



The Elizabeth H.  
and James S. McDonnell III

**McDONNELL  
GENOME INSTITUTE**  
at Washington University



**Washington**  
University in St. Louis  

---

SCHOOL OF MEDICINE

## PMBIO Module 03

### Align. Alignment algorithms, visualization, and QC

Malachi Griffith, Obi Griffith, Zachary Skidmore, Huiming Xia  
Introduction to bioinformatics for DNA and RNA sequence  
analysis (IBDR01)

29 October - 2 November, 2018  
Glasgow



## Attribution-ShareAlike 4.0 International (CC BY-SA 4.0)

This is a human-readable summary of (and not a substitute for) the [license](#). [Disclaimer](#).

### You are free to:

**Share** — copy and redistribute the material in any medium or format

**Adapt** — remix, transform, and build upon the material for any purpose, even commercially.



The licensor cannot revoke these freedoms as long as you follow the license terms.

### Under the following terms:



**Attribution** — You must give [appropriate credit](#), provide a link to the license, and [indicate if changes were made](#). You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.



**ShareAlike** — If you remix, transform, or build upon the material, you must distribute your contributions under the [same license](#) as the original.

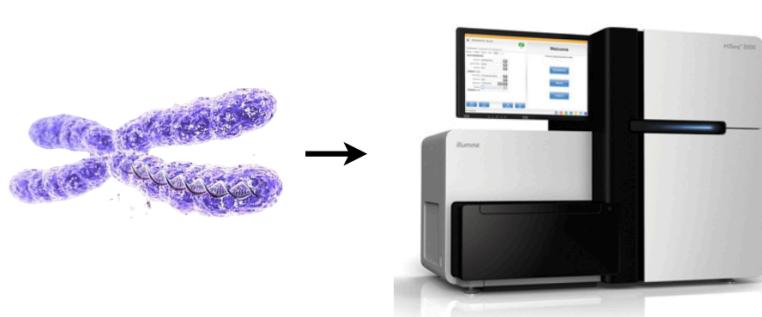
**No additional restrictions** — You may not apply legal terms or [technological measures](#) that legally restrict others from doing anything the license permits.

# Learning objectives of module 03: Align

---

- Key concepts: Sequence alignment algorithms, BAM files, genome viewers, alignment quality assessment
- Compare and contrast DNA vs. RNA sequence alignment strategies
- Perform alignment of sequence data using a few popular alignment algorithms
- Explore the BAM file format and learn approaches for summarizing, filtering, and otherwise manipulating BAM files
- Learn to use a genome viewer
- Perform a quality assessment using the aligned data

