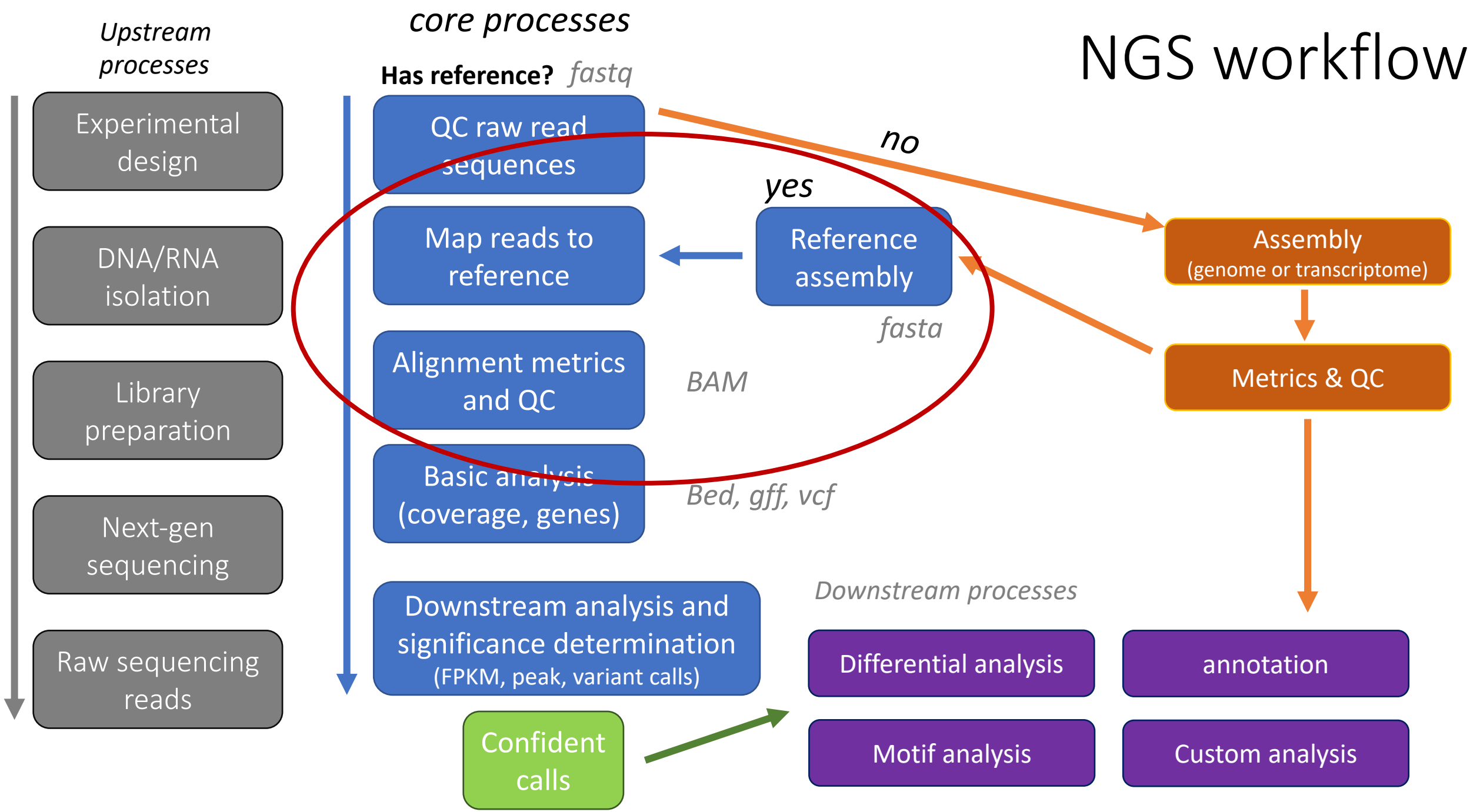
# Outline

1. NGS workflow and experiment types
2. Read sequence terminology
3. The fastq format
4. Fastq QC methods
5. Alignment to a genome
6. Reference genomes: making and using
7. Alignment metrics and QC
8. UCSC genome browser time!

# NGS workflow



# Short Read Aligners

- Short read mappers determine placement of ***query sequences*** (your reads) against a known ***reference***
  - **BLAST**:
    - one query sequence (or a few)
    - many matches for each
  - short read aligners
    - many millions of query sequences
    - want only one “best” mapping (or a few)
- Many aligners available! Two of the most popular
  - **bwa** (Burrows Wheeler Aligner) by Heng Li
    - http://bio-bwa.sourceforge.net/
  - **bowtie2** – part of the Johns Hopkins Tuxedo suite of tools
    - http://bowtie-bio.sourceforge.net/bowtie2/manual.shtml
  - **HiSat2** – a newer aligner part of JHU Suite
    - https://ccb.jhu.edu/software/hisat2/index.shtml
  - Given similar input parameters, they produce similar alignments
    - and both run relatively quickly

# Mapping vs Alignment

- ***Mapping*** determines one or more ***positions*** (a.k.a **seeds** or ***hits***) where a read shares a short sequence with the reference
- ***Alignment*** starts with the seed and determines how read bases are best ***matched***, base-by-base, around the seed
- Mapping quality and alignment scores are both reported
  - High mapping quality ≠ High alignment score
  - ***mapping quality*** describes ***positioning***
    - reflects the probability that the read is *incorrectly* mapped to the reported location
    - is a Phred score:
    - reflects the ***complexity*** or information content of the sequence (***mappability***)
  - ***alignment score*** describes ***fit***
    - reflects the correspondence between the read and the reference sequences

