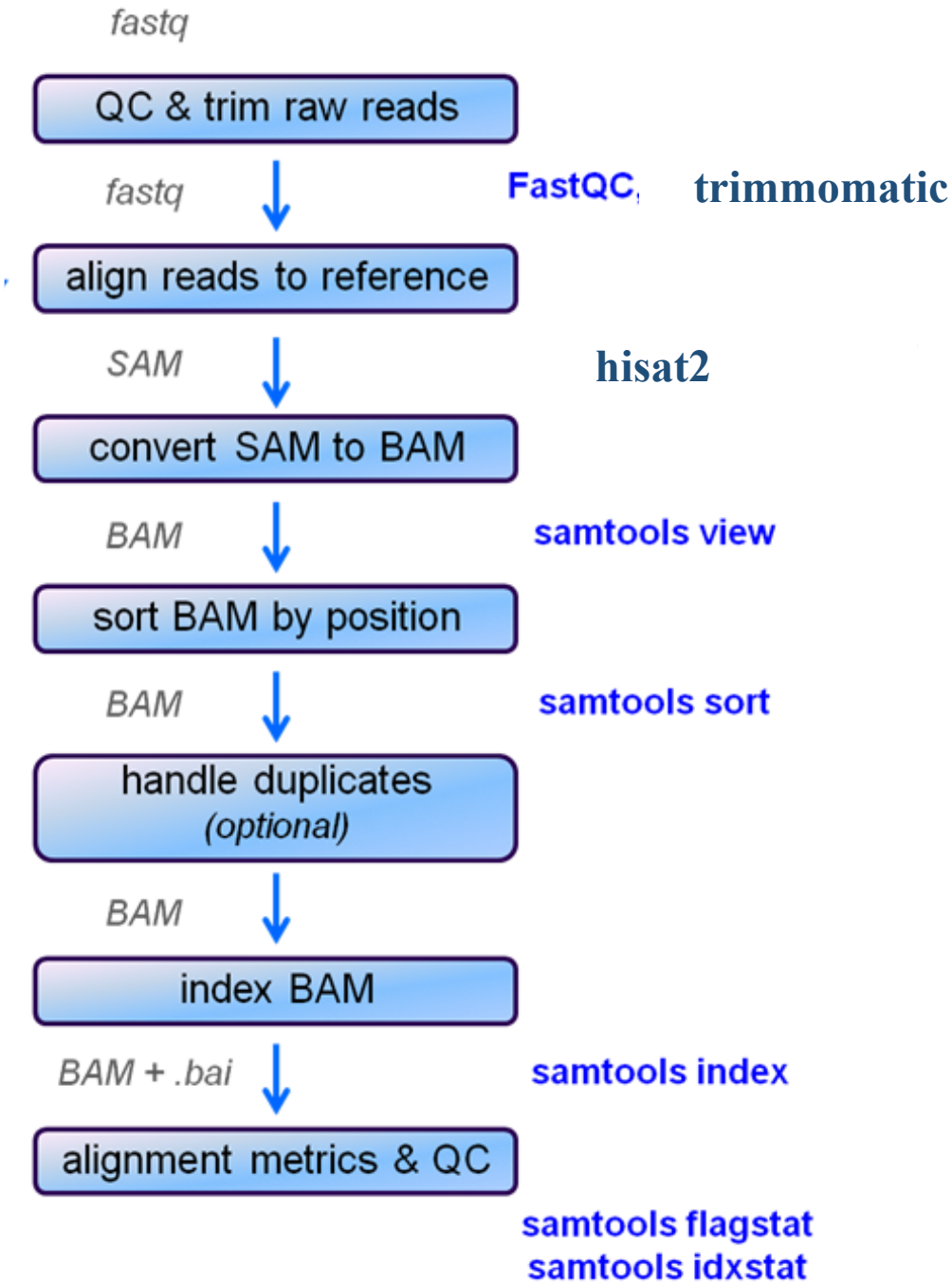
# Learning the SAM/BAM format


# Alignment Workflow



# Part 1: Provide the job submission parameters

Our script will be written in sections:

1. Will provide the job submission parameters

```
#!/bin/bash
#SBATCH --partition=bluemoon
#SBATCH --nodes=1
#SBATCH --ntasks=2
#SBATCH --mem=40G
#SBATCH --time=24:00:00
#SBATCH --job-name=align_CD8
# %x=job-name %j=jobid
#SBATCH --output=%x_%j.out
```

*https://prodriguez19.github.io/Intro-to-rnaseq/lessons/05_Mapping_with_HISAT2.html*

# Part 2: To keep the naming convention for each file output by this script

```bash
for i in reads/*.fastq
do
  SAMPLE=$(echo ${i} | sed "s/.fastq//")
  echo ${SAMPLE}.fastq
```

*Beginning of for loop*

# Part 3: Load the modules required to run the commands

```
```

module load hisat2-2.1.0-gcc-7.3.0-knvgwpc
module load samtools-1.10-gcc-7.3.0-pdbkohx
```
```

# Part 4: The *actual* commands to be executed

```
#align to GRCm39
hisat2 \
  -p ${p} \
  -x ${DBDIR}/${GENOME} \
  -U ${SAMPLE}.fastq.gz \
  -S ${SAMPLE}.sam &> ${SAMPLE}.log

#create bam file
samtools view ${SAMPLE}.sam \
  --threads 2 \
  -b \
  -o ${SAMPLE}.bam \

#remove sam files once bam file is created
rm ${SAMPLE}.sam

#output stats
samtools flagstat ${SAMPLE}.bam > ${SAMPLE}.txt

# sort the bam file based on coordinates
samtools sort ${SAMPLE}.bam -o ${SAMPLE}_sorted.bam

# index bam file
samtools index ${SAMPLE}_sorted.bam

done &> hisat2.log
```

# SAMtools usage

- http://www.htslib.org/doc/samtools.html

samtools view [*options*] *in.sam*|*in.bam*|*in.cram* [*region*...]

samtools sort [-l *level*] [-u] [-m *maxMem*] [-o *out.bam*] [-O *format*] [-M] [-K *kmerLen*] [-n] [-t *tag*] [-T *tmpprefix*] [-@ *threads*] [*in.sam*|*in.bam*|*in.cram*]

samtools index [-bc] [-m *INT*] *aln.sam*|*aln.bam*|*aln.cram* [*out.index*]

# Read alignments files: the SAM format

- ‘Sequence Alignment/Map’ format - http://samtools.sourceforge.net/SAMv1.pdf
- SAM/BAM files an be manipulated with SAMtools (and others)
- SAM files are tab-delimited files, human-readable
- The SAM file contains two sections:

1. Header section:
   - Metadata about the genome, the samples, the pipeline
   - Header lines start with @
2. Alignments (or ‘records’) section:

# To view a SAM file:

```
module load samtools-1.10-gcc-7.3.0-pdbkohx

samtools view -h SRR13423162_sorted.bam | less -S
```

