative score in current cell.

If top score is negative, replace with 0

		A	C	T	G
	0	0	0	0	0
A	0	-1, -1 1=1	-1, 0 -1=0	-1, -1 -1=0	-1, -1 -1=0
C	0	0 -1 0=0	-1 -1 2=2	-1 1,-1=1	-1 0, -1=0
G	0	-1 -1 -1=0	1 -1 -1=1	0 0 1=1	-1 -1 2 = 2

Record the source of top score (which cell you entered from)



Local alignment: Smith-Waterman algorithm (1981)

TRACEBACK:

Start at highest score in the matrix (2) and trace back to source until you get to 0 score.

		A	C	T	G
	0	0	0	0	0
A	0	-1, -1 1=1	-1, 0 -1=0	-1, -1 -1=0	-1, -1 -1=0
C	0	0 -1 0=0	-1 -1 2=2	-1 1,-1=1	-1 0, -1=0
G	0	-1 -1 -1=0	1 -1 -1=1	0 0 1=1	-1 -1 2 = 2



Local alignment: Smith-Waterman algorithm (1981)

WRITE OUT ALIGNMENT:

S1	A	C	T	G
S2	A	C	-	G

At each point of S1/S2: write a character or “-”

Entry from sides = gap (“-”)

Entry from diagonal = character

		S1				
		A	C	T	G	
		0	0	0	0	0
	A	0	-1, -1 1 = 1	0 -1 = 0	-1 -1 = 0	-1 -1 = 0
	C	0	0 -1 0 = 0	-1 2 = 2	1, -1 = 1	0, -1 = 0
	G	0	-1 -1 = 0	1 -1 = 1	0 1 = 1	-1 2 = 2



Local alignment: Smith-Waterman algorithm (1981)

Algorithm for generating the optimal pairwise alignment between two sequences

Optimal alignment: the alignment which exhibits the most correspondences and the least differences, i.e. the alignment with the highest mapping score

Time consuming to carry out for every read

- Only applied to a small subset of the reads that don't have an exact match
- Important for correctly aligning reads with insertions/deletions

We will now get a feel for this -- DONE !

Define TWO FURTHER CONCEPTS:

Local alignment: Any segment of one sequence can align to an arbitrary position in the other sequence. **Segments** of query align to **segments** of target.

Global alignment: Alignment of two complete sequences. *If sequences are very dissimilar in size then there will be lots of gaps.*

NGS read alignment

NGS: Nucleotide based alignment (also known as read mapping)

Number of 150bp reads in an 40x coverage Illumina X10 genome sequence?
 800×10^6

These all have to be aligned against a mammalian genome (3Gbp)

SMITH-WATERMAN is WAY TOO SLOW:

Two primary faster approaches:

- Hash-based alignment
- Suffix/prefix tries



Hash table alignment

Note: K-mer is a short fixed sequence of nucleotides

First thing: “wrap up” the reference genome to bring it to the reads – build a kmer hash

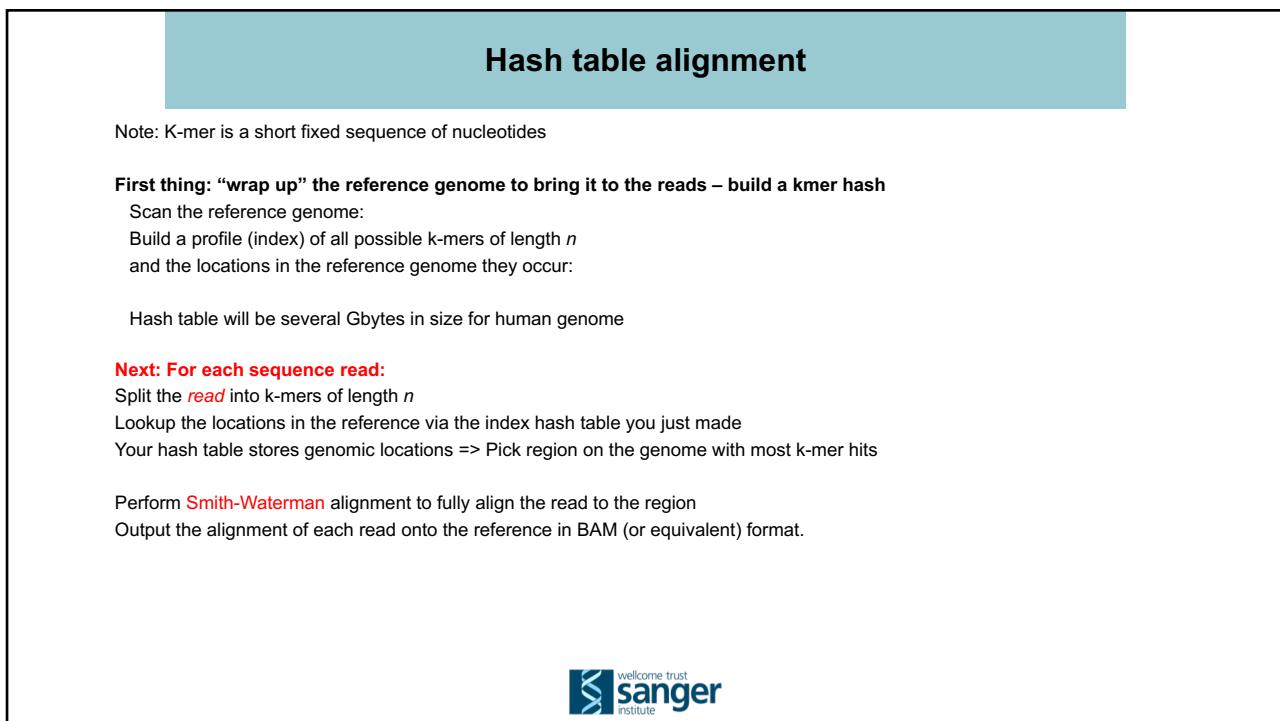
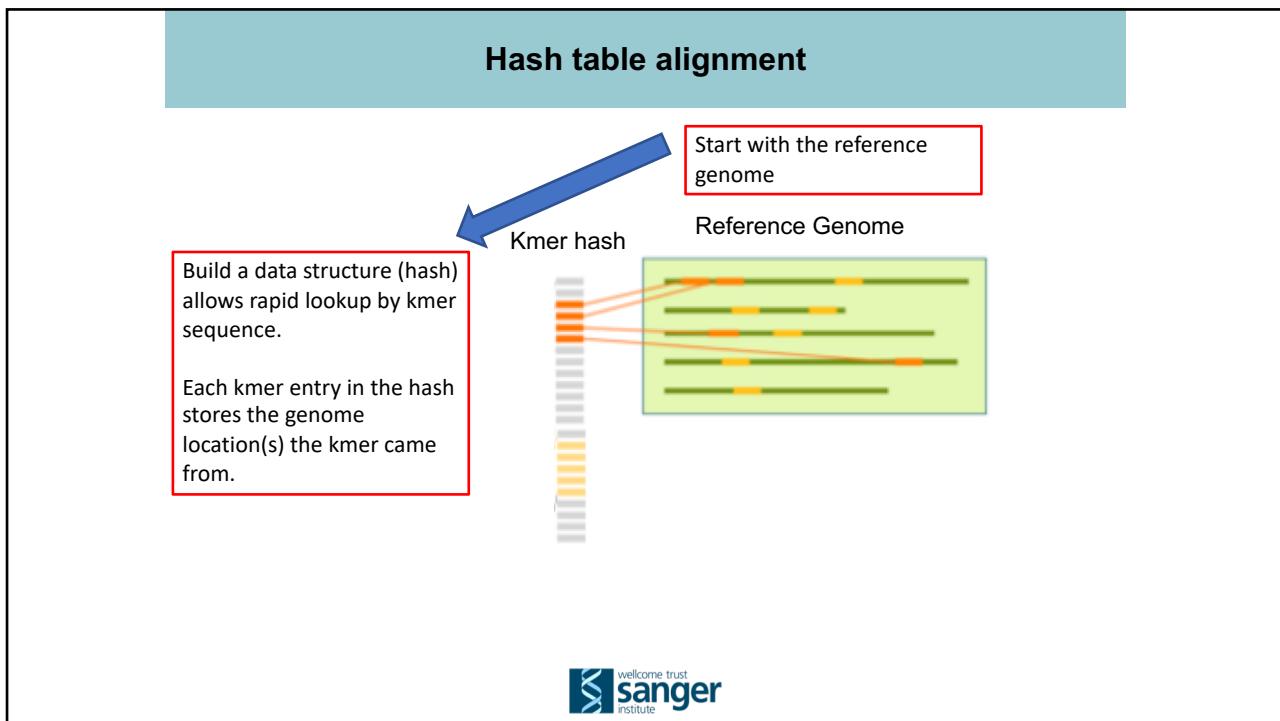
Scan the reference genome:

Build a profile (index) of all possible k-mers of length n
 and the locations in the reference genome they occur:

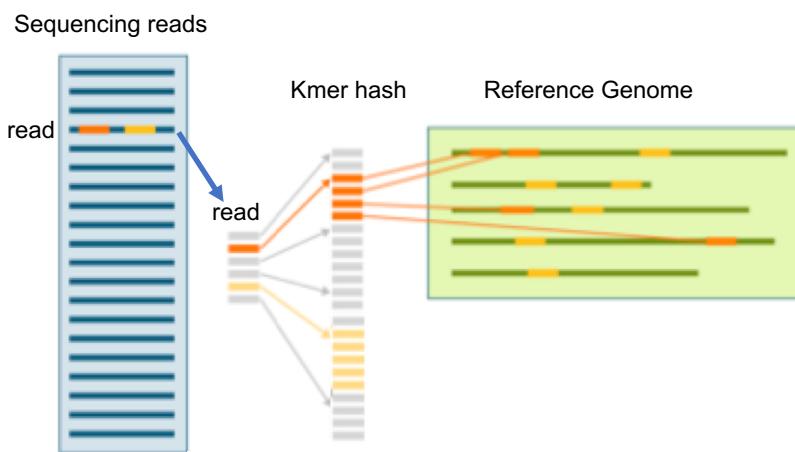
Hash table will be several Gbytes in size for human genome

sequence	ATGGAAGTCGGGAATC
7mers	ATGGAAG
	TGGAAGT
	CGAACGT
	GAAGATC
	AAGTCGC
	AGTCGCG
	GTCGGCG
	TCCGGGA
	CGCGGAA
	GCGGAAT
	CGGAATC

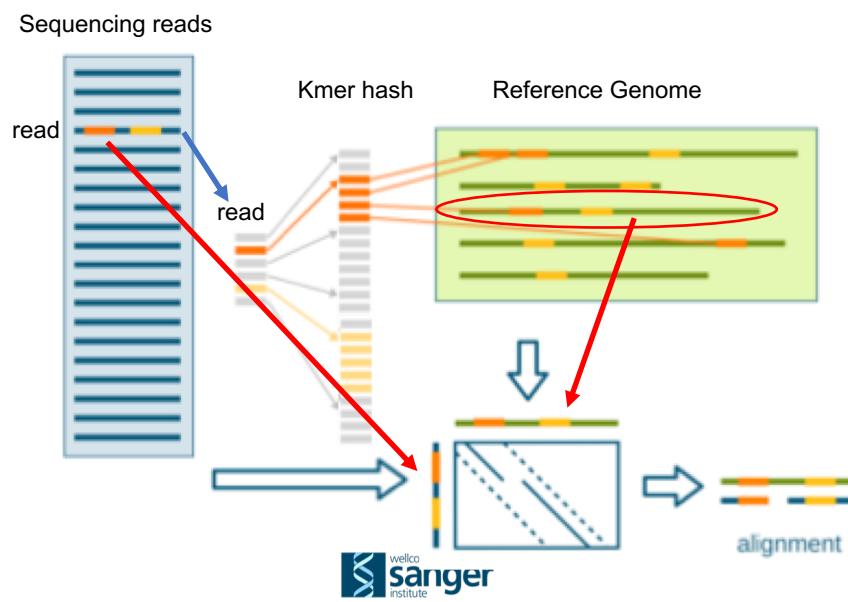




Hash table alignment



Hash table alignment



Hash table alignment

Note: K-mer is a short fixed sequence of nucleotides

First thing: “wrap up” the reference genome to bring it to the reads – build a kmer hash

Scan the reference genome:

Build a profile (index) of all possible k-mers of length n

and the locations in the reference genome they occur:

Hash table will be several Gbytes in size for human genome

Next: For each sequence read:

Split the read into k-mers of length n

Lookup the locations in the reference via the index (**seed phase**)

Pick region on the genome with most k-mer hits

Perform **Smith-Waterman** alignment to fully align the read to the region

Output the alignment of each read onto the reference in BAM (or equivalent) format.