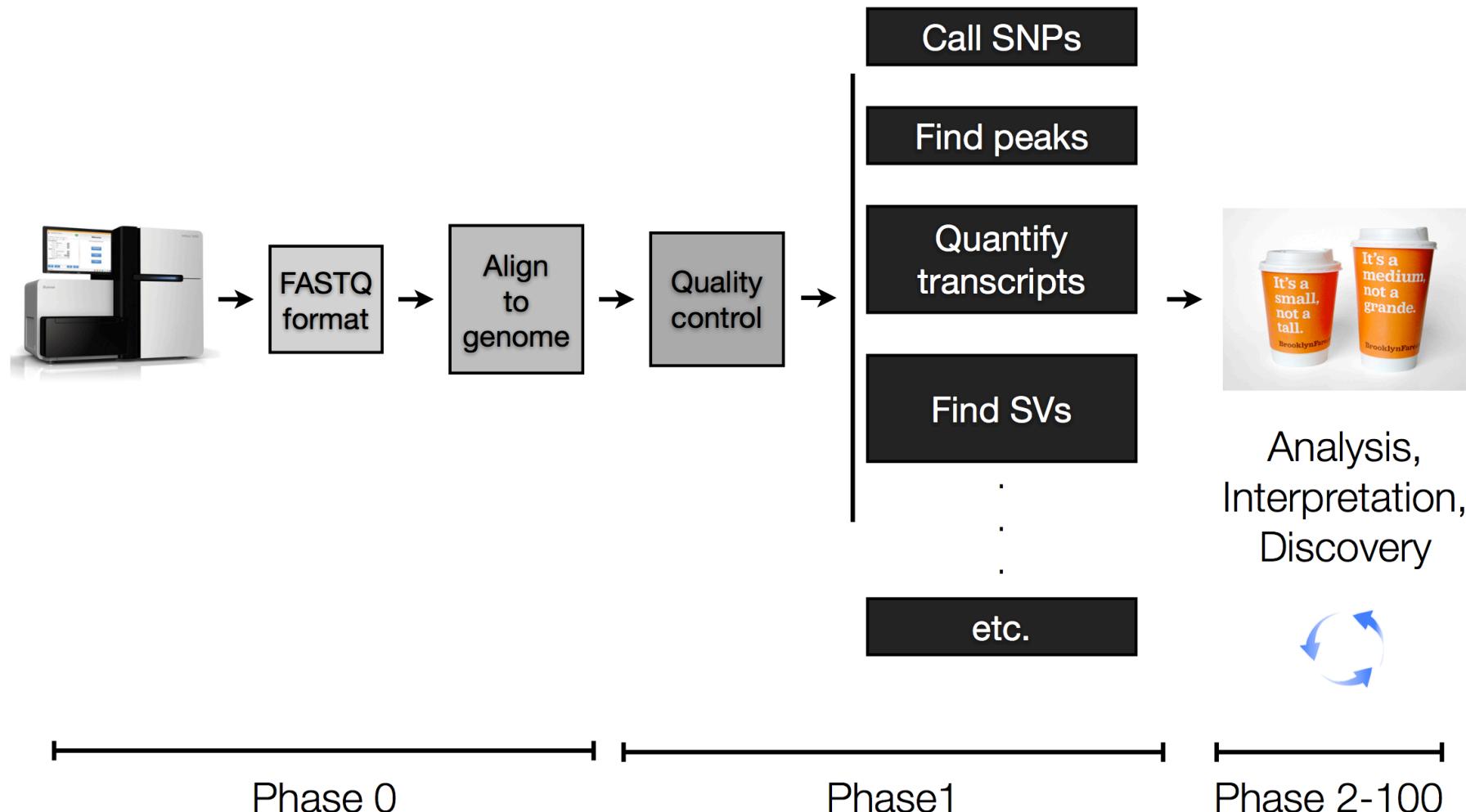
Reads are the sequencer's best guess at what it saw for a given DNA molecule. *It's the “raw” data.*



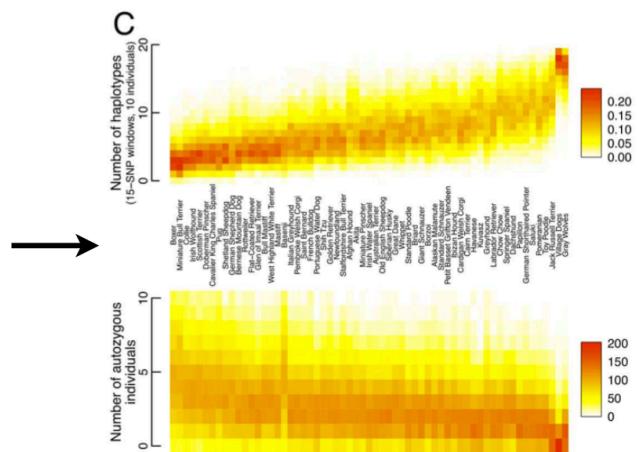
“base calling”  
(mistakes happen)

ccttaaccct  
acccctgg  
acccctgg  
ggccctgg ccttaaccct tccccctgg  
actaaccct acccctgg acccctgg  
acccctgg ccttaaccct ccttaaccct  
ctttacccct ccttaaccct acccctgg  
ccttaaccct