# SAM header descriptions

| Tag | Description |
|---|---|
| @HD | The header line. The first line if present. |
| VN* | Format version. *Accepted format*: /^[0-9]+\.[0-9]+$/. |
| SO | Sorting order of alignments. *Valid values*: **unknown** (default), **unsorted**, **queryname** and **coordinate**. For coordinate sort, the major sort key is the RNAME field, with order defined by the order of @SQ lines in the header. The minor sort key is the POS field. For alignments with equal RNAME and POS, order is arbitrary. All alignments with '*' in RNAME field follow alignments with some other value but otherwise are in arbitrary order. |
| GO | Grouping of alignments, indicating that similar alignment records are grouped together but the file is not necessarily sorted overall. *Valid values*: **none** (default), **query** (alignments are grouped by QNAME), and **reference** (alignments are grouped by RNAME/POS). |
| @SQ | Reference sequence dictionary. The order of @SQ lines defines the alignment sorting order. |
| SN* | Reference sequence name. The SN tags and all individual AN names in all @SQ lines must be distinct. The value of this field is used in the alignment records in RNAME and RNEXT fields. Regular expression: [!-)+-<>-~][!-~]* |
| LN* | Reference sequence length. *Range*: $[1,2^{31}-1]$ |
| AH | Indicates that this sequence is an alternate locus.[4] The value is the locus in the primary assembly for which this sequence is an alternative, in the format '*chr*:*start*-*end*', '*chr*' (if known), or '*' (if unknown), where '*chr*' is a sequence in the primary assembly. Must not be present on sequences in the primary assembly. |
| AN | Alternative reference sequence names. A comma-separated list of alternative names that tools may use when referring to this reference sequence.[5] These alternative names are not used elsewhere within the SAM file; in particular, they must not appear in alignment records' RNAME or RNEXT fields. *Regular expression*: *name*(,*name*)* where *name* is [0-9A-Za-z][0-9A-Za-z*+.@_\|-]* |
| AS | Genome assembly identifier. |
| M5 | MD5 checksum of the sequence. See Section 1.3.1 |
| SP | Species. |
| UR | URI of the sequence. This value may start with one of the standard protocols, e.g http: or ftp:. If it does not start with one of these protocols, it is assumed to be a file-system path. |
| @RG | Read group. Unordered multiple @RG lines are allowed. |
| ID* | Read group identifier. Each @RG line must have a unique ID. The value of ID is used in the RG tags of alignment records. Must be unique among all read groups in header section. Read group IDs may be modified when merging SAM files in order to handle collisions. |
| CN | Name of sequencing center producing the read. |
| DS | Description. |
| DT | Date the run was produced (ISO8601 date or date/time). |
| FO | Flow order. The array of nucleotide bases that correspond to the nucleotides used for each flow of each read. Multi-base flows are encoded in IUPAC format, and non-nucleotide flows by various other characters. *Format*: /*\|[ACMGRSVTWYHKDBN]+/ |

# SAM alignment section

cf. FASTQ format

| Read Name | FLAG | Chrom | AlnStart | | CIGAR | | | | Sequence | BaseQuals |
|---|---|---|---|---|---|---|---|---|---|---|
| 6_1303_10584_85775 | 99 | groupVIII | 311 | 3 | 63M3I34M | = | 780 | 572 | GGGTATTGGGC | @CFFFFFHFH |
| 6_1111_20943_90813 | 163 | groupVIII | 315 | 40 | 100M | = | 809 | 594 | TAATGAAGCCAT | @BDDFDDA+<A< |
| 6_2111_2016_88235 | 355 | groupVIII | 315 | 3 | 100M | = | 856 | 573 | TAATGAAGCCAT | @?DADDBD>D>B |
| 6_1104_8139_99999 | 163 | groupVIII | 316 | 14 | 100M | = | 818 | 602 | AATGAAGCCATT | @@FFFFFGHGHH |
| 6_1304_4167_91751 | 163 | groupVIII | 322 | 5 | 52M3I29M | = | 812 | 573 | GCCATTTTTAC | <<BDBDEHHDF |
| 6_2301_14383_16382 | 163 | groupVIII | 323 | 40 | 51M3I46M | = | 809 | 589 | CCATTTTTACT | CCFFFFFHHHH |