Hash of the reads: MAQ, ELAND, ZOOM and SHRIMP

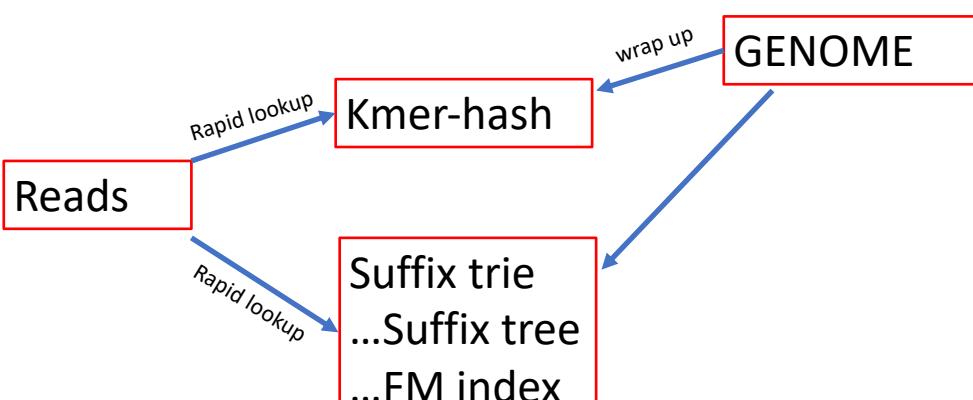
Smaller but more variable memory requirements

Hash the reference: SOAP, BFAST and MOSAIK

Advantage: constant memory cost

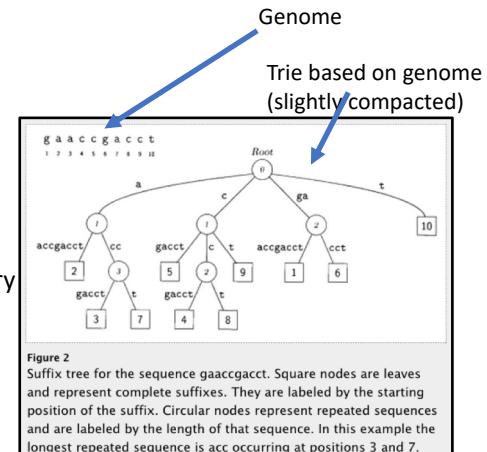


Wrapping up the genome and bringing it to your reads



Suffix/Prefix tree based aligners

For fast string matching: A suffix trie, or simply a trie, is a data structure that stores all the suffixes of a string, enabling fast string matching.



Build a suffix-trie for the genome:

- a different structure to bring the genome to the query
- But the memory requirements are huge

More terms: BW transform, FM-index ...

- Methods of reducing memory & time footprint

Examples: **BWA**, bowtie2, MUMMER



Delcher et al (1999) NAR

Mapping qualities

What if there are several possible places in the genome to align your sequencing read?

Genomes contain many different types of repeated sequences

- Transposable elements (40-50% of vertebrate genomes)
- Low complexity sequence
- Reference errors and gaps
- Plenty of closely paralogous genes!

The aligner will issue a MAPPING QUALITY to each alignment

Mapping quality is a measure of how confident the aligner is that the read is corresponds to this location in the reference genome

Typically represented as a phred score (log scale)

Q0 = read placed with identical score elsewhere

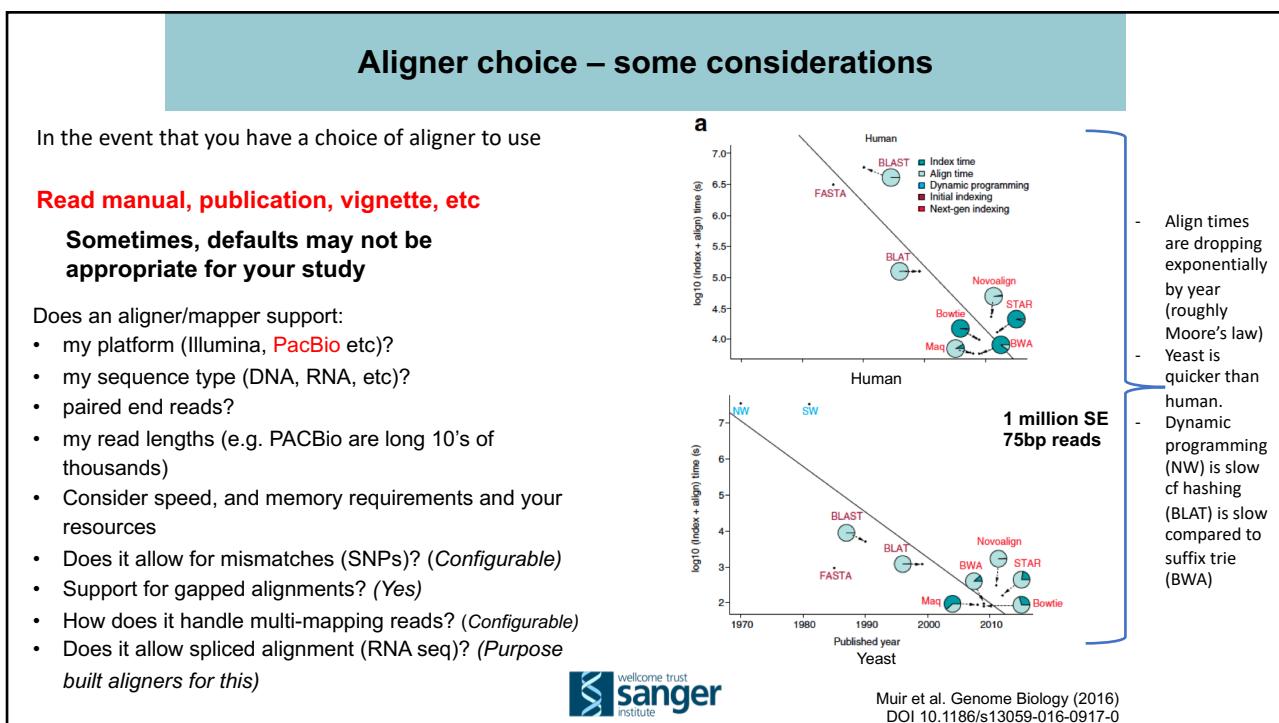
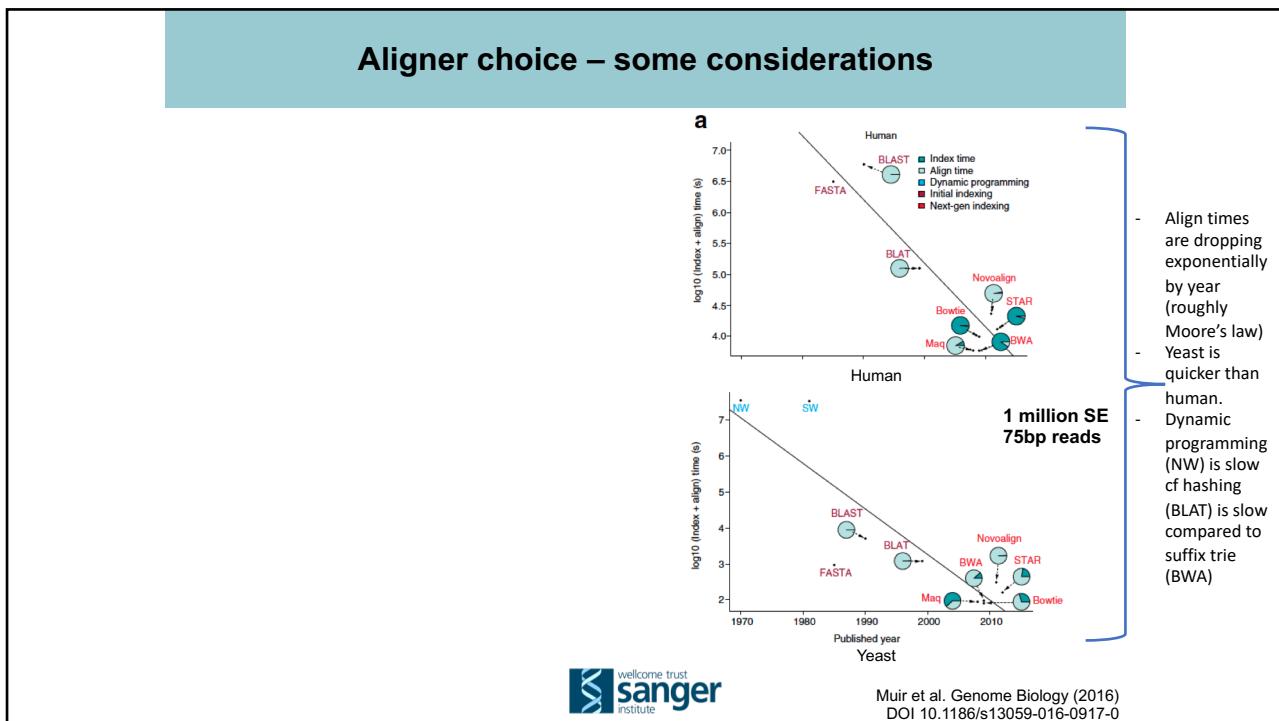
Q10 = 1 in 10 probability of being incorrect

Q20 = 1 in 100 probability of being incorrect

Paired-end sequencing is useful - if one end maps inside a repetitive elements and one outside in unique sequence: the **the aligner will use this**

Hence prefer paired-end sequencing





Some alignment options (there are lots more)								
	Aligner	Gapped alignment	SE and/or PE	Input format	Output format	Trimming?	BS-seq	Note
careful	BWA-MEM	Yes	SE, PE	FASTQ, FASTA	SAM	No	No	Seed + extend: Local alignment
Unspliced: No intron-sized gaps	Bowtie	No	SE, PE	FASTQ, FASTA	SAM	Yes	No	Mismatches < 3
	Bowtie2	Yes	SE, PE	FASTQ, FASTA qseq	SAM	Yes	No	Seed + extend: Local alignment
Spliced: Allow for big gaps inside reads, are aware of gene-structure	TopHat	Yes	SE, PE	FASTQ, FASTA qseq	SAM	Yes	No	Uses bowtie or bowtie2 as base aligner
	STAR	Yes	SE, PE	FASTQ, FASTA	SAM	Yes	No	Fastest and most accurate for RNAseq

Don't use old tech

Spliced aligns (RNAseq)

Independently performs spliced aligns (no preprocessing)

Alignments – scaling up

Question:
 IF: 50Mbp bwa mem = 5 CPU minutes.
 AND: 1 human genome sequenced at 45x = 150 Gbp per HiSeqX10 lane
 THEN: How long will it take to align your 45x human genome sequencing?



Alignments – scaling up

Question:

IF: 50Mbp bwa mem = 5 CPU minutes.

AND: 1 human genome sequenced at 45x = 150 Gbp per HiSeqX10 lane

THEN: How long will it take to align your 45x human genome sequencing?

ANSWER:

How many lots of 50Mb in 150Gb ?

$$150\text{Gb} / 50\text{Mb} = (150 \times 10^9) / (50 \times 10^6) = 3 \times 10^3 \text{ lots}$$

Each lot takes 5 minutes => total time is $5 \times 3 \times 10^3$ CPU minutes

15000 CPU minutes = 250 hours ~ 10 days **ouch.**



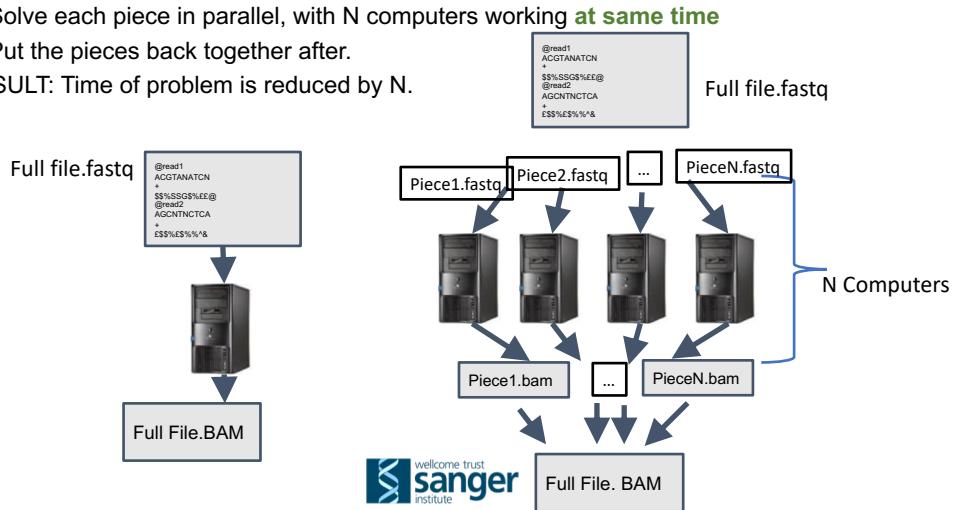
Alignments – scaling up

This is why we like compute clusters!

AS ALWAYS: The strategy is to:

- Chop up the problem into small pieces and
- Solve each piece in parallel, with N computers working **at same time**
- Put the pieces back together after.

RESULT: Time of problem is reduced by N.



Alignments – scaling up

Question:

IF: 50Mbp bwa mem = 5 CPU minutes.