# Alignment is central to most genomics applications

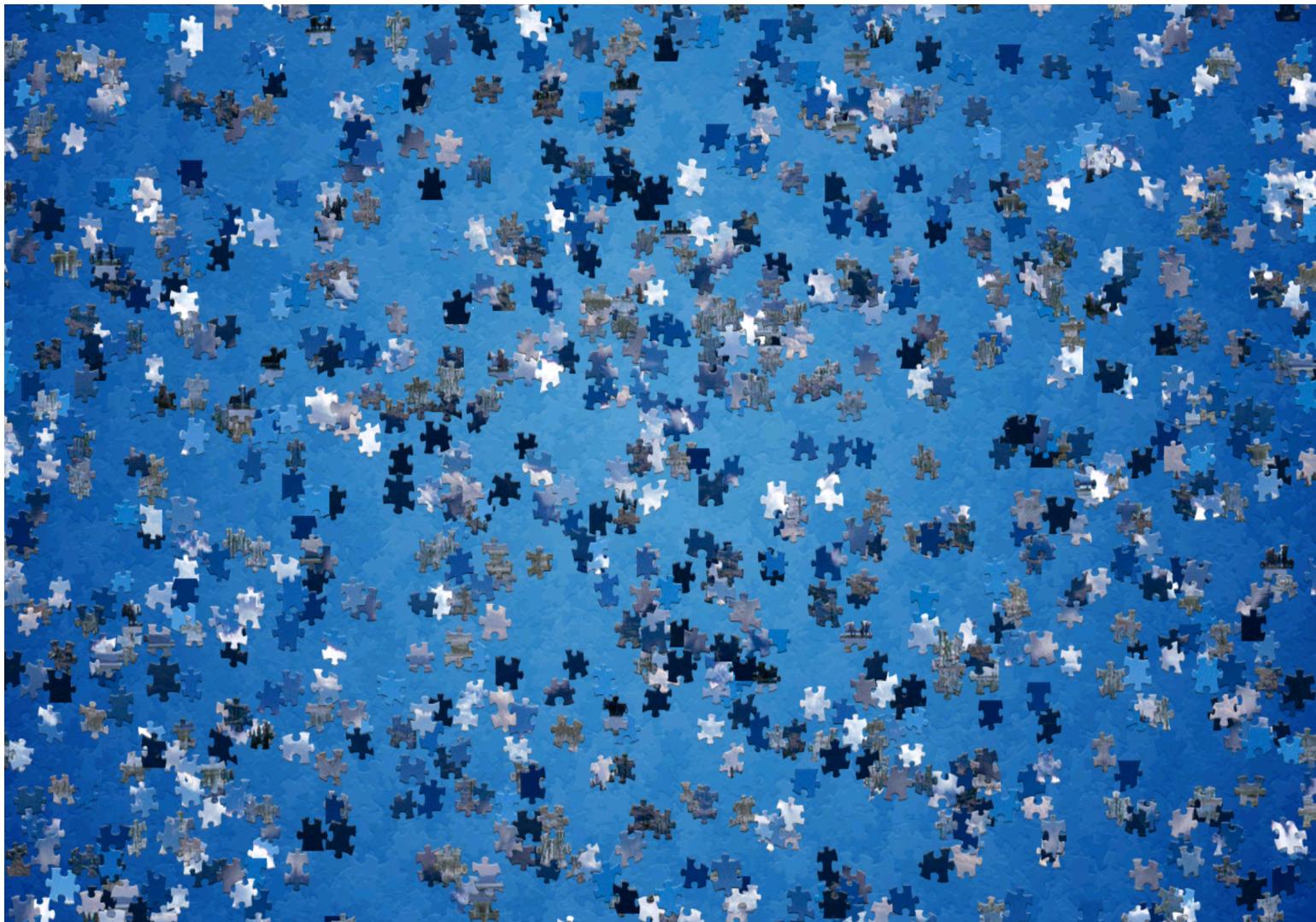


# The goal ...



Slide courtesy of Andrew Farrell (University of Utah)

# Mapping to reference genome is like solving a jigsaw puzzle



# Some parts are easier than others...



# Best case scenario

An error-free sequencing technology

ATTCGAAACA  
TTCGCGCAAT  
CTGGACTCAA



→ ATTCGAAACA → TTCGCGCAAT → Aligner →  
CTGGACTCAA

TACCTCCAGGGGGCATCCTCCC  
CCCCA**ATTCGAAACA**CAATCGTA  
GCCCTGGCACTACCTATGTGTG  
TCAATTGGAGAGAGAGAGAGATT  
ACGAAAAAAAAGT**CTGGACTCAA**  
CTAGGATACACACATTGGCTACA  
GATACCAAAAAAAAAAAAAAAA  
ATTTCAACCATTGAGGCACCACCT  
TCTCGTCGCTGCGTCGCTCTGCT  
CGCTTCGGCTAAAAA**ATTCGCGCA**  
**ATACATTGGCTACAGATACCAAA**  
AAAA

Computers are rather good at finding **exact** matches.

# Reality ...

Errors happen - frequently; work harder.