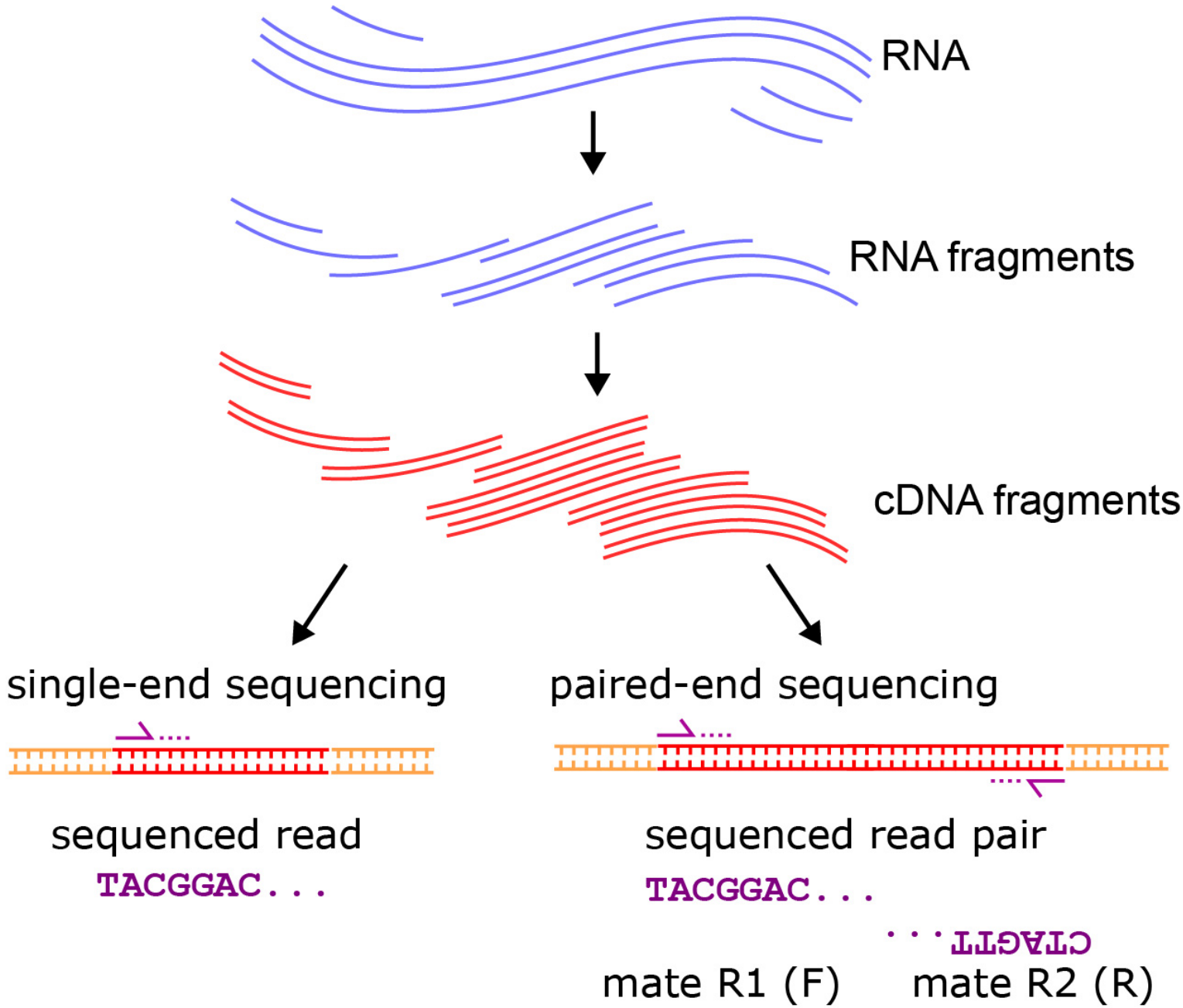
| Col | Field | Type | Regexp/Range | Brief description |
|---|---|---|---|---|
| 1 | QNAME | String | `[!-?A-~]{1,255}` | Query template NAME |
| 2 | FLAG | Int | $[0,2^{16}-1]$ | bitwise FLAG |
| 3 | RNAME | String | `\*|[!-()+-<>-~][!-~]*` | Reference sequence NAME |
| 4 | POS | Int | $[0,2^{31}-1]$ | 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 | $[0,2^{29}-1]$ | Position of the mate/next read |
| 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 |

# Column 2: Bitwise Flag

- Is a lookup code to explain certain features about the particular read
- It tells you whether the read aligned, is marked as a PCR duplicate, if its mate aligned, etc.