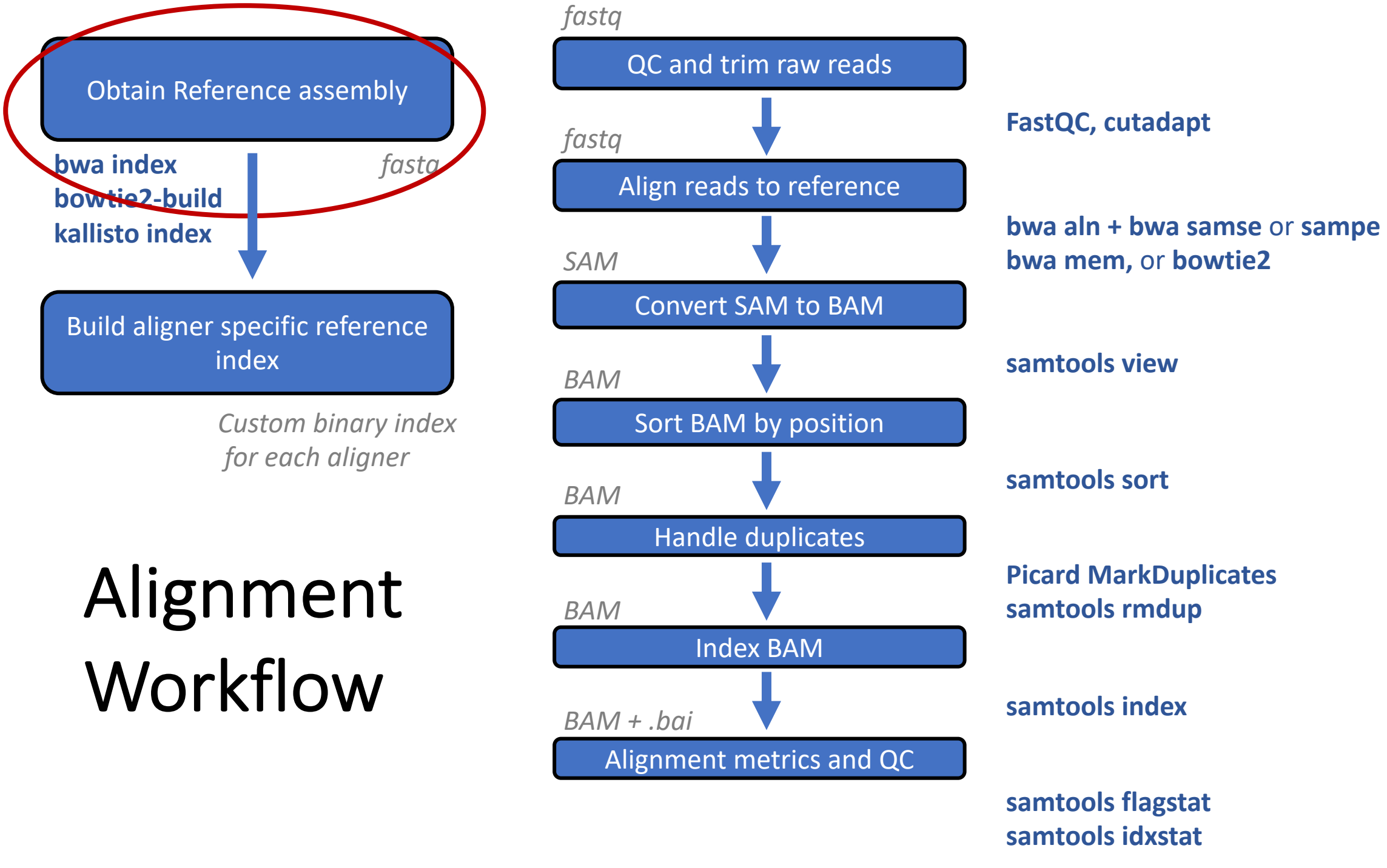
- Maps to one location
*high mapping quality*
- Has 2 mismatches
*low alignment score*


- Maps to 2 locations
*low mapping quality*
- Matches perfectly
*high alignment score*

# Paired End mapping

- Having paired-end reads improves mapping
  - mapping one read with high confidence anchors the pair
    - even when its mate read by itself maps several places equally
- Three possible outcomes of mapping an R1/R2 pair
  1. only one of a pair might map ***(singleton/orphan)***
  2. both reads can map within the most likely distance range ***(proper pair)***
  3. both reads can map but with an unexpected insert size or orientation, or to different contigs ***(discordant pair)***
- Insert size is reported in the alignment record
  - for both proper and discordant pairs

# Alignment Workflow



# Obtaining a reference

- What is a reference?
  - Any set of named sequences
    - Names are typically chromosome names
    - Generally referred to as contigs
- Assembled genomes
  - Ensembl, UCSC, Genbank for eukaryotes
    - FASTA files (.fa, .fasta), + annotations of features (genome feature files, .gff)
  - Genbank, NCBI for prokaryotes/microbes
    - Records contain both fasta sequences and annotations
- Any set of sequences of interest:
  - Transcriptome (set of transcribed gene sequences)
  - miRNA hairpins from mIRBase
  - rRNA/tRNA genes (for filtering)
  - GFP, or a tagged and expressed gene

# FASTA format

- FASTA files contain a set of sequence records
  - Sequence name line
    - Always starts with >
      - Followed by a name and other (optional) descriptive information
  - One or more sequence lines
    - Never starts with >
- Mitochondrial DNA sequence, hg19

```
>chrM
GATCACAGGTCTATCACCCTATTAACCACTCACGGGAGCTCTCCATGCAT
TTGGTATTTTCGTCTGGGGGGTGTGCACGCGATAGCATTGCGAGACGCTG
GAGCCGGAGCACCCTATGTCGCAGTATCTGTCTTTGATTCCTGCCTCATT ...
```