ATTCGAAACA



→ ATTTGAAACA → Aligner

TACCTCCAGGGGGCATCCTCCC  
CCCCAATTGAAACACAATCGTA  
GCCCTGGCACTACCTATGTGTG  
TCAATTGGAGAGAGAGAGAGATTG  
GAAACAAAAAAGTGCTACAGATA  
CCACTAGGATACACACATTGGC  
TACAGATACCAAAAAAAAAAAAAAA  
AAAAATTTCAACCATTGAGGCACC  
ACCTTCTCGTCGCTGCGTCGCTC  
TGCTCGCGCTAAAAAATTAGAAA  
CAACATTGGCTACAGATACCAAA  
ATT

“Fuzzy” matching is much more computationally expensive.

# Read alignment algorithms attempt to solve this problem

---

- There are optimal solutions
  - Smith-Waterman, Needleman-Wunsch
  - Computationally expensive (i.e. slow)
- Faster solutions that make some compromises
  - Hash based solutions
  - Burrows-Wheeler transform
    - Used in this course
- Extensive algorithmic details are outside of the scope of this course (refer to pmbio.org for citations and further info)
- Bottomline: aligners take raw read, determine alignment to reference genome and output a **SAM/BAM file**

# SAM/BAM/CRAM files represent sequence alignments

- The specification
  - <http://samtools.sourceforge.net/SAM1.pdf>
- The SAM format consists of two sections:
  - Header section
    - Used to describe source of data, reference sequence, method of alignment, etc.
  - Alignment section
    - Used to describe the read, quality of the read, and nature alignment of the read to a region of the genome
- BAM/CRAM are compressed versions of SAM.
  - BAM compressed using lossless BGZF format
  - CRAM compressed further using knowledge of reference. May or may not be lossless
- BAM/CRAM files are usually ‘indexed’
  - A ‘.bai’ file will be found beside the ‘.bam’ file
  - Indexing aims to achieve fast retrieval of alignments

## A BAM file is divided in header and alignment sections

## Example SAM/BAM header section (abbreviated)

```
mgriffit@linus270 ~> samtools view -H /gscmnt/gc13001/info/model_data/2891632684/build136494552/alignments/136080019.bam | grep -P "SN:22|HD|RG|PG"
@HD VN:1.4 SO:coordinate
@SQ SN:22 LN:51304566 UR:ftp://ftp.ncbi.nih.gov/genbank/genomes/Eukaryotes/vertebrates_mammals/Homo_sapiens/GRCh37/special_requests/GRCh37-lite.fa.gz AS:GRCh37-lite M5:a718aca6135fdca8357d5bfe9
2411dd SP:Homo sapiens
@RG ID:2888721359 PL:illumina PU:D1BA4ACXX.3 LB:H_KA-452198-0817007-cDNA-3-lib1 PI:365 DS:paired end DT:2012-10-03T19:00:00-0500 SM:H_KA-452198-0817007 CN:WUGSC
@PG ID:2888721359 VN:2.0.8 CL:tophat --library-type fr-secondstrand --bowtie-version=2.1.0
@PG ID:MarkDuplicates PN:MarkDuplicates PP:2888721359 VN:1.85(exported) CL:net.sf.picard.sam.MarkDuplicates INPUT=/gscmnt/gc13001/info/build_merged_alignments/merged-alignment-blade10-2-5.gsc.wustl.edu-jwalker-15434-136080019/scratch-Ilg6Y/H_KA-452198-0817007-cDNA-3-lib1-2888360300.bam OUTPUT=/gscmnt/gc13001/info/build_merged_alignments/merged-alignment-blade10-2-5.gsc.wustl.edu-jwalker-15434-136080019/scratch-Ilg6Y/H_KA-452198-0817007-cDNA-3-lib1-2888360300.post_dup.bam METRICS_FILE=/gscmnt/gc13001/info/build_merged_alignments/merged-alignment-blade10-2-5.gsc.wustl.edu-jwalker-15434-136080019/staging-1iuJS/H_KA-452198-0817007-cDNA-3-lib1-2888360300.metrics REMOVE_DUPLICATES=false ASSUME_SORTED=true MAX_FILE_HANDLES_FOR_READ_ENDS_MAP=9500 TMP_DIR=/gscmnt/gc13001/info/build_merged_alignments/merged-alignment-blade10-2-5.gsc.wustl.edu-jwalker-15434-136080019/scratch-Ilg6Y VALIDATION_STRINGENCY=SILENT MAX_RECORDS_IN_RAM=500000 PROGRAM_RECORD_ID=MarkDuplicates PROGRAM_GROUP_NAME=MarkDuplicates_MAX_SEQUENCES_FOR_DISK_READ_ENDS_MAP=50000 SORTING_COLLECTION_SIZE_RATIO=0.25 READ_NAME_REGEX=[a-zA-Z0-9]+:[0-9]+([0-9]+):([0-9]+).*, OPTICAL_DUPLICATE_PIXEL_DISTANCE=100 VERBOSITY=INFO QUIET=false COMPRESSION_LEVEL=5 CREATE_INDEX=false CREATE_MD5_FILE=false
mgriffit@linus270 ~>
```