| 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 |

***A combination of the flags, results in one integer, which makes it difficult to interpret***

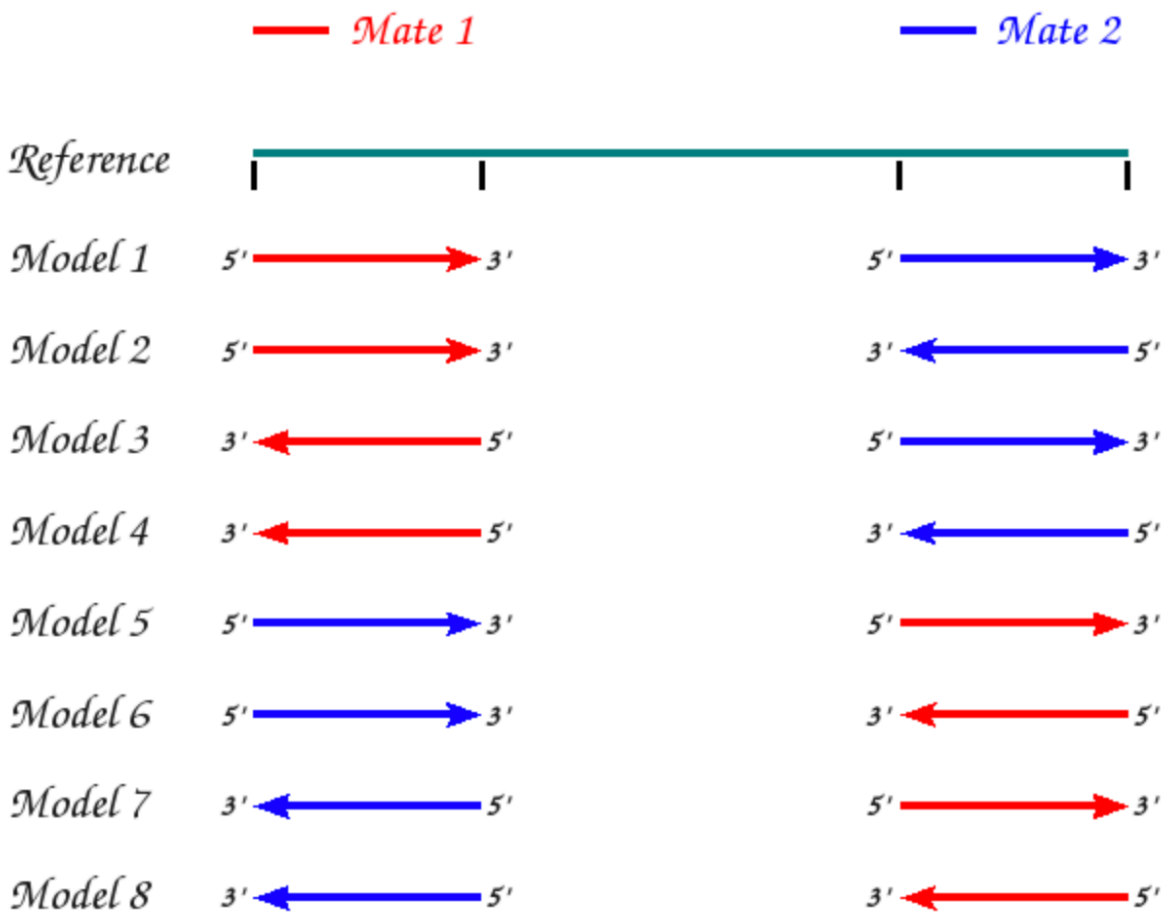

RNA
RNA fragments
cDNA fragments
single-end sequencing
paired-end sequencing
sequenced read
TACGGAC...
sequenced read pair
TACGGAC...
...CTAGTT
mate R1 (F)
mate R2 (R)