AND: 1 human genome sequenced at 45x = 150 Gbp per HiSeqX10 lane

THEN: How long will it take to align your 45x human genome sequencing?

ANSWER:

How many lots of 50Mb in 100Gb ?

$$150\text{Gb} / 50\text{Mb} = (150 \times 10^9) / (50 \times 10^6) = 3 \times 10^3$$

Each lot takes 5 minutes => total time is $5 \times 3 \times 10^3$ CPU minutes

15000 CPU minutes = 250 hours ~ **10 days ouch.**

Compute Cluster with N nodes: Time of problem is reduced by N.

E.g. if you have dedicated 200 nodes, the time for alignment drops to ~ 1.25 hrs

- Which is ok for a single sample.
- If you have 1000 samples, you need more nodes + better optimisation.
- At WTSI, alignment is more much more efficient (dedicated resources)



Overview

Intro

Methods / Aligners

Alignment Outputs

Alignment Viewers

NGS Workflows, QC and BAM Improvement



LET'S HAVE A BREAK



Shutterstock.com • 152027796

Overview

Intro

Methods / Aligners

Alignment Outputs

Alignment Viewers

NGS Workflows, QC and BAM Improvement



Pre-alignment file formats

Data generation



FASTQ

The same 4 line format for paired reads (PE sequencing)

```
@IL16_4408:3:5:17860:13258
CAGTTTTTCAGAGCAGTAGCCATTAGGCACAAATGTGATT
+
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFAFBFFF
@IL16_4408:3:14:13276:1210
AGTCACAGATGCCCTTGAGCTTAAGAATTTCAGCAGAAG
+
FFFFFFFFFFFFFFFFFFDFFFFFFFFFFFFFDCF
@IL16_4408:3:83:5210:21157
GGGCCATGTCGTCCTCCGCACTGAAGACCATGCTAT
```



Post-alignment file formats

SAM (Sequence Alignment/Map) format

- Single unified format for storing read alignments to a reference genome
 - Developed by the 1000 Genomes group in 2009

BAM (Binary Alignment/Map) format

- Binary equivalent of SAM
 - Developed for fast processing/indexing
 - Block GZIP compression for random access of regions

Key features

- Can store alignments from most aligners
 - Supports multiple sequencing technologies
 - Supports indexing for quick retrieval/viewing
 - Compact size (e.g. 112Gbp Illumina = 116Gbytes disk space)
 - Reads can be grouped into logical groups e.g. lanes, libraries, samples
 - Widely support by variant calling software packages

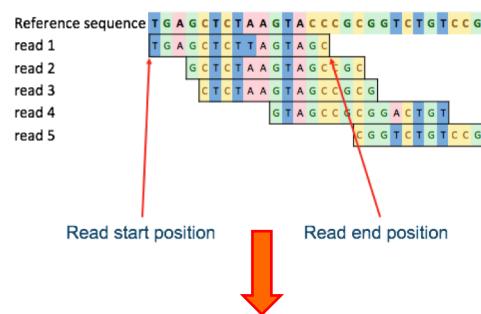
Specification maintained by the Global Alliance for Genomics and Health



Reference based compression

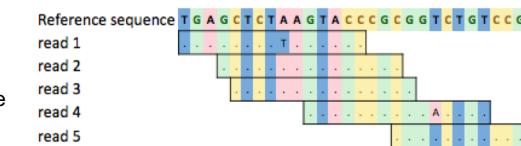
BAM stores all of the data

- Every read base
- Every base quality
- Using a single conventional compression technique for all types of data



CRAM: Important concepts

- Reference based compression
- Controlled loss of quality information
- Different compression methods to suit the type of data, e.g. base qualities vs. meta-data vs. extra tags



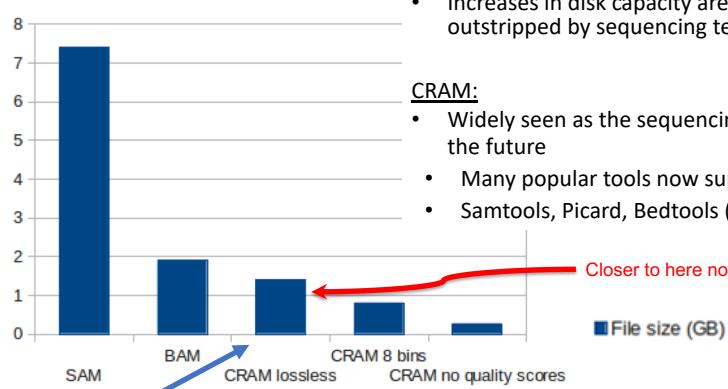
CRAM format

WHY CRAM?

- BAM files are too large
 - ~1.5-2 bytes per base pair
- Increases in disk capacity are being far outstripped by sequencing technologies

CRAM:

- Widely seen as the sequencing format of the future
- Many popular tools now support CRAM
- Samtools, Picard, Bedtools (in the future)



Sanger stores everything as lossless cram



SAM/BAM format

SAM fields

Col	Field	Type	Regexp/Range	Brief description
1	QNAME	String	[!-?A-^]{1,254}	Query template NAME
2	FLAG	Int	[0,2 ¹⁶ -1]	bitwise FLAG
3	RNAME	String	* ![!-()+-<>-][!-]*	Reference sequence NAME
4	POS	Int	[0,2 ³¹ -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 read
8	PNEXT	Int	[0,2 ³¹ -1]	Position of the mate/next read
9	TLEN	Int	[-2 ³¹ +1,2 ³¹ -1]	observed Template LENgth
10	SEQ	String	* [A-Za-z.=.]+	segment SEQuence
11	QUAL	String	[!-]+	ASCII of Phred-scaled base QUALity+33

CIGAR string

Cigar has been traditionally used as a compact way to represent a sequence alignment: to go "FROM one string TO another you perform these operations".

Operations include

M - match or mismatch
I - insertion
D - deletion
S - soft clip (ignore these bases)
H - hard clip (ignore and remove these bases)

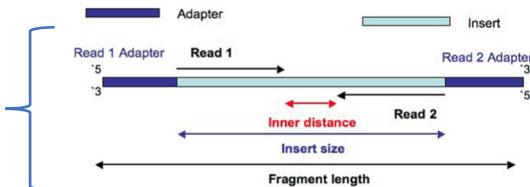
E.g. Read: ACGCA-TGCAGTtagacgt
Ref: ACTCAGTG--GT
Cigar: 5M 1D 2M 2I 2M 7S

INTERLUDE – PAIRED END SEQUENCING PARAMETERS

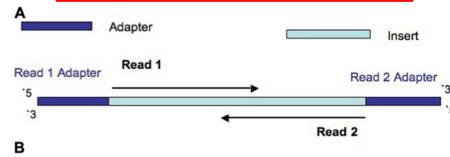


Fragment length, insert size and inner mate distance

Insert size > length(read 1 + read 2)



Insert size < length(read 1 + read 2)



Insert size < length(read 1);
Insert size < length(read 2)

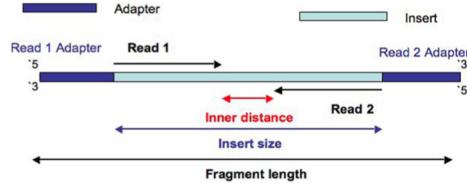
Turner, 2014. PMID:24523726



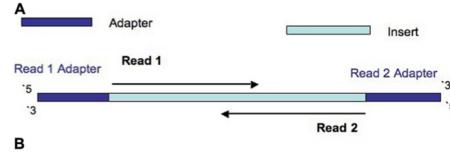
Fragment length, insert size and inner mate distance

Insert size > length(read 1 + read 2)

Both
reads get
the same
“name”



Insert size < length(read 1 + read 2)



Insert size < length(read 1);
Insert size < length(read 2)

Turner, 2014. PMID:24523726



NOW WE RESUME OUR NORMAL PROGRAMMING



SAM/BAM/CRAM format in practice

```
HX8_24050:5:2103:20770:9449 83 7 4189207 0 151M = 4188954 -404 GTTG <JJJ MC:Z:151M MD:Z:151 NM:i:0 AS:i:151 XS:i:151
...
HX8_24050:5:2103:20770:9449 163 7 4188954 0 151M = 4189207 404 TTGG AAFF MC:Z:151M MD:Z:151 NM:i:0 AS:i:151 XS:i:151
...
HX8_24050:3:2212:32319:38772 99 7 4188947 0 151M = 4189300 436 ATGG AAAF MC:Z:68S83M MD:Z:151 NM:i:0 AS:i:151 XS:i:151
...
HX8_24050:3:2212:32319:38772 147 7 4189300 0 68S83M = 4188947 -436 ACAG --7- MC:Z:151M MD:Z:5C10T4T61 NM:i:3 AS:i:68 xs:i:75
```

QNAME: read id
- matches fastq

FLAG:
> samtools flags 83: PAIRED, PROPER_PAIR, REVERSE, READ1
> samtools flags 163: PAIRED, PROPER_PAIR, MREVERSE, READ2

RNAME, POS – chromosome & base pair position

MAPQ mapping quality
CIGAR M? S? I? D?

RNEXT, PNEXT - paired read chr & position
RNEXT “=” means same as first read
PNEXT: read2 aligns 5' of read1.
TLEN: insert size

SEQunce
QUALity (alignment doesn't use them, but variant callers will)

ALIGNMENT TAGS:
<http://www.samformat.info/sam-format-alignment-tags>

MC:mate cigar
MD:string for mismatching positions
NM>Edit distance to the reference (no clipping)
AS:Aligner's Alignment score
XS:suboptimal alignment score (aha!!)

SAM/BAM/CRAM format in practice

```
HX8_24050:5:1217:22455 83 7 87483848 60 118M1I32M = 87483669 -329 TG.. FJ.. MC:Z:151M MD:Z:150 NM:i:1 AS:i:143 XS:i:0
```

QNAME:

FLAG

RNAME

POS

MAPQ

CIGAR

RNEXT, PNEXT

RNEXT

PNEXT

TLEN: can we work this out from the POS and PNEXT?

SEQ:

QUAL:

MC: how does the mate match?

MD: Does the mate have a mismatch?

NM: This read: what is the edit distance?

AS: what is the current alignment score?

XS: what is the align score of the next-best-hit?

Overview

Intro

Methods / Aligners

Alignment Outputs

Alignment Viewers

NGS Workflows, QC and BAM Improvement



Three viewers, in order of sophistication

Context:

You have a mouse.

You extracted DNA from mouse kidney and performed paired-end Whole-Genome Sequencing.

Core sequencing facility (and/or an informatician and / or you) produces the alignments and summary stats.

They do variant calling! You find some variants you like, or some differentially expressed genes.

YOU HAVE TO LOOK AT YOUR ALIGNMENTS.

You will need:

Genome file (ie the reference)

Genome file: GRCm38.fa. Remember, there's always a reference back there ...

Gene annotations.

Alignment file: **MySample.bam**

Moral

... You need to have a look at it. You cannot just trust a spreadsheet of variants given to you by someone else.



Viewer 1: samtools

You could just use samtools **view** at the command line.

- This shows you the bam header: chromosomes/contigs, read-groups, programs which have acted
- This shows you each read id and CIGAR string, as well as mate structure (as we saw before)

\$ samtools view -H MySample.bam

```
@HD VN:1.5 SO:coordinate
@SQ SN:1 LN:195471971 AS:GRCm38
@SQ SN:10 LN:130694993 AS:GRCm38
...
@SQ SN:X LN:171031299 AS:GRCm38
@SQ SN:Y LN:91744698 AS:GRCm38
...
@RG ID:332872 PL:ILLUMINA SM:MD5638a DS:WGS_ILLUMINA_short PU:24050_1
@RG ID:332873 PL:ILLUMINA SM:MD5638a DS:WGS_ILLUMINA_short PU:24050_2
...
@PG ID:basecalling_0 PN:Unknown PP:SCS_0 DS:Basecalling Package VN:Unknown
@PG ID:bambi_0 PN:bambi CL:/software/solexa/pkg/bambi/0.9.11/bin/bambi i2b ....
@PG ID:bamadapterfind_0 ...
@PG ID:bwa_0 PN:bwa CL:/software/solexa/pkg/bwa/ ...

HX8_24050:4:2105:31456:16147 99 1 3000000 40 20M1I104M26S = 3000301 45 ...
```



Viewer 1: samtools

You could just use samtools at the command line.