## Example SAM/BAM alignment section (only 10 alignments shown)

## BAM header section provides general information about alignment strategy

---

- Used to describe source of data, reference sequence, method of alignment, etc.
- Each section begins with character ‘@’ followed by a two-letter record type code. These are followed by two-letter tags and values
  - @HD The header line
    - VN: format version
    - SO: Sorting order of alignments
  - @SQ Reference sequence dictionary
    - SN: reference sequence name
    - LN: reference sequence length
    - SP: species
  - @RG Read group
    - ID: read group identifier
    - CN: name of sequencing center
    - SM: sample name
  - @PG Program
    - PN: program name
    - VN: program version

# BAM alignment section provides details for each read alignment

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

1 QNAME e.g. HWI-ST495\_129147882:1:2302:10269:12362 (QNAME)  
2 FLAG e.g. 99  
3 RNAME e.g. 1  
4 POS e.g. 11623  
5 MAPQ e.g. 3  
6 CIGAR e.g. 100M  
7 RNEXT e.g. =  
8 PNEXT e.g. 11740  
9 TLEN e.g. 217  
10 SEQ e.g. CCTGTTCTCCACAAAGTGTACTTTGGATTTGCCAGTCTAACAGGTGAAGCCCTGGAGATTCTTATTAGTGATTGGGCTGGGCATGT  
11 QUAL e.g. CCCFFFFFHJHJIJFIJJJJJJJJJHIIJJJJJJJJGGHIIJHIIJJJJJJJJGHGGIJJJJJJJEEHHHFFFFCDCDDDDDDDB@ACDD

# BAM flags describe several alignment properties in a single number

- <http://broadinstitute.github.io/picard/explain-flags.html>
- 12 bitwise flags describing the alignment
- These flags are stored as a binary string of length 11 instead of 11 columns of data
- Value of '1' indicates the flag is set. e.g. 00100000000
- All combinations can be represented as a number from 1 to 2048 (i.e.  $2^{11}-1$ ). This number is used in the BAM/SAM file. You can specify 'required' or 'filter' flags in samtools view using the '-f' and '-F' options respectively

Bit	Description
1	0x1 template having multiple segments in sequencing
2	0x2 each segment properly aligned according to the aligner
4	0x4 segment unmapped
8	0x8 next segment in the template unmapped
16	0x10 SEQ being reverse complemented
32	0x20 SEQ of the next segment in the template being reverse complemented
64	0x40 the first segment in the template
128	0x80 the last segment in the template
256	0x100 secondary alignment
512	0x200 not passing filters, such as platform/vendor quality controls
1024	0x400 PCR or optical duplicate
2048	0x800 supplementary alignment