# “Proper” mate-pairing

*Same direction*

*Face-in*

*Face-out*



A combination of the flags, results in one integer, which makes it difficult to interpret

https://broadinstitute.github.io/picard/explain-flags.html

# Column 6: ‘CIGAR strings’

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 6_1303_10584_85775 | 99 | groupVIII | 311 | 3 | 63M3I34M | = | 780 | 572 | GGGTATTGGGC | @CFFFFFHFH |
| 6_1111_20943_90813 | 163 | groupVIII | 315 | 40 | 100M | = | 809 | 594 | TAATGAAGCCAT | @BDDFDDA+<A< |
| 6_2111_2016_88235 | 355 | groupVIII | 315 | 3 | 100M | = | 856 | 573 | TAATGAAGCCAT | @?DADDBD>D>B |
| 6_1104_8139_99999 | 163 | groupVIII | 316 | 14 | 100M | = | 818 | 602 | AATGAAGCCATT | @@FFFFFGHGHH |
| 6_1304_4167_91751 | 163 | groupVIII | 322 | 5 | 52M3I29M | = | 812 | 573 | GCCATTTTTAC | <<BDBDEHHDF |
| 6_2301_14383_16382 | 163 | groupVIII | 323 | 40 | 51M3I46M | = | 809 | 589 | CCATTTTTACT | CCFFFFFHHHH |

| 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 |

100M — 100 matching nucleotides (i.e. no gaps)

63M-3I-34M — 63 matching nucleotides
3 nucleotides not in the reference (3bp insertion)
34 matching nucleotides

**Aligned Read**

TGCAGGATGGATGTGTTCCTCCTCAGCTGCTTATTTTAACTCCACTGCACAACATGTTTTGTGTTATATTCTTTCGCTGTGTAGTCTGTAAGC

TGCAGGGACTGCAGGATGGATGTGTTCCTCCTCAGCTGCTTATTTTAACTCCAC---ACAACATGTTTTGTGTTATATTCTTTCGCTGTGTAGTCTGTAAGCAGAGTATGATACTG

**Reference**

# Column 6: ‘CIGAR strings’

| 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 |

- Example: intron = 81 bases

ERR022486.8388510 81 22 32099 255 **58M81N18M** = 27484 -4772
CCTTGGTCTTGCCGAAGTAGATCTCATTGAGAGTGGAGCGGATCTTGTTCTCCATTTCCTCCA
CCAGGCGTCCGAT :9=<==;<<><=><?>>?<?==>>?>><?>>??<AA?
@AFADDD;GDGAG@GGCBE@GG?GG>GGGG?GGGGGGGG NM:i:0 XS:A:- NH:i:1