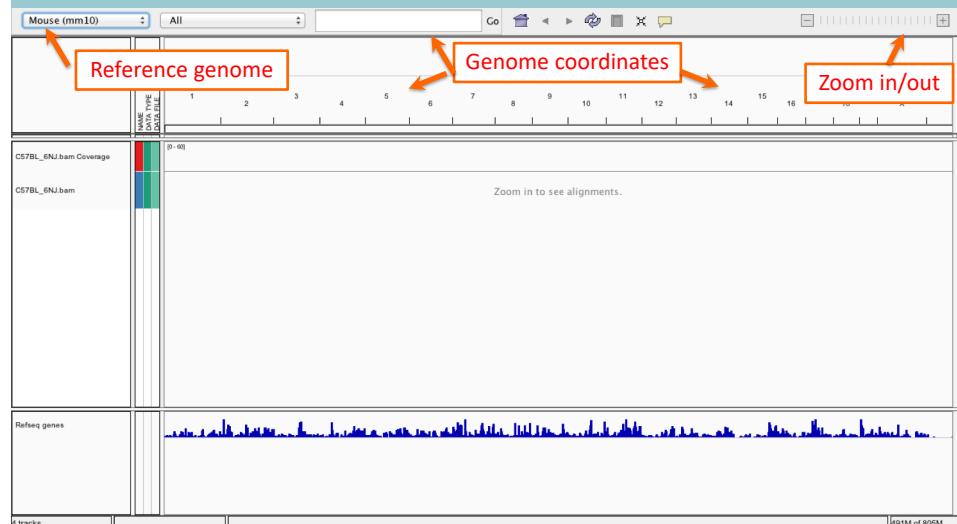
- This shows you the bam header.
- This shows you each read id and CIGAR string, as well as mate structure.
- But otherwise it's a pretty poor way of visualizing anything.
- No feel for relationship of pairs or how multiple reads show the same signal

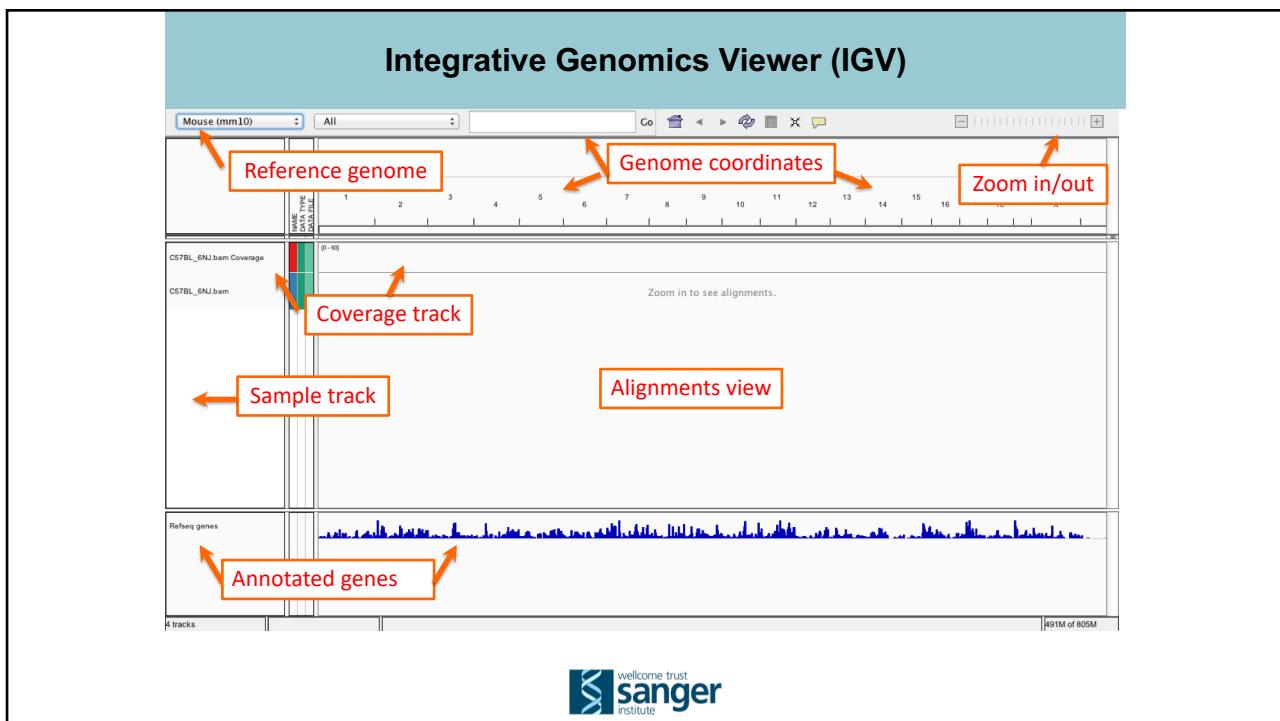
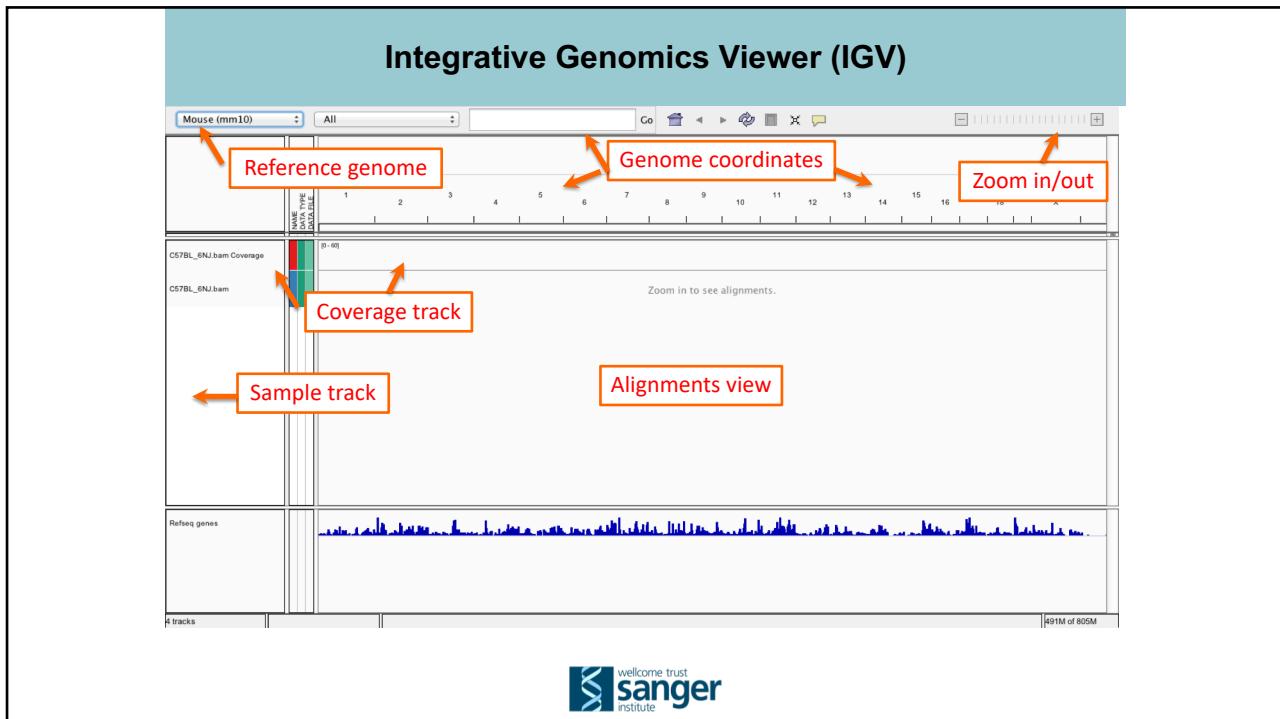
The whole point of short-read sequencing is to infer variation etc from PILEUP – the cooperative action of lots of reads together, all having the same variation at the same point
Samtools doesn't cut it.

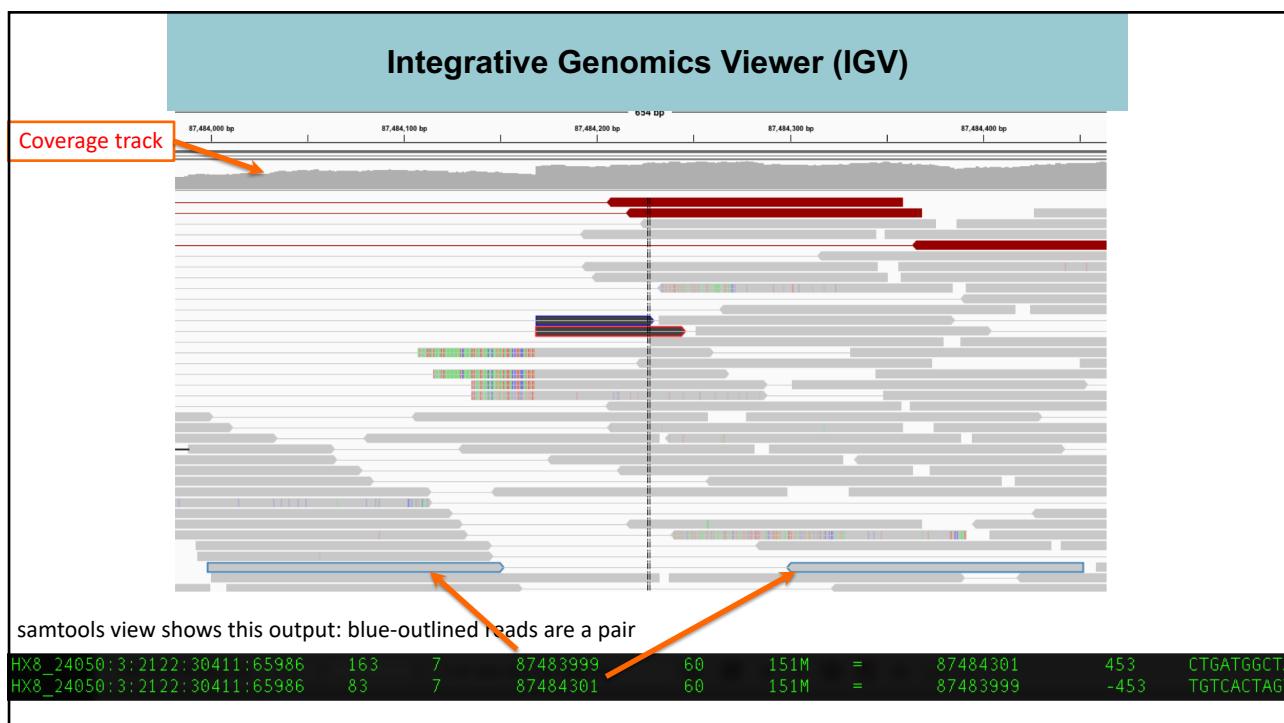
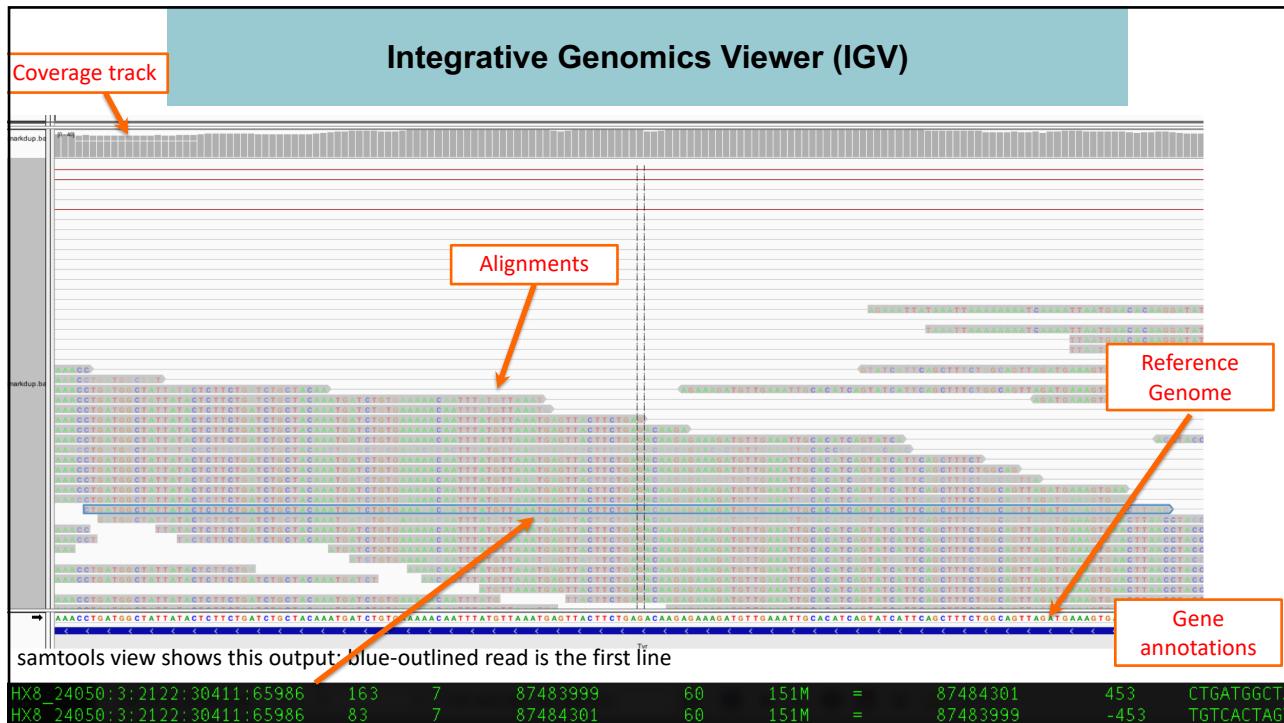
```
HX8_24050:5:2103:20770:9449 83 7 4189207 0 151M = 4188954 -404 GTTG <JJJ MC:Z:151M MD:Z:151 NM:i:0 AS:i:151 XS:i:151
...
HX8_24050:5:2103:20770:9449 163 7 4188954 0 151M = 4189207 404 TTGG AAFF MC:Z:151M MD:Z:151 NM:i:0 AS:i:151 XS:i:151
...
HX8_24050:3:2212:32319:38772 99 7 4188947 0 151M = 4189300 436 ATGG AAAF MC:Z:68S83M MD:Z:151 NM:i:0 AS:i:151 XS:i:151
...
HX8_24050:3:2212:32319:38772 147 7 4189300 0 68S83M = 4188947 -436 ACAG --7- MC:Z:151M MD:Z:5C10T4T61 NM:i:3 AS:i:68 XS:i:75
```