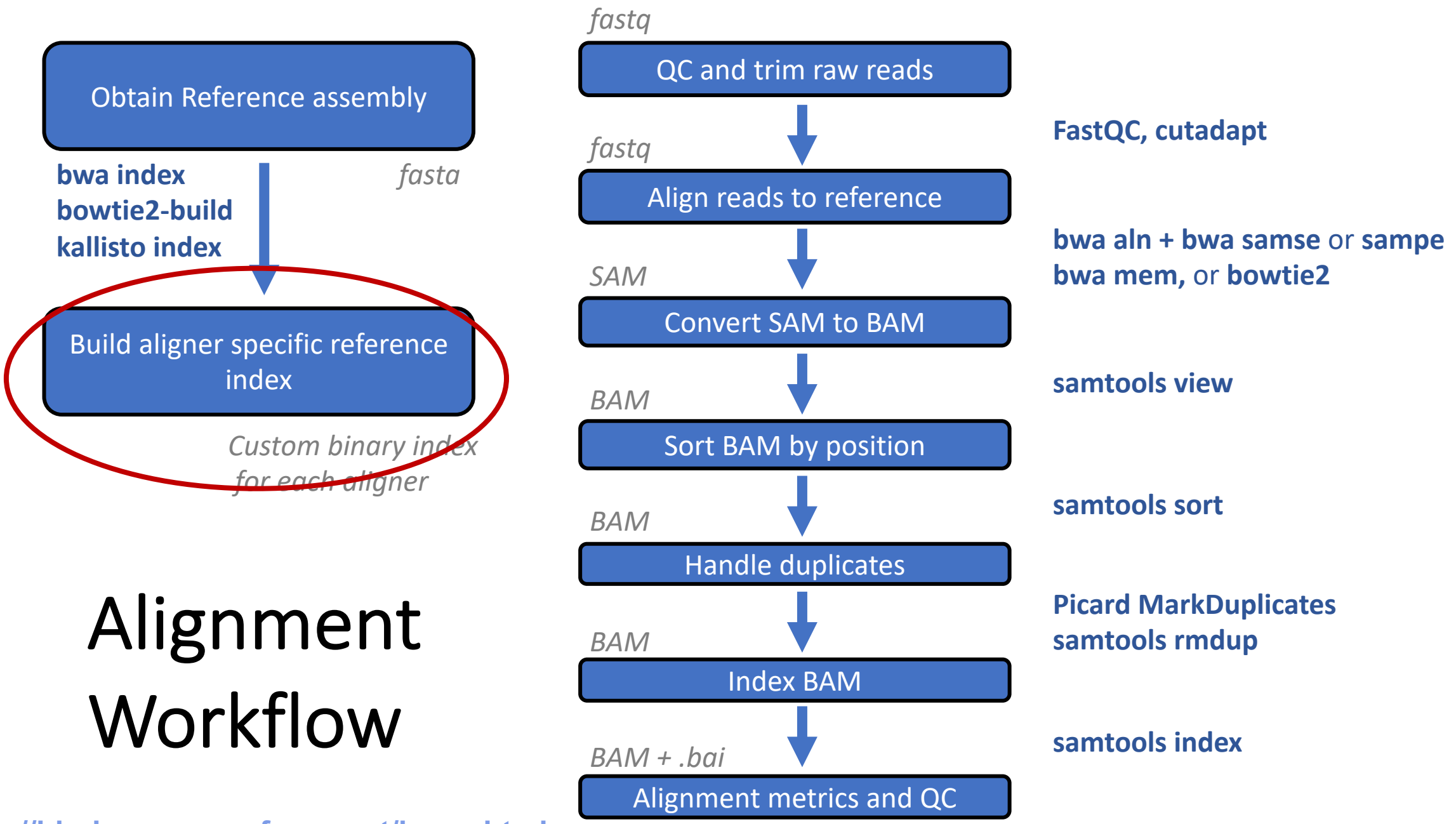
- Let-7e miRNA, mIRBase v21

```
>hsa-let-7e MI0000066 Homo sapiens let-7e stem-loop
CCCGGGCTGAGGTAGGAGGTTGTATAGTTGAGGAGGACACCCAAGGAGATCACTATACGG
CCTCCTAGCTTTCCCCAGG
```

# Reference Consideration

- Does it make sense for your study?
  - Close enough to your species? Complete?
- Does it contain repeats?  What types?
  - Know this up front or you'll be confused
- From which source? And which version?
  - UCSC hg38 vs Ensembl GrCh38
- What annotations exist?
  - References lacking feature annotations are more challenging
    - Need some *de novo* transcriptome assembly, or predictions
- Watch out for sequence name issues!
  - Sequence names may be different between different sources, such as UCSC and Ensembl
    - Chr12 vs 12
  - *Annotation sequence names must match names in your reference!*