***Spliced***

**Aligned Read** 58M 81N 18M

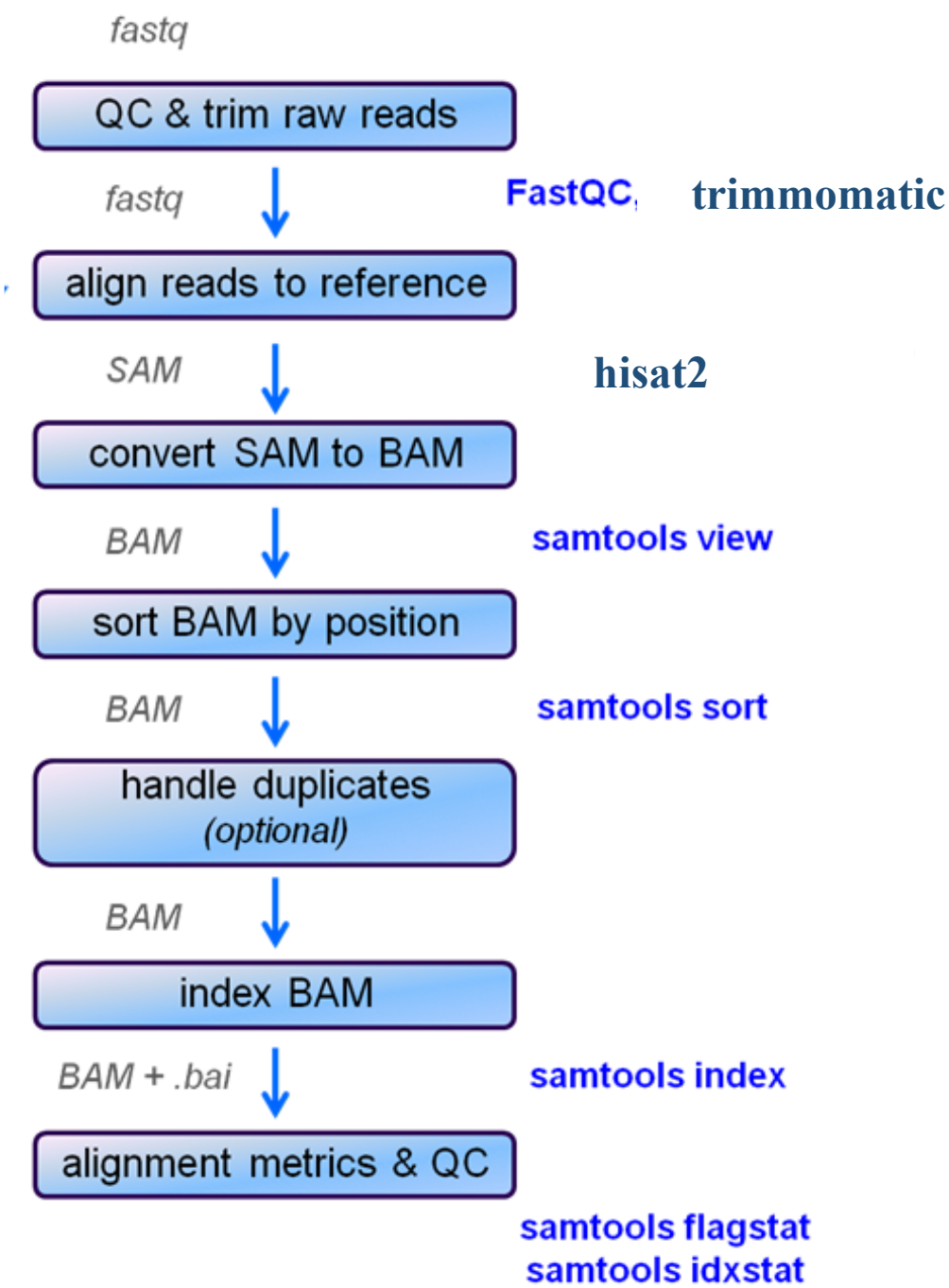
```
TGCAGGATGGATGTGTTCCTCCTCAGCTGCTTA---------------------------TATATTCTTTCGCTGTGTAGTCTGTAAGC

TGCAGGGACTGCAGGATGGATGTGTTCCTCCTCAGCTGCTTATTTTAACTCCACACAACATGTTTTGTGTTATATTCTTTCGCTGTGTAGTCTGTAAGCAGAGTATGATACTG
```

**Reference**

We never want to keep a SAM file - so we immediately convert it to a BAM file

# Alignment Workflow



# BAM file

- BAM (Binary Alignment/Map) format:

- Compressed binary representation of SAM
- Greatly reduces storage space requirements to about 27% of original SAM
- Not human-readable

# Common order of operations

1. SAM files are converted to BAM files (*samtools view*)
2. BAM files are sorted by reference coordinates (*samtools sort*)
3. SORTED BAM files are indexed (*samtools index*)

# samtools view

```
samtools view -b input.sam > input.bam
```

- Input is usually a SAM file, but can also use a BAM
- Common uses: extracting a subset of data into a new file, converting between SAM/BAM files, or just viewing raw files

# samtools sort

```
samtools sort sample.bam -o sample.sorted.bam
```

- Reads need to be ordered in “genomic order” – not the order in which they were sequenced

# samtools index

```
samtools index sorted.bam
```

- Creates index file that allows for fast look-up
- Generates *.bam.bai file