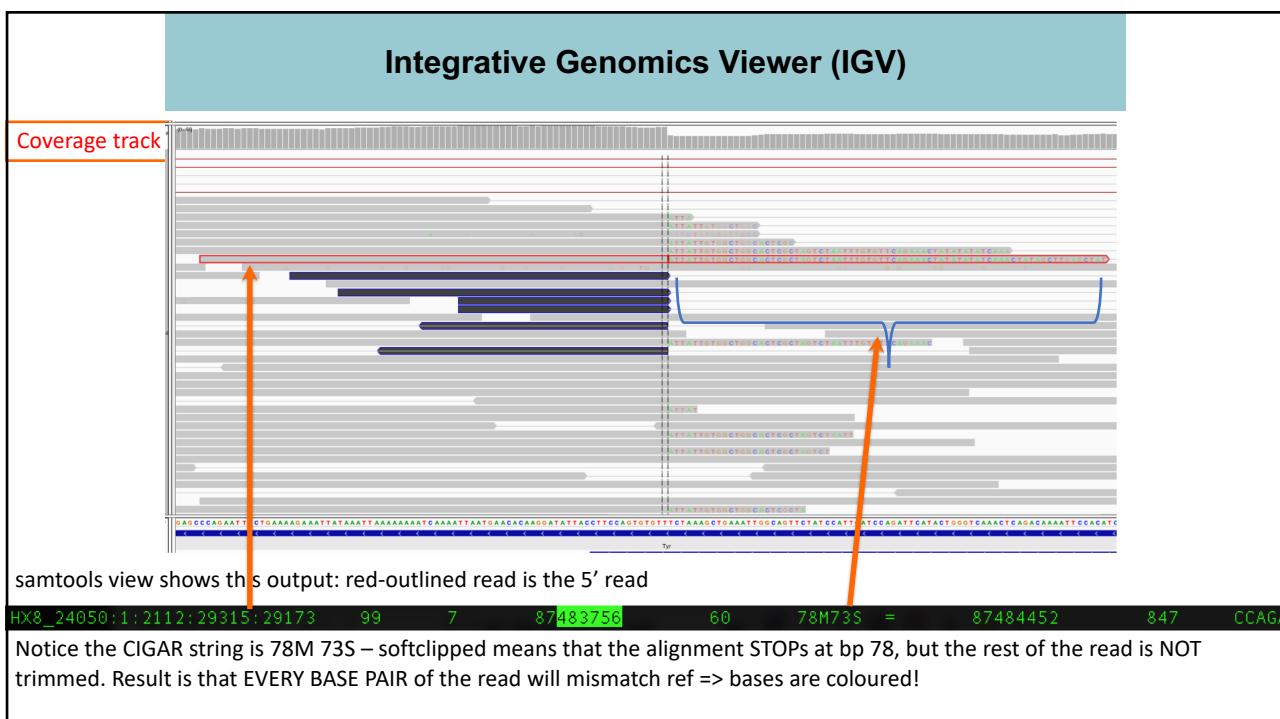
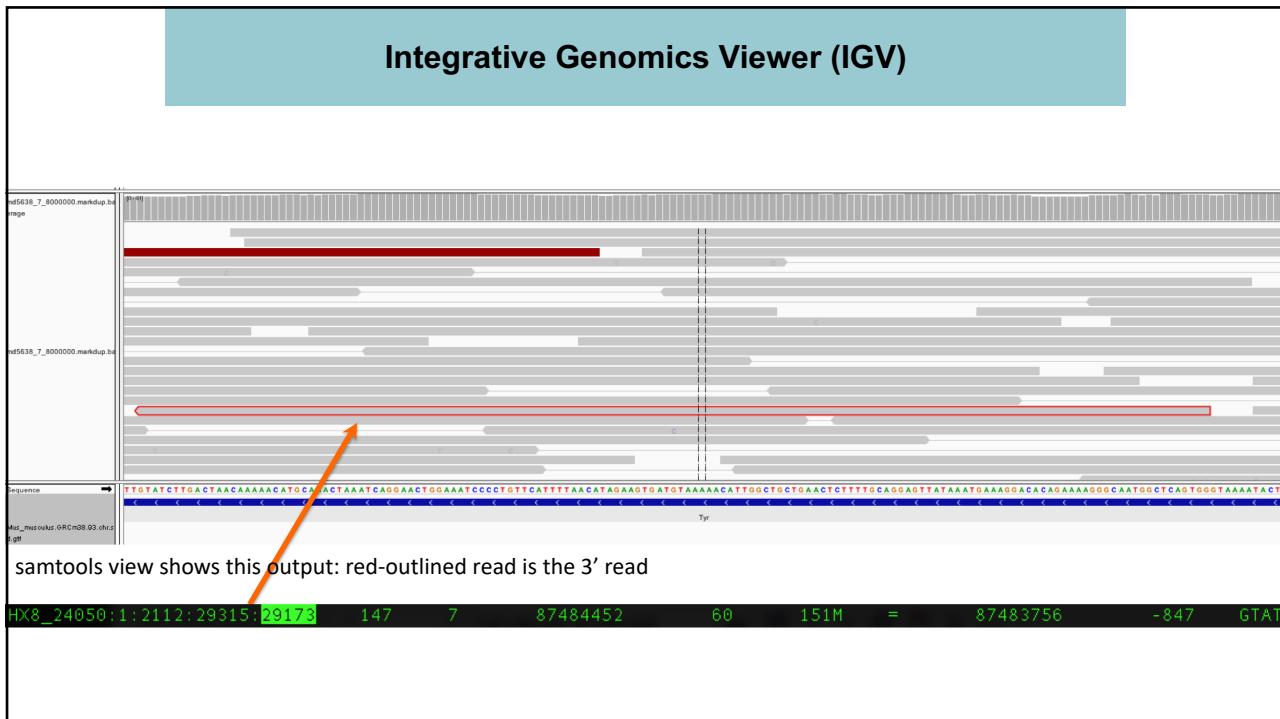


Option 3: Integrative Genomics Viewer (IGV)









Integrative Genomics Viewer (IGV)

The screenshot shows a genomic track with three reads aligned to a reference genome. The reads are shown as red bars with orange arrows pointing to them. Below the track is a table of read alignments:

HX8_24050:1:2112:29315:29173	99	7	87483756	60	78H73S	=	87484452	847	CCAGA
HX8_24050:1:2112:29315:29173	2147	7	87484179	60	76H75M	=	87484452	434	TTATT
HX8_24050:1:2112:29315:29173	147	7	87484452	60	151M	=	87483756	-847	GTAT

But wait! There's more

samtools view will actually show three lines in the file with the same read name.

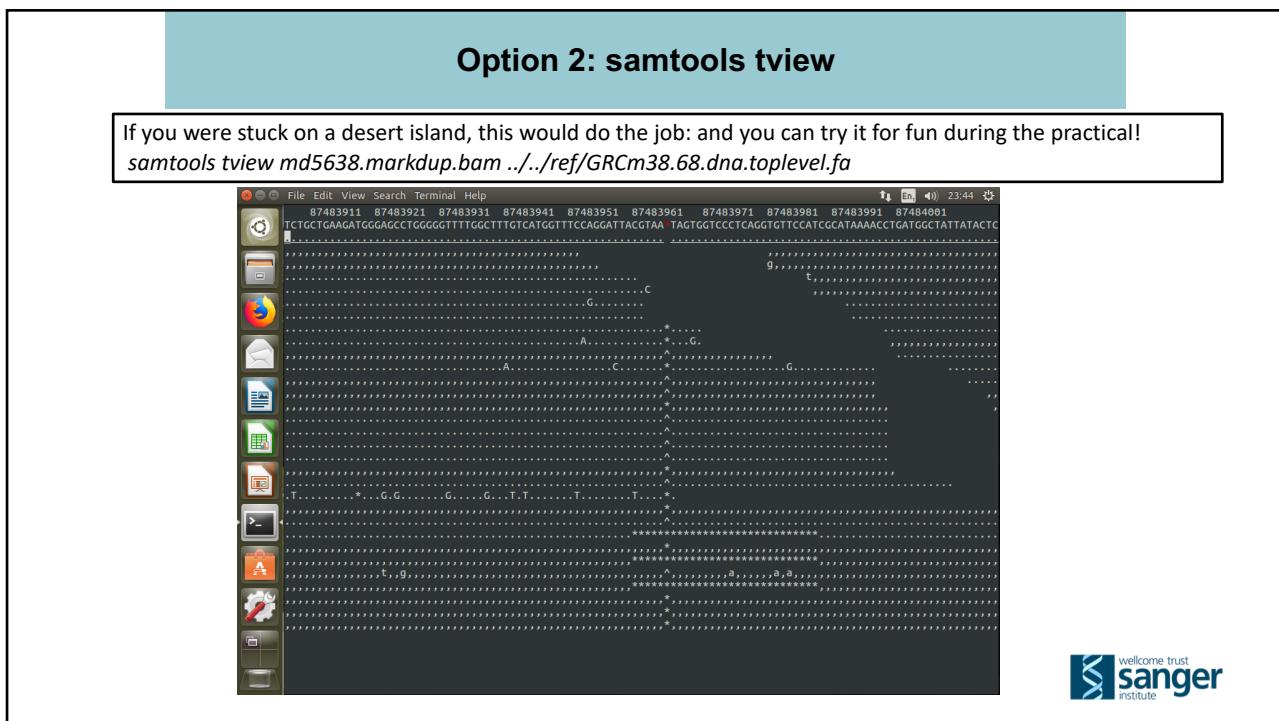
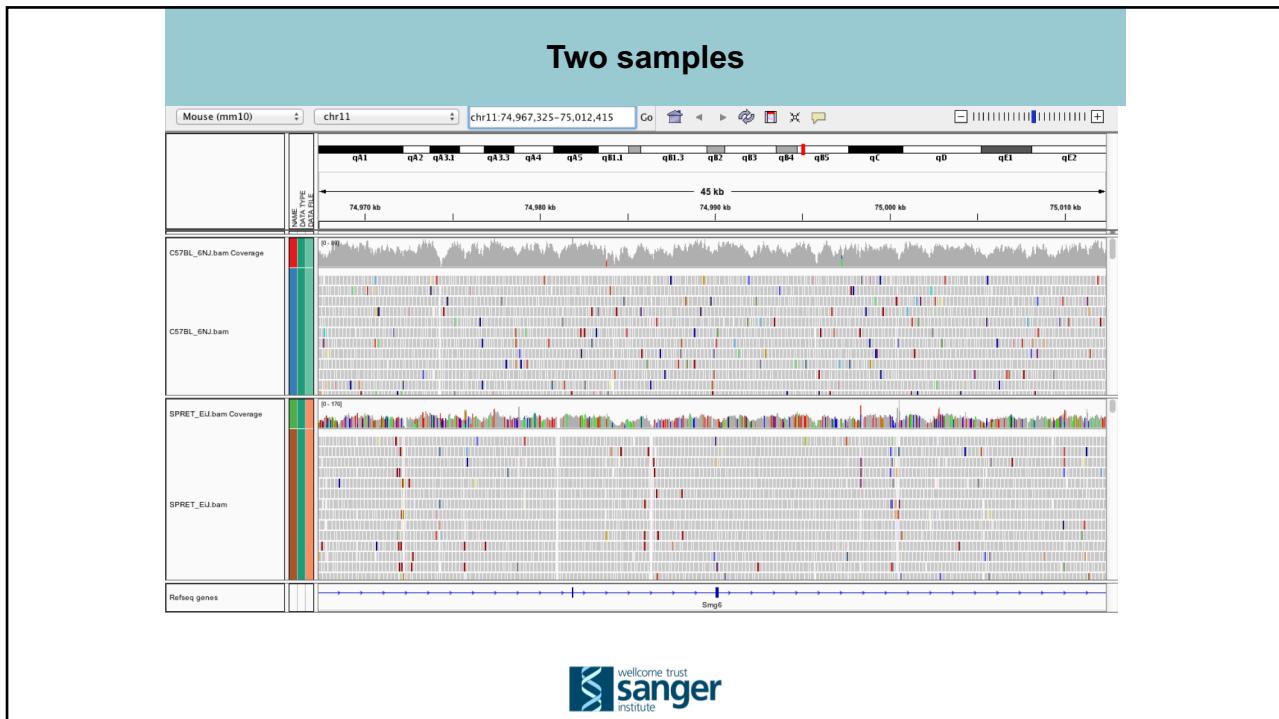
We have examined the left read (softclipped) and the right read (perfectly aligned).