# Alignment Workflow



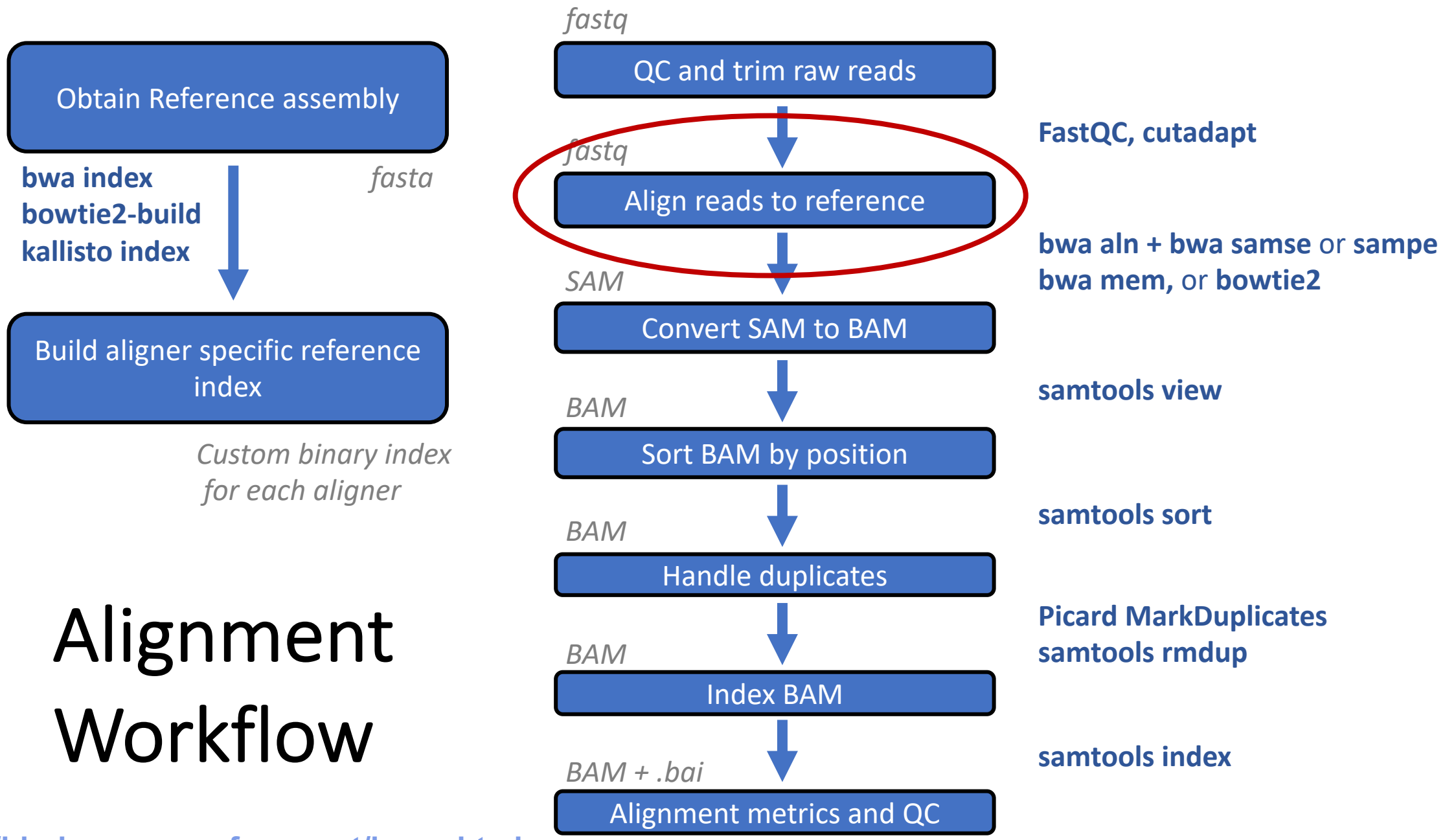

http://bio-bwa.sourceforge.net/bwa.shtml

http://bowtie-bio.sourceforge.net/bowtie2/manual.shtml

# Building a reference index

- Index format is specific to each aligner
  - May take several hours to build
    - But you build once and re-use for using that aligner to that reference
  - Input:
    - One or more fasta files, sometimes a gtf (transcriptome-aware aligners)
  - Output:
    - A number of binary alignment files that the aligner will use
- Best practice
  - Make each index in it’s own well-named directory:
    - /references/bwa/UCSC/hg38
    - /references/kallisto/Ensembl/mm10

# Alignment Workflow




http://bio-bwa.sourceforge.net/bwa.shtml

http://bowtie-bio.sourceforge.net/bowtie2/manual.shtml

# SAM file format

- Aligners take an index file and FASTQ sequences as input
  - Output alignments in **S**equence **A**lignment **M**ap format (SAM)
    - Community file format that describes how reads align to a reference
  - The SAM compendium: http://samtools.github.io/hts-specs/SAMv1.pdf
- SAM file consists of:
  - A header
    - Includes records for reference sequence names and lengths
  - Alignment records, one for each sequence read
    - Can include both mapped and unmapped reads
    - Alignments for R1 and R2 have separate records
      - With fields that refer to the mate
    - 11 fixed fields + extensible-format **key:type:value** tuples

# SAM file format fixed fields

| Col | Field | Type | Regexp/Range | Brief description |
|---|---|---|---|---|
| 1 | QNAME | String | `[!-?A-~]{1,254}` | Query template NAME **Read name from fastq** |
| 2 | FLAG | Int | $[0, 2^{16} - 1]$ | bitwise FLAG |
| 3 | RNAME | String | `\*|[:rname:^*=][:rname:]*` | Reference sequence NAME[11] |
| 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 | `\*|=|[:rname:^*=][:rname:]*` | Reference 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 **Insert size, if paired** |
| 10 | SEQ | String | `\*|[A-Za-z=.]+` | segment SEQuence |
| 11 | QUAL | String | `[!-~]+` | ASCII of Phred-scaled base QUALity+33 |

**Negative for - strand reads**

```
CA3GHANXX161126:5:2301:11371:42747 403 1 14473 0 101M = 14467 -107
GGCTGGGTGGAGCCGTCCCCCCATGGAGCACAGGCAGACAGAAGTCCCCGCCCCAGCTGTGTGGCCTCAAGCCAGCCTTCCGCTCCTTGAAGCTGGTCTCC
FFFFFFF<FFFBFFFBFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFBBBBB
MC:Z:101M MD:Z:101 PG:Z:MarkDuplicates.19 RG:Z:CA3GH.5 NH:i:5 HI:i:2 NM:i:0 MQ:i:0 UQ:i:0 AS:i:196
```

**Positive for + strand reads**

```
CA5MAANXX161126:3:2205:9269:22086 99 1 14474 3 101M = 14565 192
GCTGGGTGGAGCCGTCCCCCCATGGAGCACAGGCAGACAGAAGTCCCCGCCCCAGCTGTGTGGCCTCAAGCCAGCCTTCCGCTCCTTGAAGCTGGTCTC
CA BBBBBBFFBFFFFBFFFFFFBFFF//<FFFFFFFFFFFFFFFFBFBBFFBFFFFFFFFFFFFFFFFBFFFFFFFFFBFFFFBBBF/B<F<<<//F<BFF#
MC:Z:101M MD:Z:101 PG:Z:MarkDuplicates.Z RG:Z:CA5MA.3 NH:i:2 HI:i:0 NM:i:0 MQ:i:3 UQ:i:0 AS:i:200
```

# SAM format: bitwise flags

| Bit Decimal | Hex | 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 |

read did map = 0

plus read strand = 0

1 = part of a read pair
1 = properly paired
1 = read did not map
1 = mate did not map
1 = minus read strand
1 = mate on minus strand
1 = R1 read
1 = R2 read
1 = secondary alignment
1 = marked as duplicate
1 = maps to ALT contig

