gatk talks

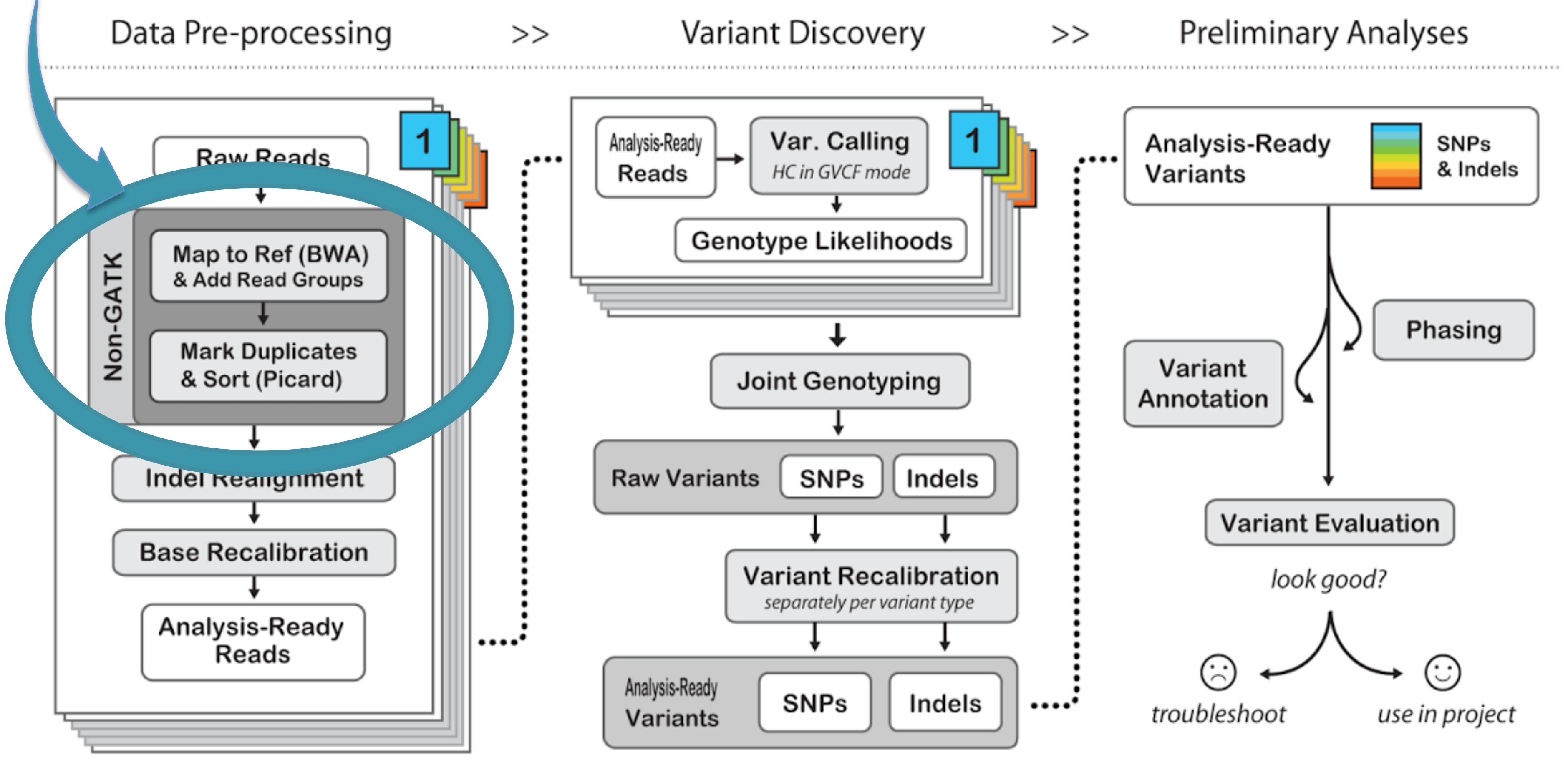
Mapping, processing, and duplicate marking with Picard tools