The middle read alignment is the 'right half' of the left read. It is hard-clipped and marked as 'secondary' – see flags

Viewing reads as pairs

The screenshot shows a genomic track with multiple reads aligned to a reference genome. The reads are shown as red bars with orange arrows pointing to them. Below the track is a table of read alignments:

HX8_24050:1:2112:29315:29173	99	7	87483756	60	78H73S	=	87484452	847	CCAGA
HX8_24050:1:2112:29315:29173	2147	7	87484179	60	76H75M	=	87484452	434	TTATT
HX8_24050:1:2112:29315:29173	147	7	87484452	60	151M	=	87483756	-847	GTAT



Overview

Intro

Methods / Aligners

Alignment Outputs

Alignment Viewers

NGS Workflows, QC and BAM Improvement



LET'S HAVE A BREAK



SHUTTERSTOCK.COM • 152027796

Overview

Intro

Methods / Aligners

Alignment Outputs

Alignment Viewers

NGS Workflows, QC and BAM Improvement

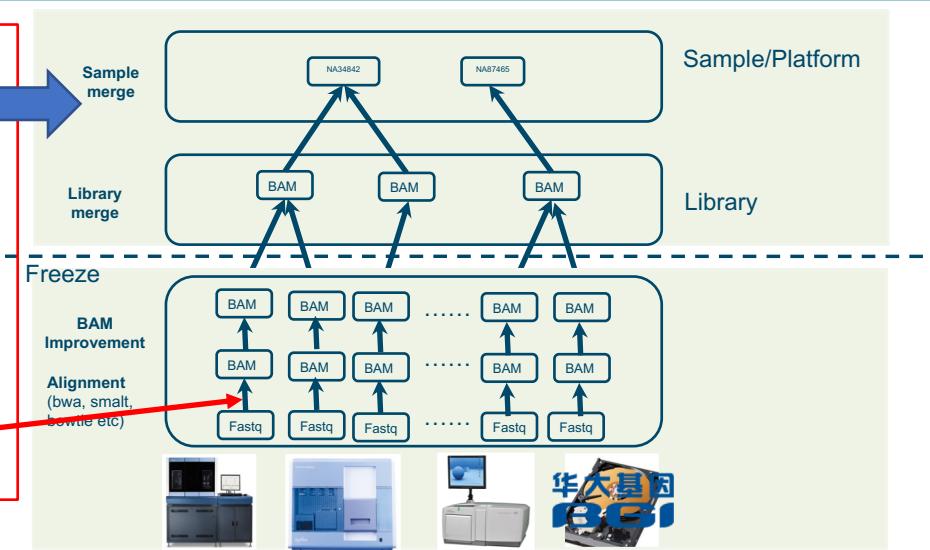


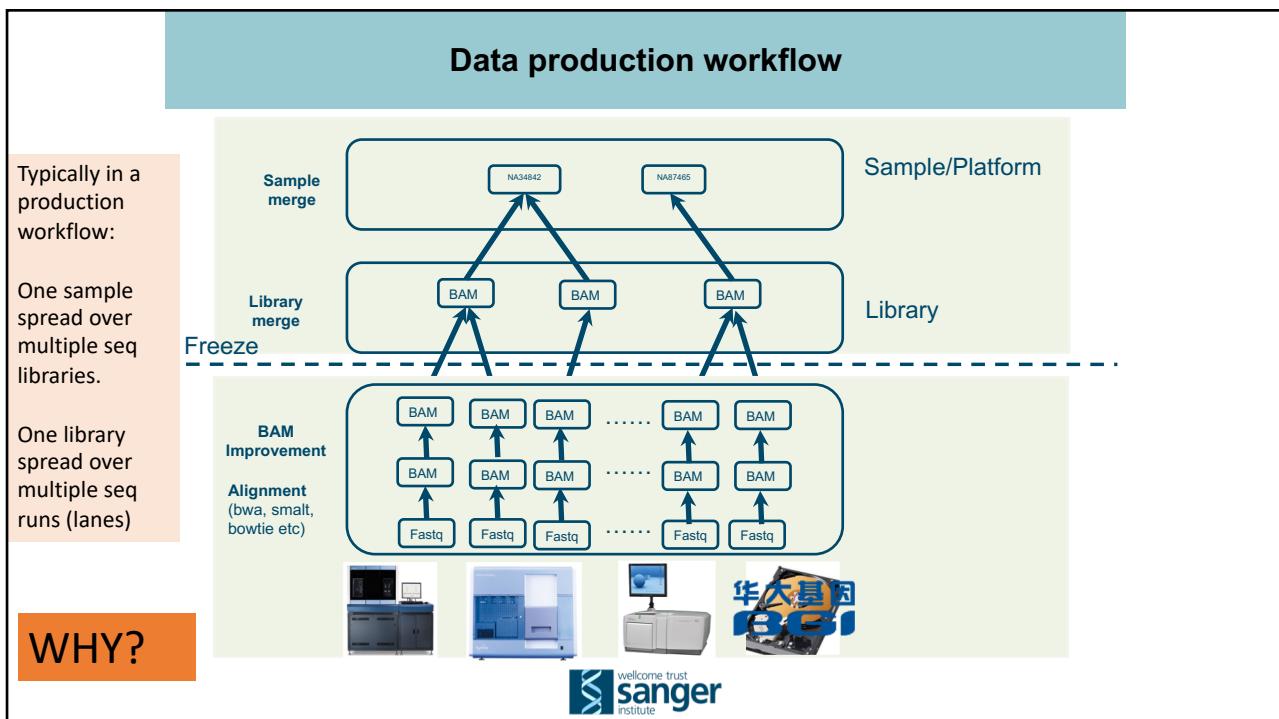
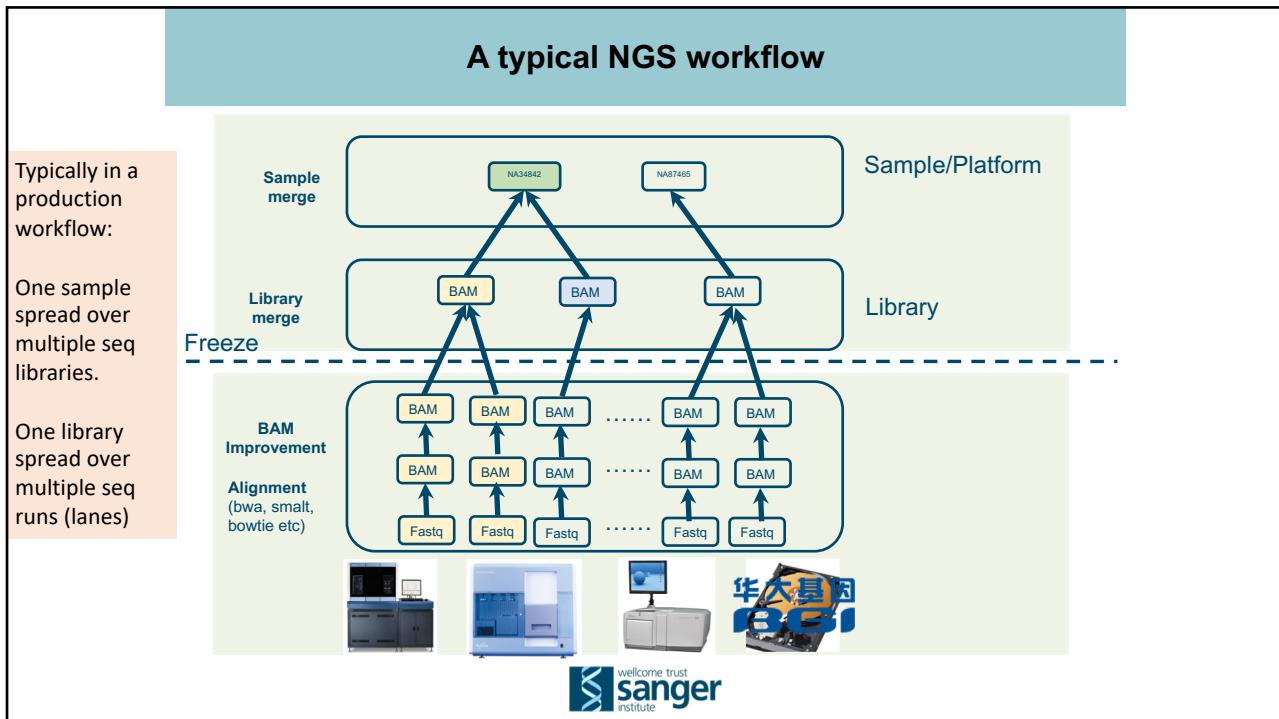
A typical NGS workflow

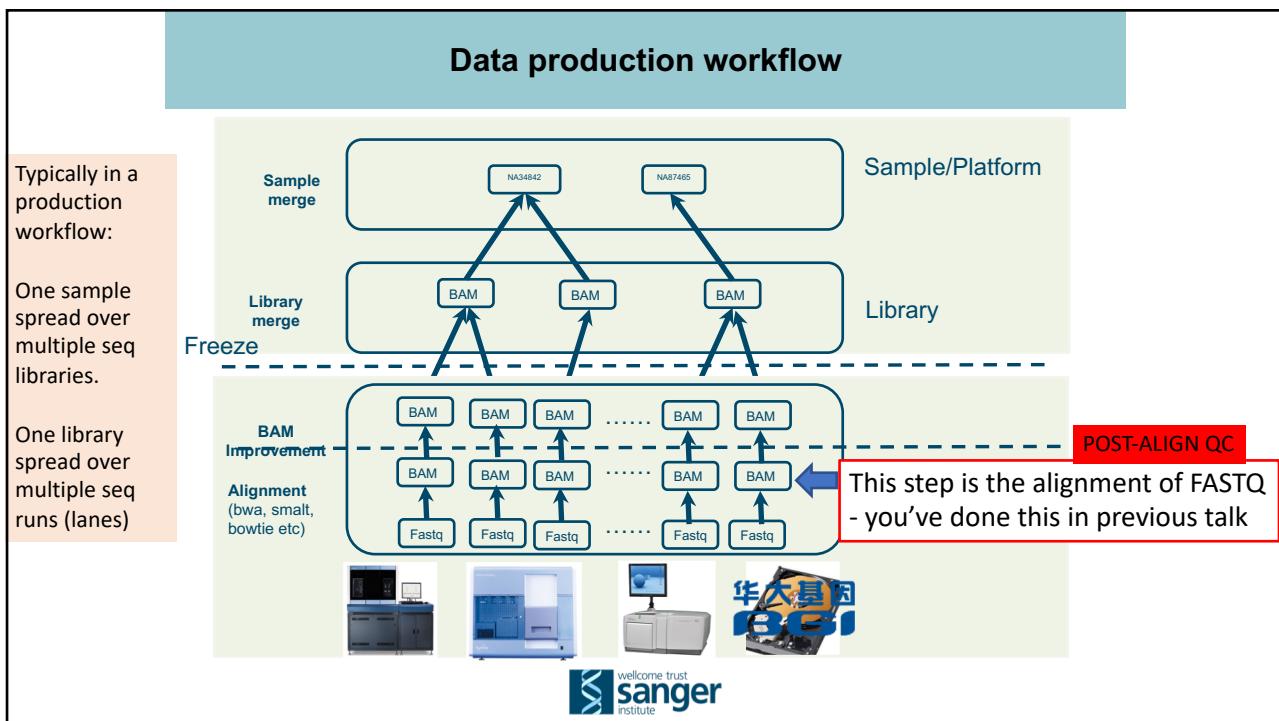
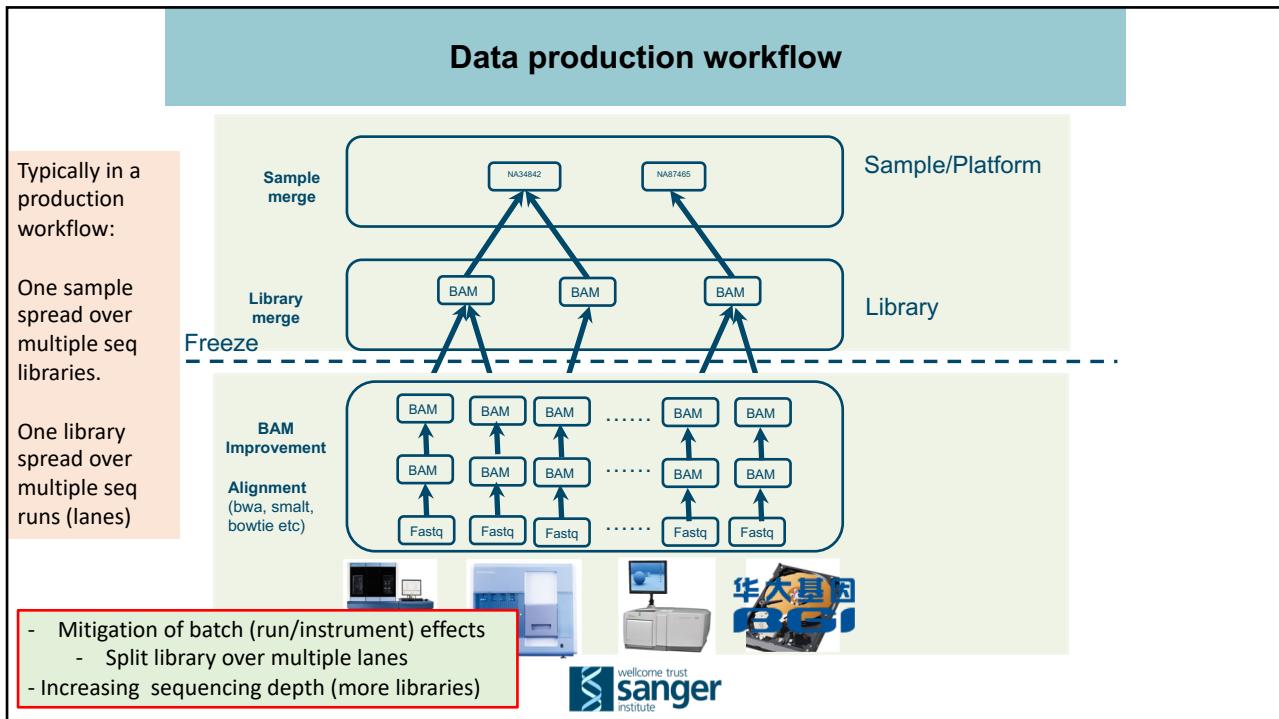
There might be
10's, 100's or
thousands of
samples.