| Decimal | Hex |
|---|---|
| 99 | = 0x63 |
| = 64 | = 0x40 |
| + 32 | + 0x20 |
| + 2 | + 0x02 |
| + 1 | + 0x01 |

```
CA3GHANXX161126:5:2301:11371:42747 403 1 14473 0 101M = 14467 -107
GGCTGGGTGGAGCCGTCCCCCCATGGAGCACAGGCAGACAGAAGTCCCCGCCCCAGCTGTGTGGCCTCAAGCCAGCCTTCCGCTCCTTGAAGCTGGTCTCC
FFFFFFF<FFFBFFFBFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFBBBBB
MC:Z:101M MD:Z:101 PG:Z:MarkDuplicates.19 RG:Z:CA3GH.5 NH:i:5 HI:i:2 NM:i:0 MQ:i:0 UQ:i:0 AS:i:196

CA5MAANXX161126:3:2205:9269:22086 99 1 14474 3 101M = 14565 192
GCTGGGTGGAGCCGTCCCCCCATGGAGCACAGGCAGACAGAAGTCCCCGCCCCAGCTGTGTGGCCTCAAGCCAGCCTTCCGCTCCTTGAAGCTGGTCTC
CA BBBBBBFFBFFFFBFFFFFFBFFF//<FFFFFFFFFFFFFFFFBFBBFFBFFFFFFFFFFFFFFFBFFFFFFFFFBFFFFBBBF/B<F<<<//F<BFF#
MC:Z:101M MD:Z:101 PG:Z:MarkDuplicates.Z RG:Z:CA5MA.3 NH:i:2 HI:i:0 NM:i:0 MQ:i:3 UQ:i:0 AS:i:200
```

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

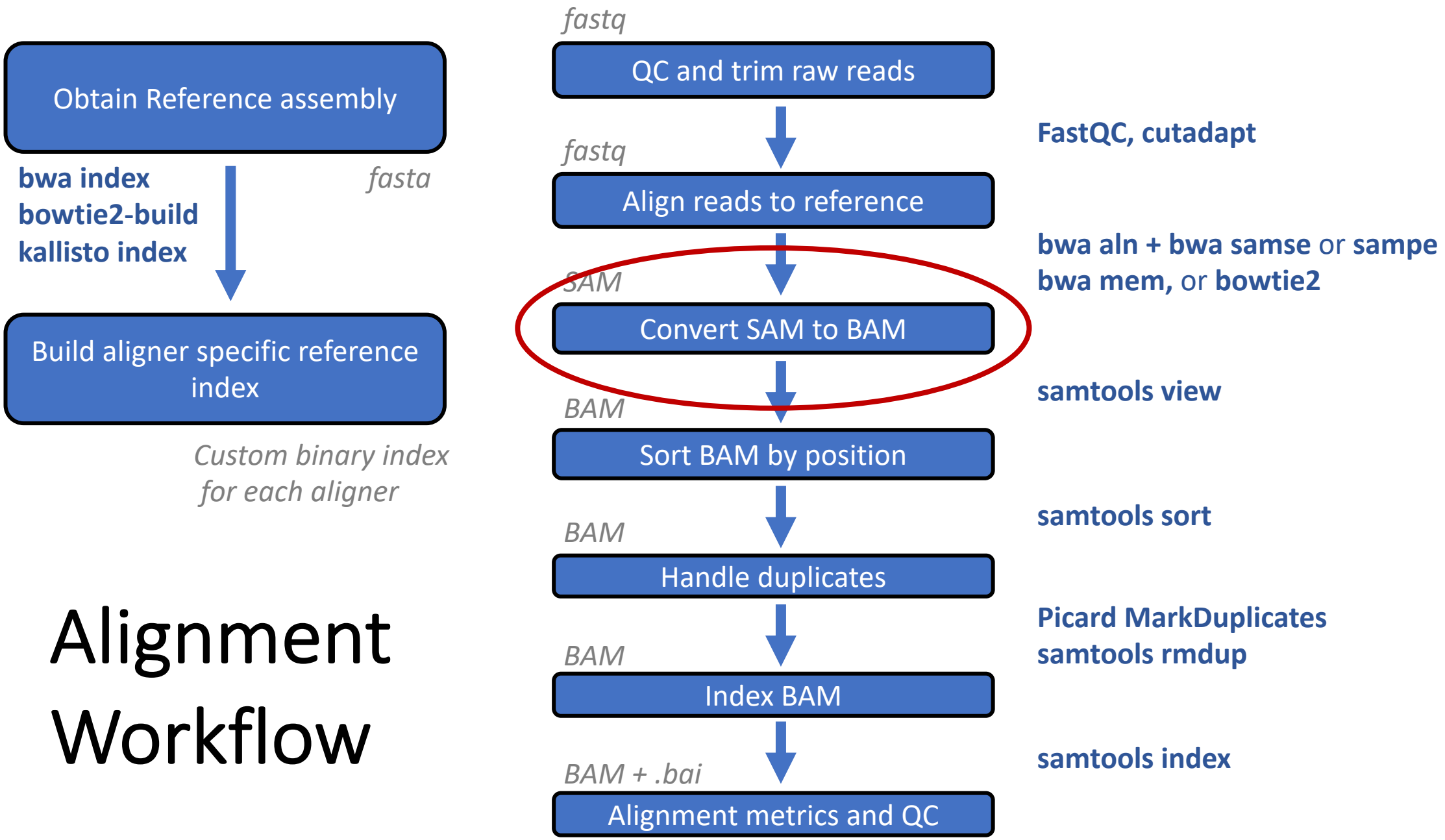
# Interpreting CIGAR strings

| Op | BAM | Description | Consumes query | Consumes reference |
|---|---|---|---|---|
| M | 0 | alignment match (can be a sequence match or mismatch) | yes | yes |
| I | 1 | insertion to the reference | yes | no |
| D | 2 | deletion from the reference | no | yes |
| N | 3 | skipped region from the reference Splicing event in RNA-seq BAMs | no | yes |
| S | 4 | soft clipping (clipped sequences present in SEQ) | yes | no |
| H | 5 | hard clipping (clipped sequences NOT present in SEQ) | no | no |
| P | 6 | padding (silent deletion from padded reference) | no | no |
| = | 7 | sequence match | yes | yes |
| X | 8 | sequence mismatch | yes | yes |

```
CA3GHANXX161126:2:1211:9321:65511 99 1 14740 1 41S90M140N10M = 15913 1931
AGCGGTGGCGGCAGAGGAGGGATGGAGTCTGACACGCGGGCAAAGGCTCCTCCGGGCCCCTCACCAGCCCCAGGTCCTTTCCCAGAGATGCCCTTGCGCCT
BBBBBFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFBFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFB
MC:Z:2S35M659N64M MD:Z:100 PG:Z:MarkDuplicates.1A RG:Z:CA3GH.2 NH:i:3 HI:i:1 NM:i:0 MQ:i:1 UQ:i:0 AS:i:196
XS:A:-
```