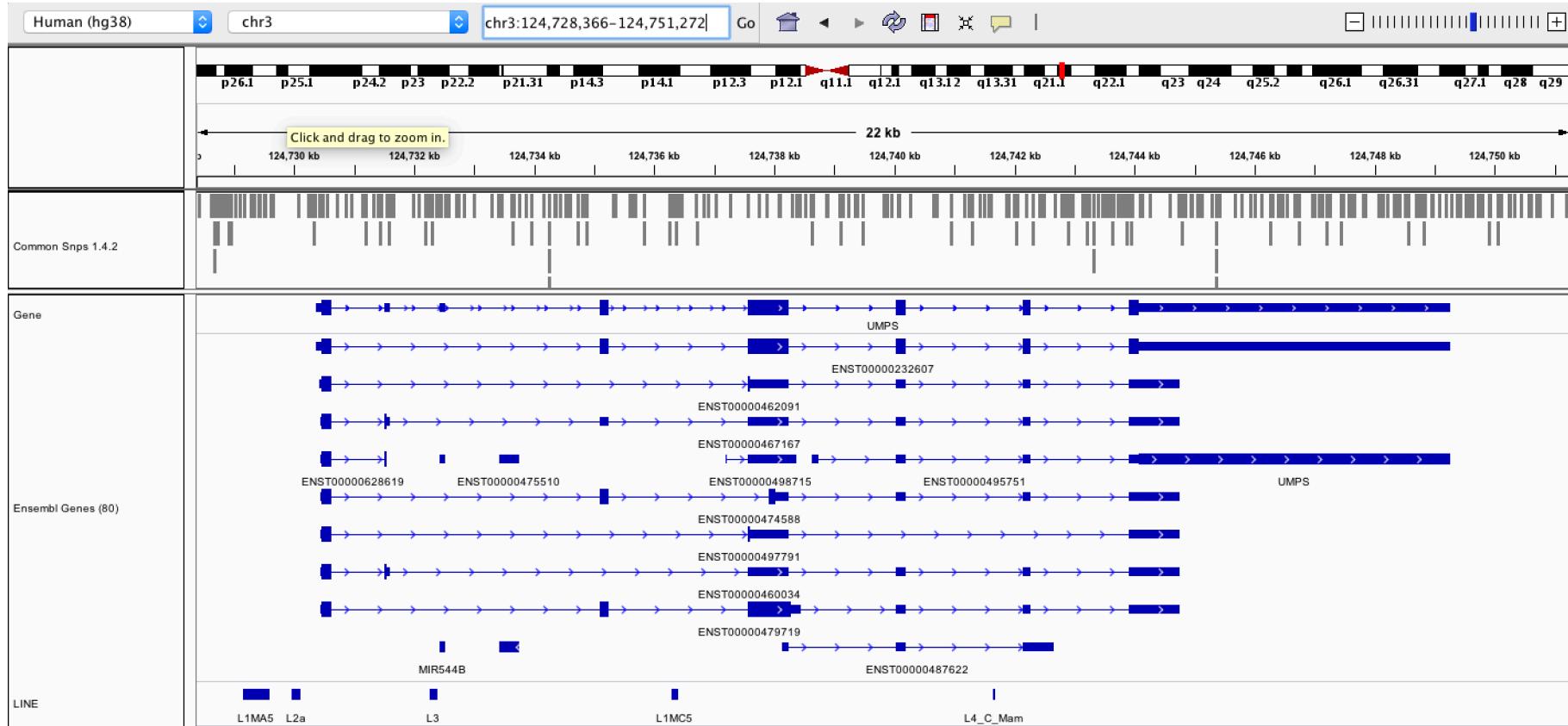
Note that to maximize confusion, each bit is described in the SAM specification using its hexadecimal representation (i.e., '0x10' = 16 and '0x40' = 64).