This needs a
coordinated
workflow:

Alignment is just
one part







Data QC from alignments – a brief revision

```
 samtools stats my_aligned_reads.bam > my_aligned_reads.stats.txt  
 plot-bamstats my_aligned_reads.stats.txt
```

Already covered by the QC lecture: several useful metrics to assess the quality of your data and alignments produced:

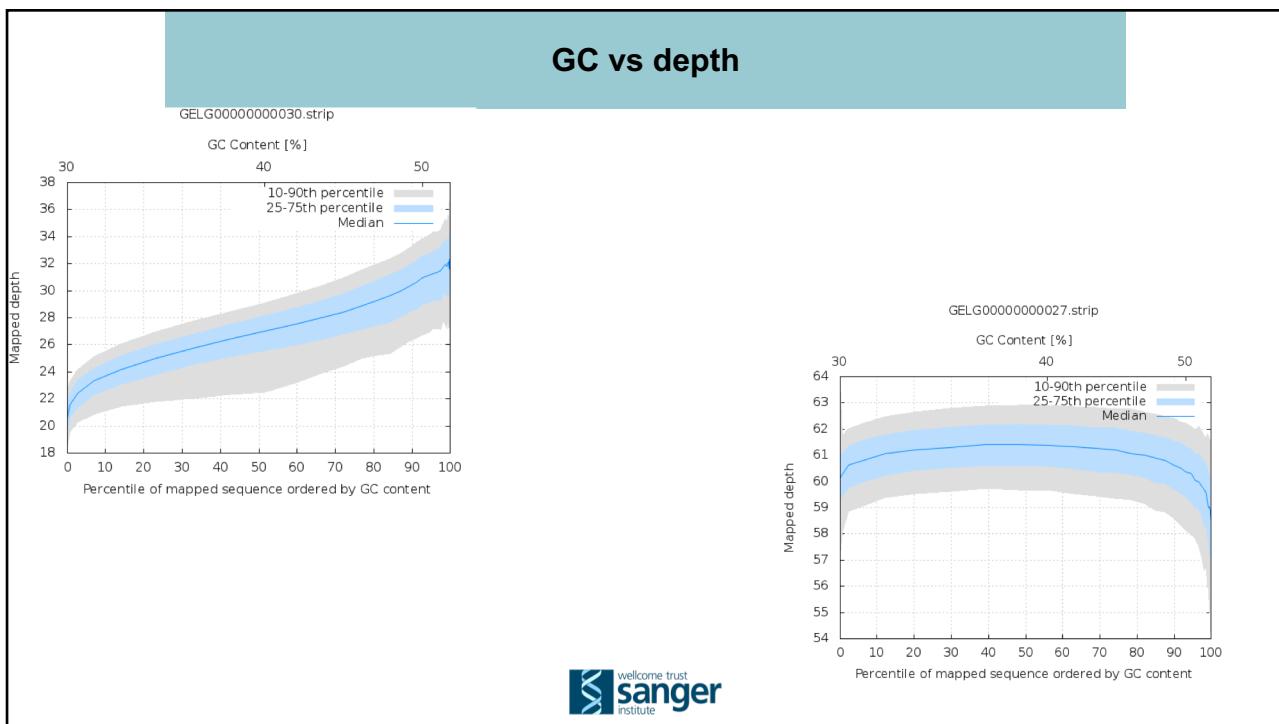
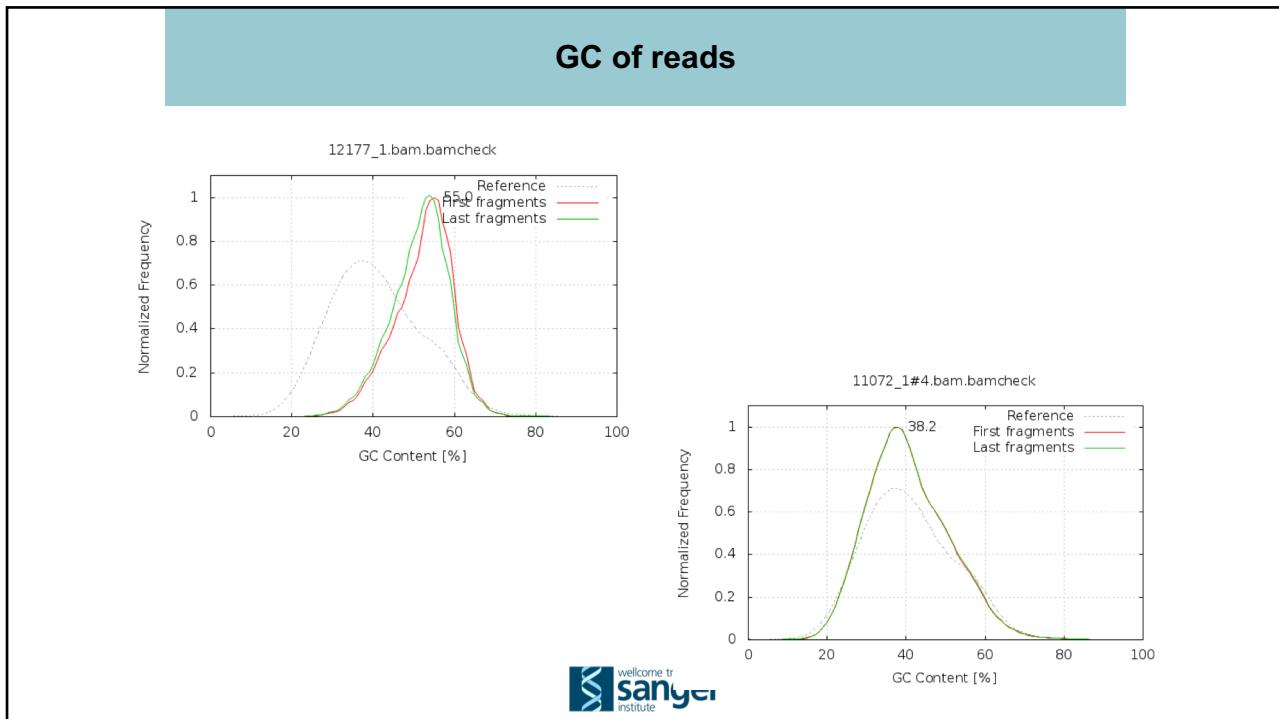
- Mapping rates – should be > 90%
- Duplicate marked rates – some want 5%, *I see anecdotally around 15%*
- GC content vs read depth – which does need mapping (why?). ~ Flat.
- Fragment size distribution. *Depends on read length and insert size.*
- Indels by cycle. *No weird spikes.*

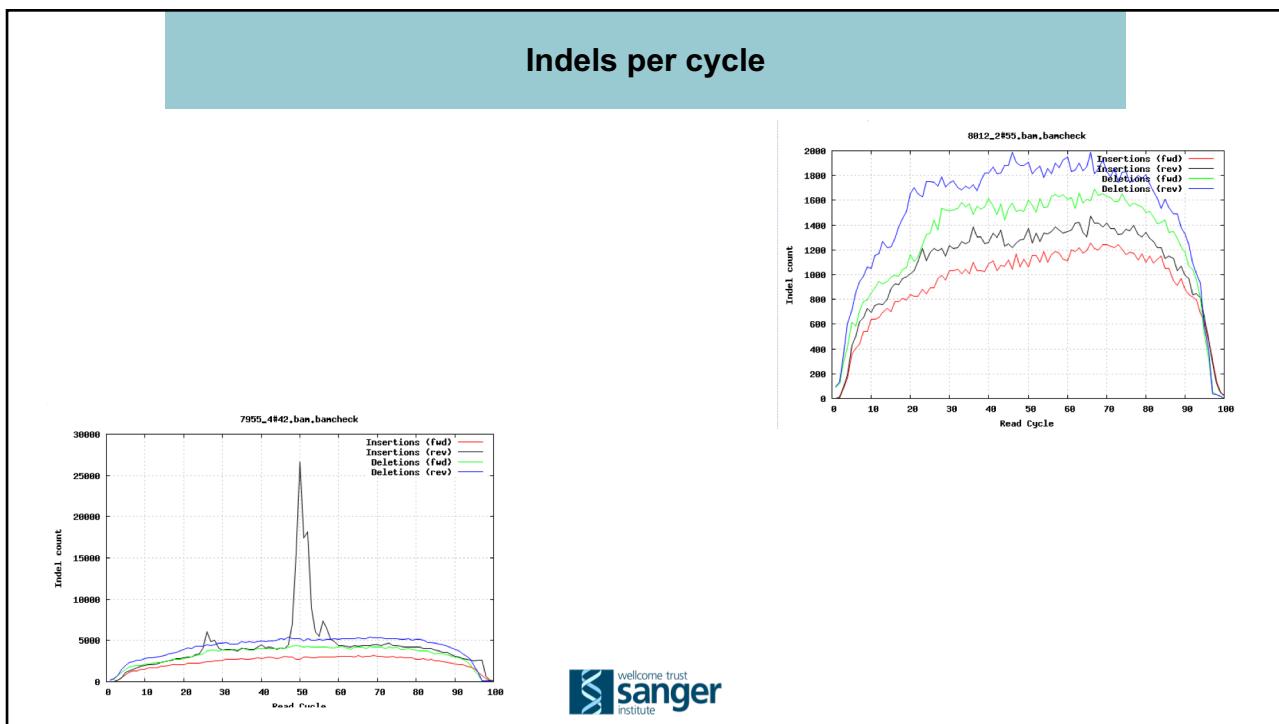
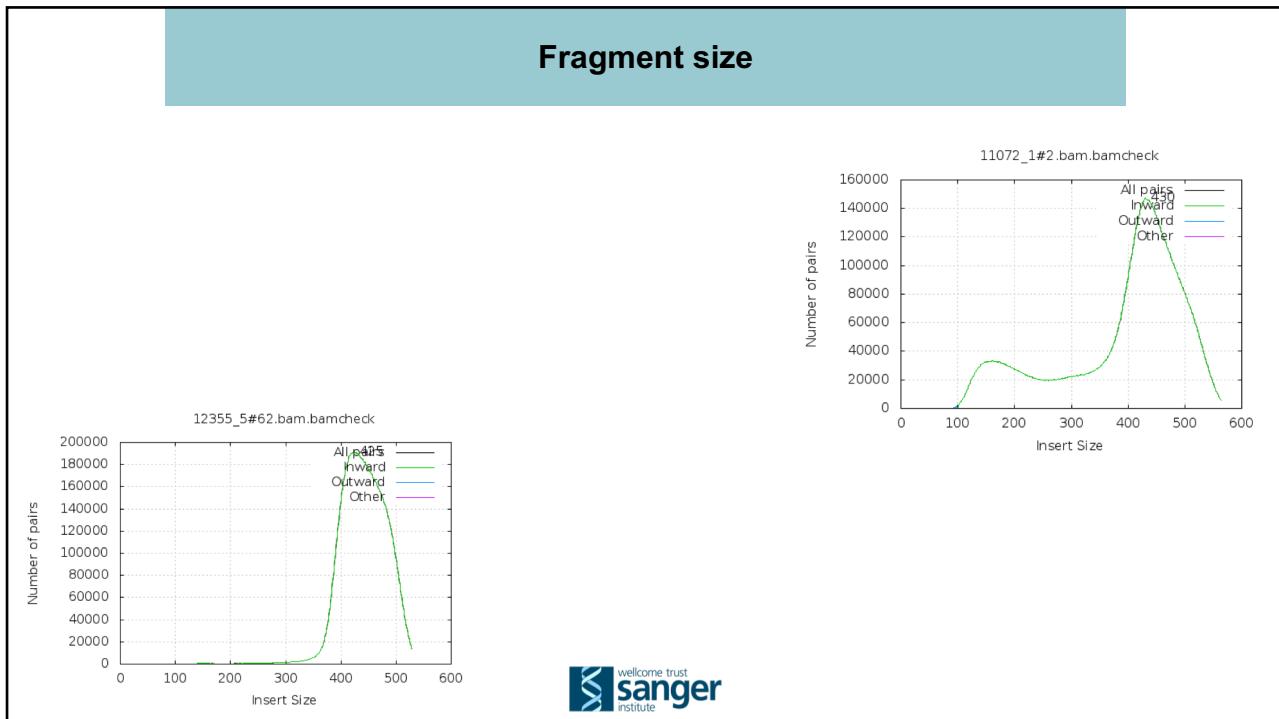


Data QC from alignments – a brief revision

And now, a quiz!