CIGAR = “Concise Idiosyncratic Gapped Alignment Report”

# Alignment Workflow

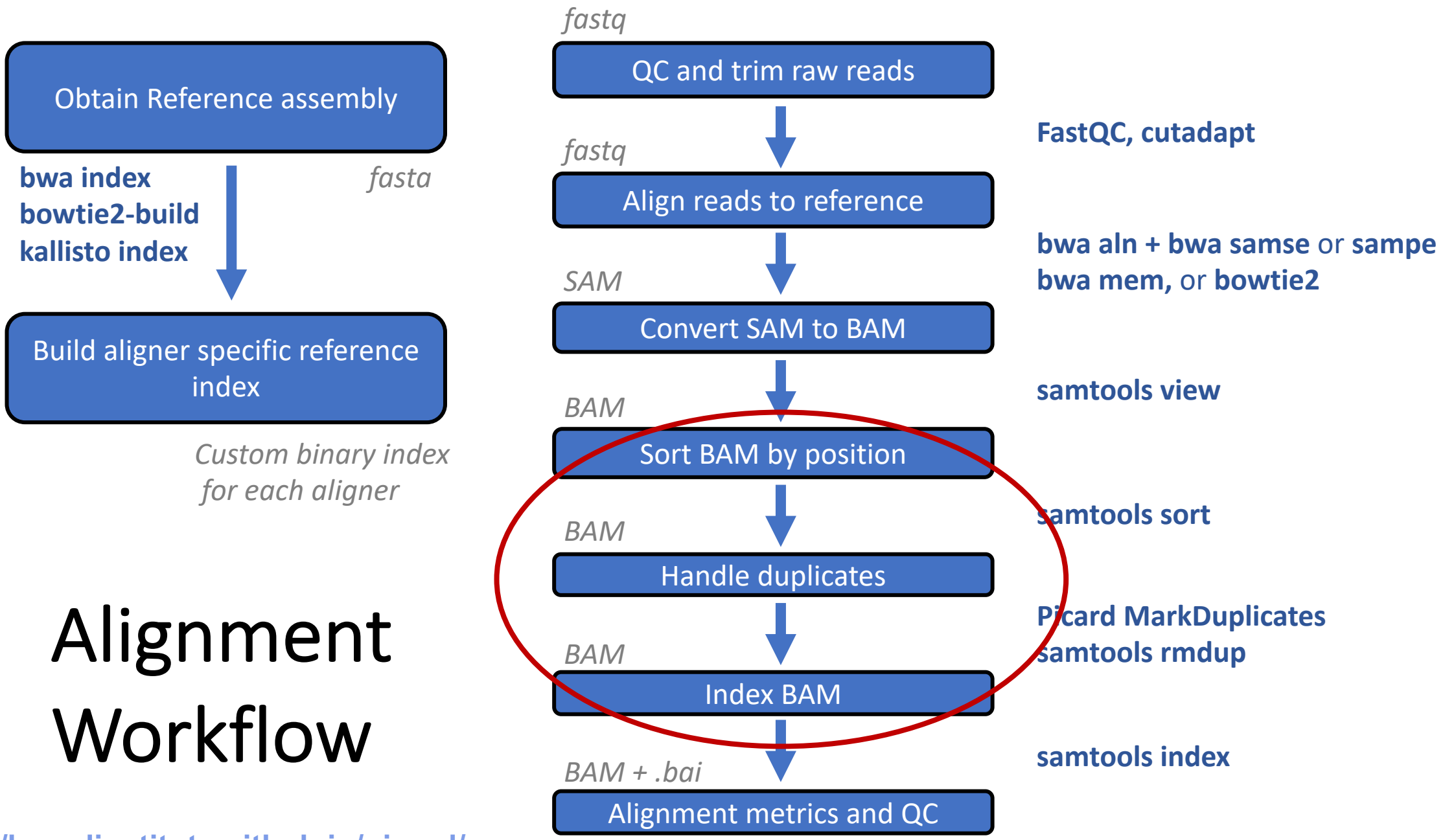



http://samtools.sourceforge.net/samtools.shtml

# SAM/BAM files

- SAM and BAM are two forms of the same data
  - SAM – **S**equence **A**lignment **M**ap
    - plain text format
- BAM – **B**inary **A**lignment **M**ap
  - ***same data*** in a custom compressed (**gzip**'d) format
- Differences
  - BAMs are ***much*** smaller than SAM files due to compression
  - BAM files support fast random access; SAM files do not
    - requires the BAM file to be *indexed*
  - most tools support BAM format and may require indexing
- Best practices
  - remove intermediate SAM and BAM files created during alignment and only save the final sorted, indexed BAM
  - keep your alignment artifacts (BAM, statistics files, log files) separate from the original FASTQ files
    - alignments can be re-generated – raw sequences cannot