## CIGAR strings similarly describe the entire alignment in as few characters as possible

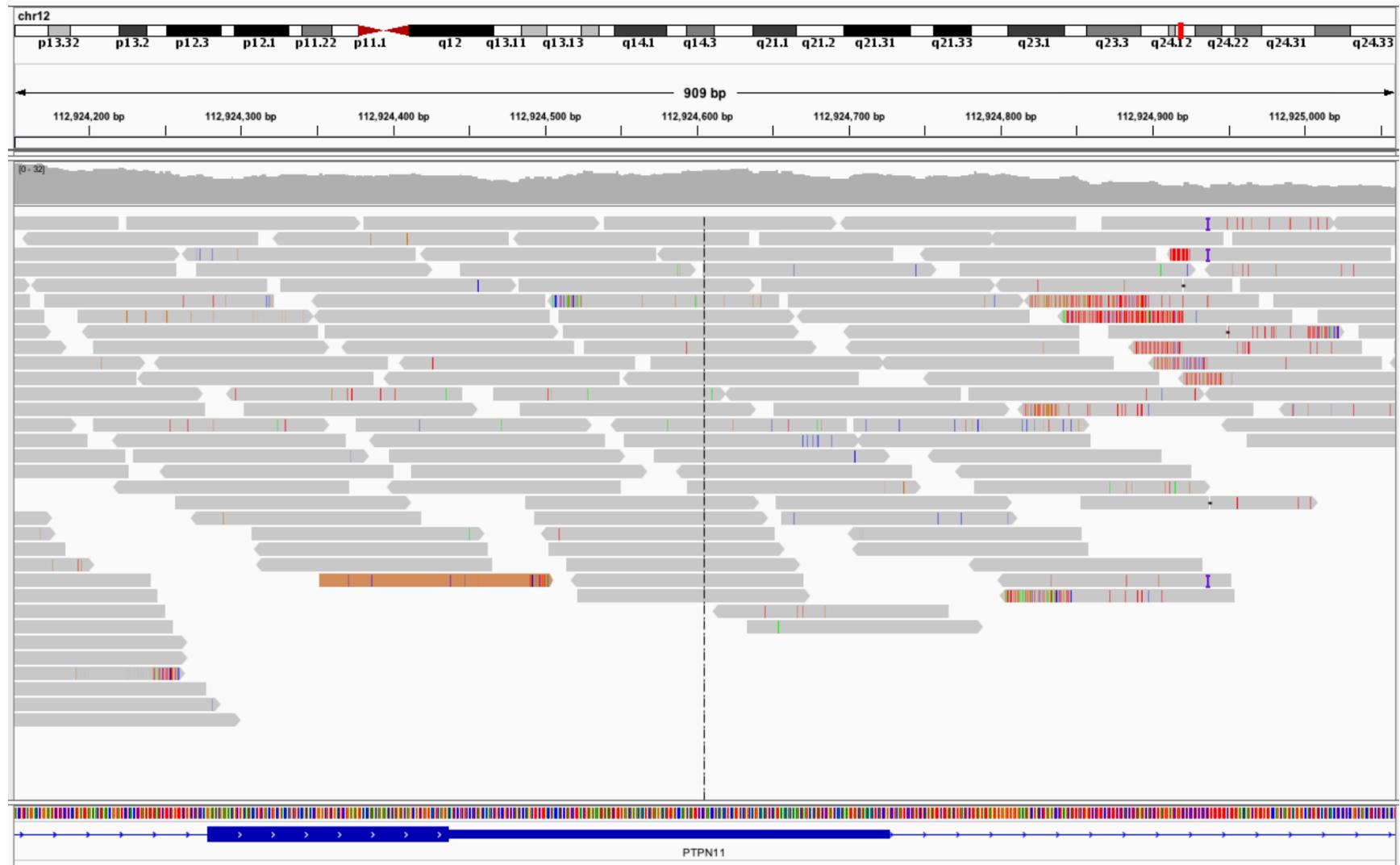
Op	BAM	Description
M	0	alignment match (can be a sequence match or mismatch)
I	1	insertion to the reference
D	2	deletion from the reference
N	3	skipped region from the reference
S	4	soft clipping (clipped sequences present in SEQ)
H	5	hard clipping (clipped sequences NOT present in SEQ)
P	6	padding (silent deletion from padded reference)
=	7	sequence match
X	8	sequence mismatch

- The CIGAR string is a sequence of base lengths and associated ‘operations’ that are used to indicate which bases align to the reference (either a match or mismatch), are deleted, are inserted, represent introns, etc.
- e.g. 81M859N19M
  - A 100 bp read consists of: 81 bases of alignment to reference, 859 bases skipped (an intron), 19 bases of alignment

# Genome browsers - IGV



# Genome browsers - IGV



# Alignment QC ...



A modular tool to aggregate results from bioinformatics analyses across many samples into a single report.

Report generated on 2018-10-29, 21:51 based on data in: /workspace/align

Welcome! Not sure where to start?

[Watch a tutorial video](#) (6:06)

don't show again

## General Statistics

Sample Name	M Reads Mapped	% Aligned	Fold Enrichment	Target Bases 30X	Insert Size	Median Coverage	Bases ≥ 30X	% Dups	% GC	M Seqs
Exome_Norm_flagstat	12.7									
Exome_Norm_sorted_mrkdup_bqsr		99%	23	83%	243 bp			16.2%	47%	12.7
Exome_Tumor_flagstat	16.6									
Exome_Tumor_sorted_mrkdup_bqsr		99%	23	87%	240 bp			26.5%	48%	16.7
WGS_Norm_merged_flagstat	77.9									
WGS_Norm_merged_sorted_mrkdup_bqsr		99%			347 bp	29.0X	47%	7.8%	40%	78.5
WGS_Tumor_merged_flagstat	149.8									
WGS_Tumor_merged_sorted_mrkdup_bqsr		99%			310 bp	51.0X	83%	14.8%	40%	150.5