Data QC from alignments – a brief revision

QUIZ IS OVER

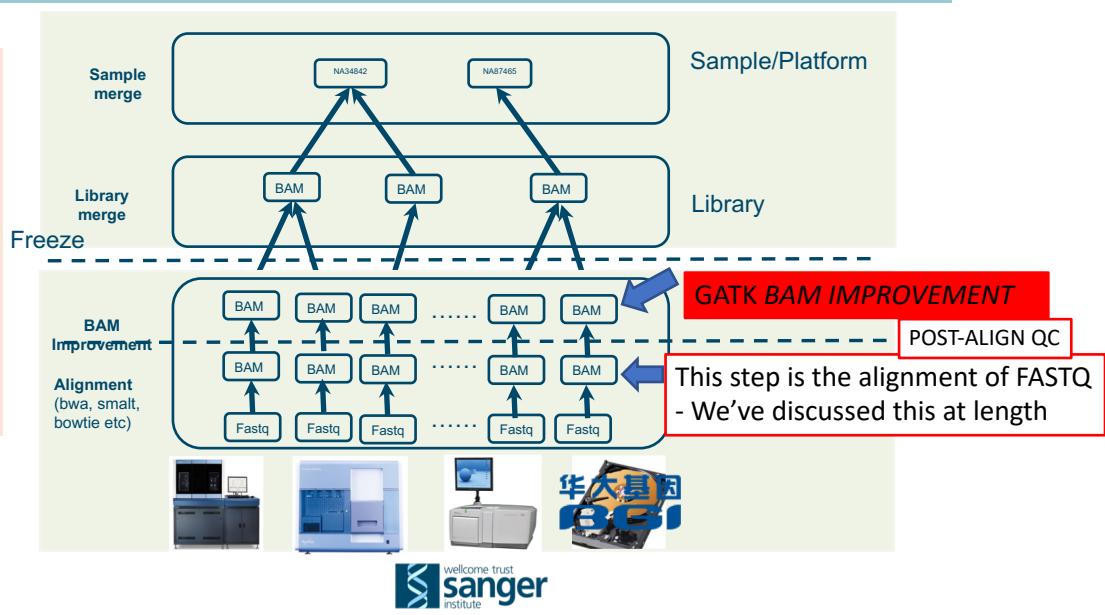


Data production workflow

Typically in a production workflow:

One sample spread over multiple seq libraries.

One library spread over multiple seq runs (lanes)



BAM improvement

FIRST - align each separate FASTQ: library x lane

Input: BAM

~~Process 1: INDEL (local) realignment~~

Process 2: Base quality recalibration

Output: BAM

BAM improvement
Part of ...
“GATK Best Practice”

Merge independent lanes BAMS in same library together

Process 3: library - level mark-duplication



BAM Improvement - Indel Realignment

THIS IS NOW DEPRECATED (even by GATK!).

I will mention it briefly because you may see it in different contexts.

PROBLEM: read aligners act on each read individually.

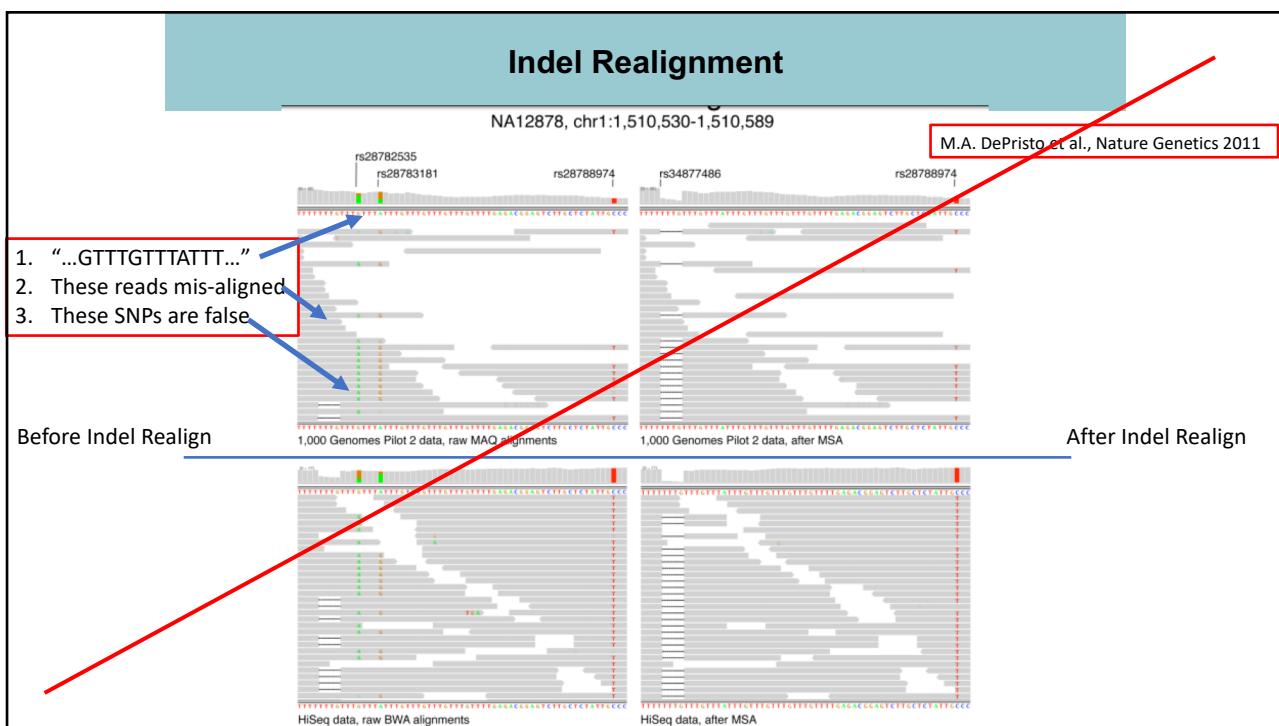
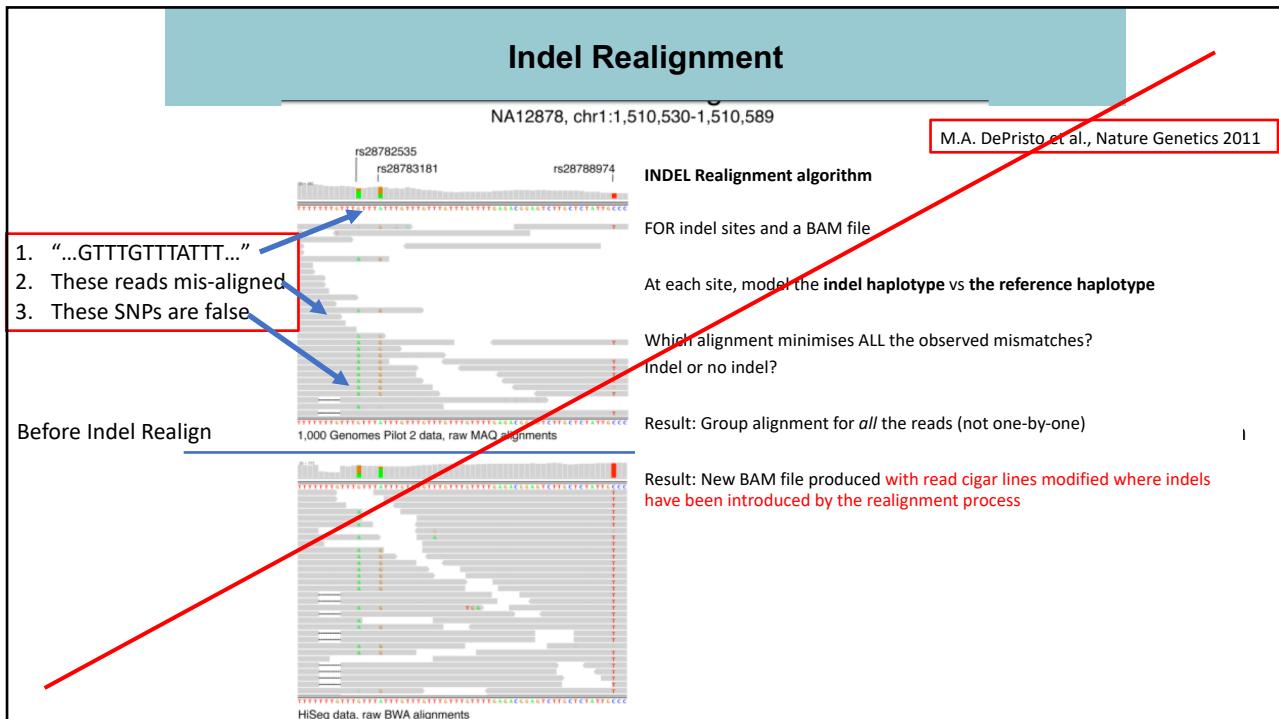
IF the read had an indel,

THEN the aligner might get it right.

OR – if the indel happened near the edge of a read – it may get alignment wrong and miss the indel.

THEN the read would have lots of MISMATCHES instead of a single indel.





BAM Improvement - Indel Realignment

THIS IS NOW DEPRECATED (even by GATK!).

Reads are getting longer and aligners are getting better – the problem is going away.

Local realignment now automatically performed by variant callers (downstream) ANYWAY.

So GATK have removed it from their "best practice" docs.



BAM Improvement - Base Quality Score Recalibration

Each base call has an associated base call quality

- What is the chance that the base call is incorrect?
 - Illumina evidence: intensity values + cycle
- Phred values (log scale)
 - Q10 = 1 in 10 chance of base call incorrect
 - Q20 = 1 in 100 chance of base call incorrect
- Accurate base qualities essential measure in variant calling

Rule of thumb: Anything less than Q20 is not useful data

BUT:

The base quality scores produced by a sequencer can be influenced by *systematic technical error*:

They can vary with sequence context, position in read etc.

Therefore the quality score can be off

Therefore variant calling can be influenced.

BQSR: adjusts the quality of each base to adjust for these systematic errors



<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3409179/>

Base quality recalibration

The idea of recalibration:

- Use the alignment of your reads to a human reference
- **Remove** all known variants dbSNP+1000G etc SNP sites
- **Assume** all other mismatches in your data **are sequencing errors**
- Now you can infer the mean observed ("empirical") error rate for these bins:
 - Position of base in read (1 => length of read)
 - Dinucleotide Sequence Context: AA, AT ..., CA, CT ...
 - Empirical quality = number of mismatches / total number in bin.
- Compare empirical rate to *reported* sequencer base quality in each of these categories.
- Derive an adjustment for each category, to be applied to each reported base quality value

e.g. Sequencer reports Q25 base calls for a "T" in context "AT"

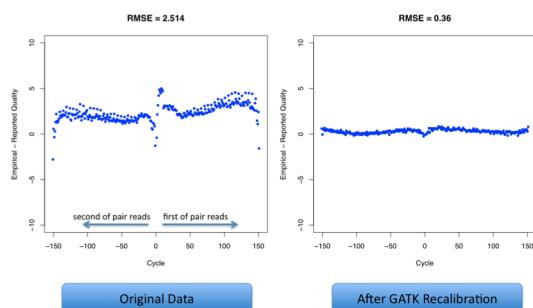
- After alignment - it may be that "T" in context "AT" actually mismatches the reference at a 1 in 100 rate, so are actually Q20
- Adjustment: for every T in "AT" – subtract 5 from BQ. (30=>25, 21=> 16 etc)

NOTE: requires a reference genome and a catalog of variable sites



Base quality recalibration effects

Residual Error by Machine Cycle



Residual Error by Dinucleotide