# Alignment Workflow




http://broadinstitute.github.io/picard/

http://samtools.sourceforge.net/samtools.shtml

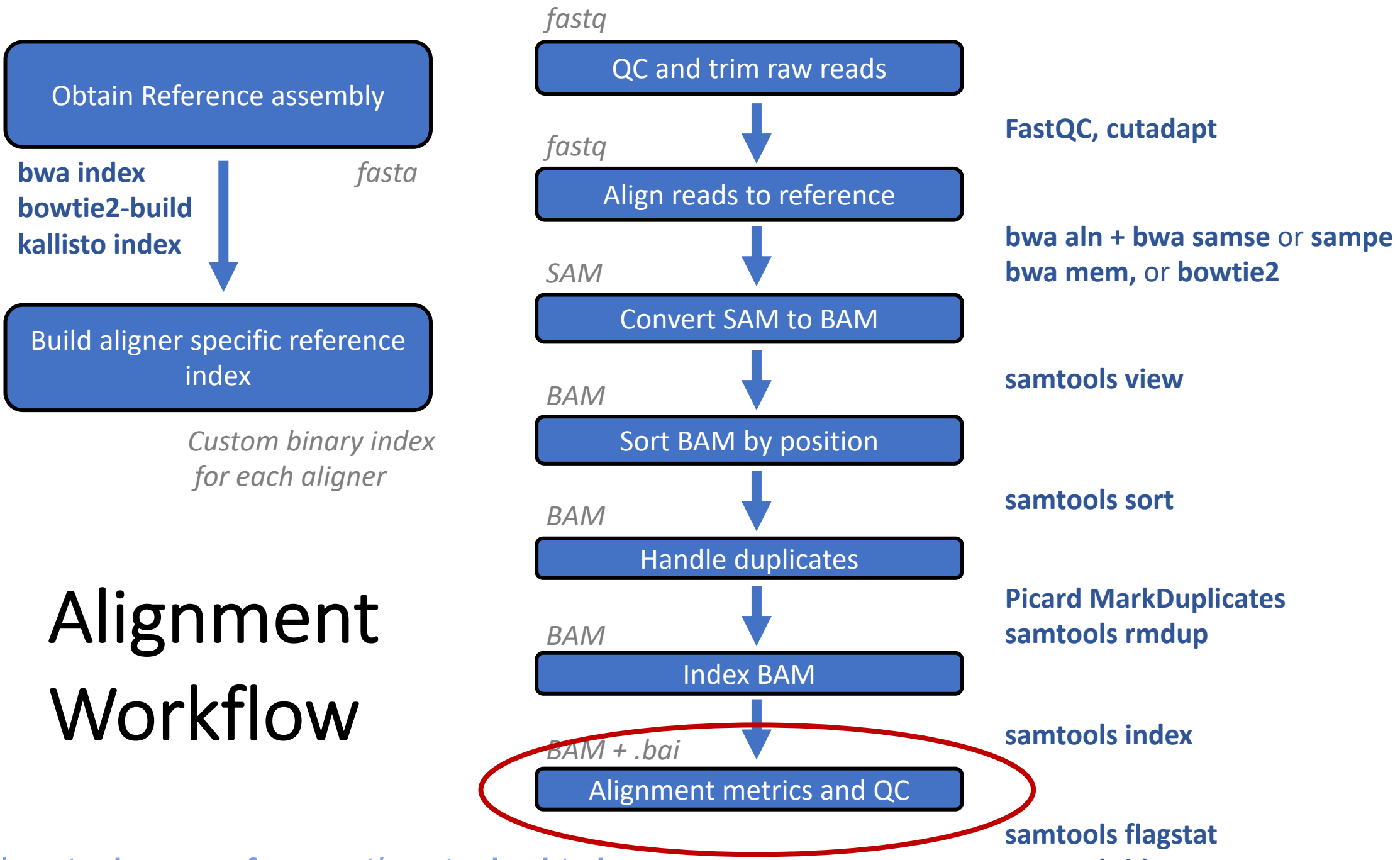
# Sorting/indexing SAM/BAM files

- SAM created by aligner contains read records in ***name order***
  - same order as read names in the input FASTQ file
    - R1, R2 have adjacent SAM records
  - SAM → BAM conversion does not change the name-sorted order
- Sorting BAM puts records in ***position (locus) order***
  - by contig name then start position (leftmost)
    - contig name order given in SAM/BAM header
    - based on order of sequences in FASTA used to build reference
  - ***sorting is very compute and I/O intensive***
    - can take hours for large BAM
- Indexing a locus-sorted BAM allows fast random access
  - creates a binary alignment index file **(.bai**)
  - quite fast

# Handling Duplicates

- Optional step, but very important for many protocols
- Definition of ***alignment duplicates***:
  - single-end reads or singleton/discordant PE alignment reads
    - alignments have the same ***start*** positions
  - properly paired reads
    - pairs have same ***external*** coordinates (5’ + 3’ coordinates of the ***insert***)
- Two choices for handling:
  - **samtools rmdup** – ***removes*** duplicates entirely
    - faster, but data is lost
    - does not intelligently handle data from multiple lanes
  - **Picard MarkDuplicates** – ***flags*** duplicates only (**0x400** bam flag)
    - slower, but all alignments are retained
    - alignments from different lanes/replicates are considered separately
    - also newer **MarkDuplicatesWithMateCigar** tool
      - takes CIGAR string(s) into account; slower than plain **MarkDuplicates**
  - both tools are quirky in their own ways

# Alignment Workflow




http://samtools.sourceforge.net/samtools.shtml

# Alignment metrics

- Samtools flagstat
  - simple statistics based on alignment record flag values
    - total sequences (R1+R2), total mapped
    - number properly paired
    - number of duplicates (0 if duplicates were not marked)

```
samtools flagstat LA1221_CTCF.sort.clean.dup.bam
114865076 + 0 in total (QC-passed reads + QC-failed reads)
0 + 0 secondary
0 + 0 supplementary
4681545 + 0 duplicates
112275199 + 0 mapped (97.75% : N/A)
114865076 + 0 paired in sequencing
57432538 + 0 read1
57432538 + 0 read2
111052396 + 0 properly paired (96.68% : N/A)
111528682 + 0 with itself and mate mapped
746517 + 0 singletons (0.65% : N/A)
235373 + 0 with mate mapped to a different chr
125919 + 0 with mate mapped to a different chr (mapQ>=5)
```

# Alignment metrics

- Samtools idxstats
  - Reads aligning to each contig

| contig | contig length | # map | # not mapped |
|---|---|---|---|
| 1 | 249250621 | 4815867 | 47995 |
| 2 | 243199373 | 2255986 | 21349 |
| 3 | 198022430 | 1714479 | 15927 |
| 4 | 191154276 | 1187721 | 11459 |
| 5 | 180915260 | 1648589 | 15497 |
| 6 | 171115067 | 1600135 | 13607 |
| 7 | 159138663 | 1720016 | 16749 |
| 8 | 146364022 | 1000096 | 9261 |
| 9 | 141213431 | 1057255 | 9747 |
| 10 | 135534747 | 1832309 | 16808 |
| 11 | 135006516 | 1747831 | 16028 |
| 12 | 133851895 | 2250062 | 20083 |
| 13 | 115169878 | 528022 | 5131 |
| 14 | 107349540 | 1363426 | 12641 |
| 15 | 102531392 | 1458562 | 12583 |
| 16 | 90354753 | 986821 | 9153 |
| 17 | 81195210 | 1655054 | 16213 |
| 18 | 78077248 | 496684 | 4706 |
| 19 | 59128983 | 1487864 | 13911 |
| 20 | 63025520 | 722429 | 6869 |
| 21 | 48129895 | 267246 | 2398 |
| 22 | 51304566 | 890182 | 7457 |
| X | 155270560 | 855129 | 8464 |
| Y | 59373566 | 22902 | 303 |
| MT | 16569 | 10836924 | 74267 |

# Samtools Notes

- There are 2 main “eras” of the **samtools** program
  - “old” **samtools**
    - v 0.1.19 last stable version
  - “new” **samtools**
    - v 1.0, 1.1, 1.2 – avoid these (very buggy!)
    - v 1.3+ stable
  - some functions have different arguments!
- **samtools** v 1.3+ has many new features
  - **samtools stats**
    - produces *many* different statistical reports
  - faster sorting
    - can use multiple threads
- On prem:
  - -bash:login02:~ $ use -la | grep sam
    - .**sam**tools-1.5 - .**sam**tools
    - .**sam**tools-1.7 - .**sam**tools
    - .**sam**tools-1.8 - .**sam**tools

# Computing average insert size

- Needed for some downstream analysis
  - e.g. RNAseq alignment using **tophat**
- Simple **awk** script that computes average insert size for a BAM
  - **-F 0x4** filter to **samtools view** says only consider mapped reads
    - technically “not unmapped”
  - the **-f 0x2** filter says consider only properly paired reads
    - they have reliable “insert size" values in column 9
  - insert size values are negative for minus strand reads
    - can ignore because each proper pair will have one plus and one minus strand alignment, with same insert size

```
samtools view -F 0x4 -f 0x2 my_pe_data.bam | awk \
  'BEGIN{ FS="\t"; sum=0; nrec=0; }
   { if ($9 > 0) {sum += $9; nrec++;} }
   END{ print sum/nrec; }'
```

# Interpreting alignment metrics

## ChIP-seq alignment metrics

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | LA sample | factor | date (MM_YYYY) | seq tech | PE/SE | read length | aligner | total # reads | # aligned reads | alignment rate | # duplicate reads | dupe rate | properly paired | properly paired percentage | notes |
| 2 | LA1623 | CTCF | 03_2019 | NextSeq | PE | 75 | bwa | 2,930,063 | 2,834,760 | 96.75% | 27,264 | 0.96% | 2,803,992 | 95.70% | |
| 3 | LA1623 | h3k4me1 | 03_2019 | NextSeq | PE | 75 | bwa | 145,647,782 | 142,067,894 | 97.54% | 4,936,246 | 3.47% | 140,719,623 | 96.62% | |
| 4 | LA1623 | h3k4me3 | 03_2019 | NextSeq | PE | 75 | bwa | 125,837,626 | 121,664,130 | 96.68% | 4,510,682 | 3.71% | 120,016,594 | 95.37% | |
| 5 | LA1623 | h3k27ac | 03_2019 | NextSeq | PE | 75 | bwa | 153,609,130 | 151,439,523 | 98.59% | 5,025,127 | 3.32% | 150,455,923 | 97.95% | |
| 6 | LA1623 | h3k27me3 | 03_2019 | NextSeq | PE | 75 | bwa | 141,557,430 | 137,370,397 | 97.04% | 5,520,637 | 4.02% | 135,612,149 | 95.80% | |
| 7 | LA1623 | h3k9me3 | 03_2019 | NextSeq | PE | 75 | bwa | 139,526,114 | 133,880,148 | 95.95% | 5,403,496 | 4.04% | 131,643,343 | 94.35% | |
| 8 | LA1623 | h3k4me3 | 03_2019 | NextSeq | PE | 75 | bwa | 125,837,626 | 121,664,130 | 96.68% | 4,510,682 | 3.71% | 120,016,594 | 95.37% | |
| 9 | LA1623 | input | 03_2019 | NextSeq | PE | 75 | bwa | 99,090,385 | 96,712,278 | 97.60% | 5,425,748 | 5.61% | 95,763,494 | 96.64% | |
| 10 | LA1623 | IgG | 02_2019 | NextSeq | PE | 75 | bwa | 39,595,877 | 37,349,783 | 94.33% | 1,144,191 | 3.06% | 36,913,806 | 93.23% | |

## RNA-seq alignment metrics

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | sample | total reads | pseudoaligned reads | alignment rate | avg frag length |
| 2 | P01202_LA | 28,391,864 | 25,246,357 | 88.92% | 276.211 |
| 3 | P01623_LA | 31,015,355 | 27,578,832 | 88.92% | 315.267 |
| 4 | P01202_LV | 79,917,658 | 69,582,385 | 87.07% | 276.342 |
| 5 | P01623_LV | 33,851,669 | 29,651,916 | 87.59% | 117.84 |
| 6 | P01221_LA | 24,597,427 | 22,118,400 | 89.92% | 288.833 |
| 7 | P01221_LV | 32,875,280 | 29,402,504 | 89.44% | 174.749 |
| 8 | P01294_LA | 9,275,813 | 8,570,593 | 92.40% | 284.892 |
| 9 | P01294_LV | 40,533,408 | 35,667,609 | 88.00% | 194.794 |
| 10 | P01600_LA | 26,645,101 | 23,879,872 | 89.62% | 298.685 |
| 11 | P01600_LV | 28,881,930 | 24,709,271 | 85.55% | 252.443 |
| 12 | | 33,598,551 | 29,640,774 | 88.22% | |

# Alignment wrap-up

- Many tools involved
  - choose one or two and learn their options well
- Many steps are involved in the full alignment workflow
  - important to go through manually a few times for learning
    - but gets tedious quickly!
  - best practice
    - automate series of complex steps by wrapping into a ***pipeline script***
      - e.g. **bash** or **python** script
    - Use variables here!
      - https://www.lifewire.com/pass-arguments-to-bash-script-2200571

/medpop/afib/ahall/code/batch/chip_seq_bwa_align_LA1221_CTCF_2019_08_07.sh

# Final thoughts….

- Good judgement comes from experience
  - *unfortunately…*
- Experience comes from bad judgement!
- So go get started making your 1st 1,000 mistakes….
  - Just don’t rm-rf anything without knowing exactly what you are doing!!