From raw basecalls to GATK-ready reads

BROAD INSTITUTE

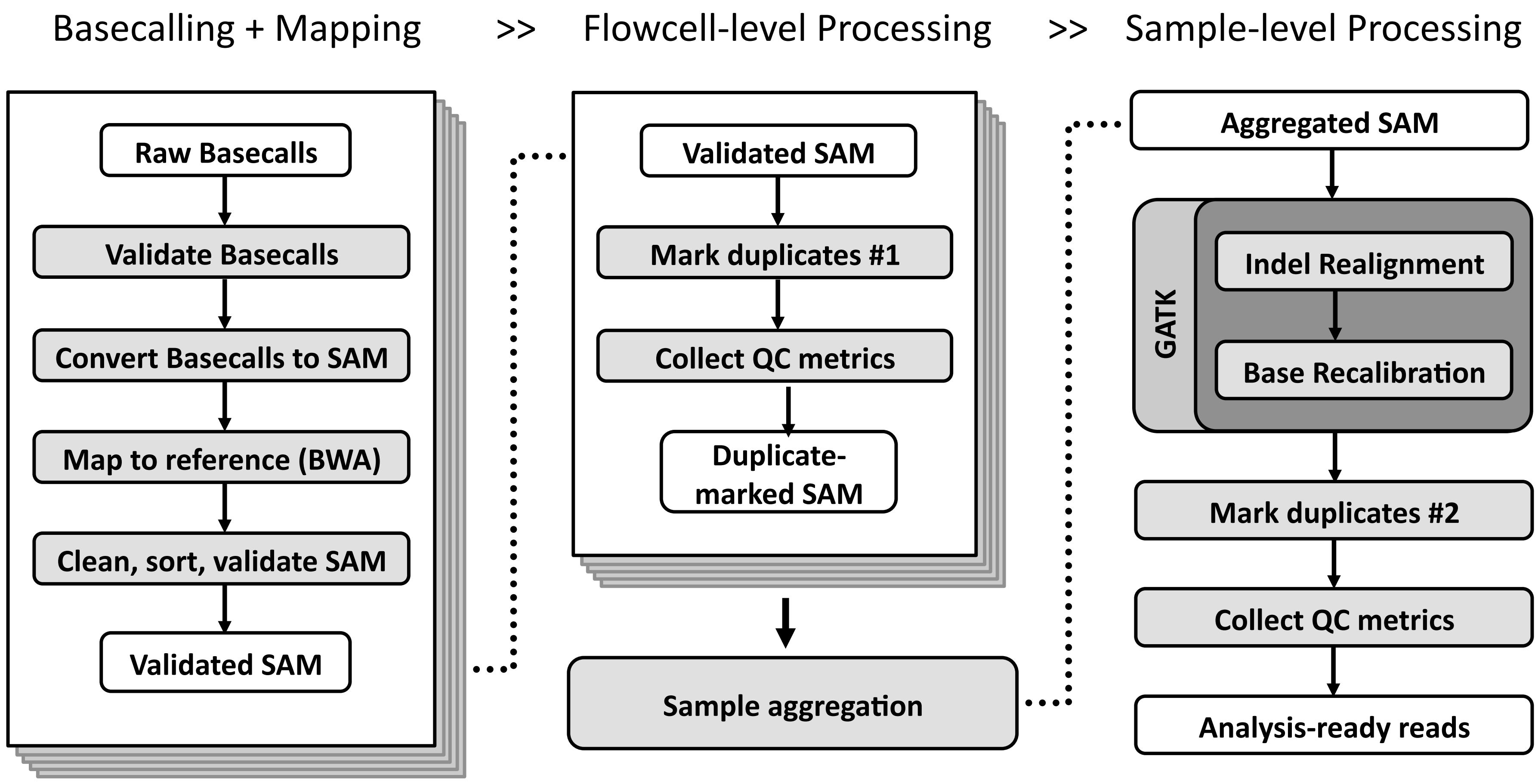
We are here in the Best Practices workflow

Mapping and Marking Duplicates



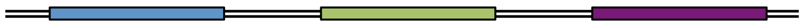
Lots of hidden complexity

Mapping and Marking Duplicates

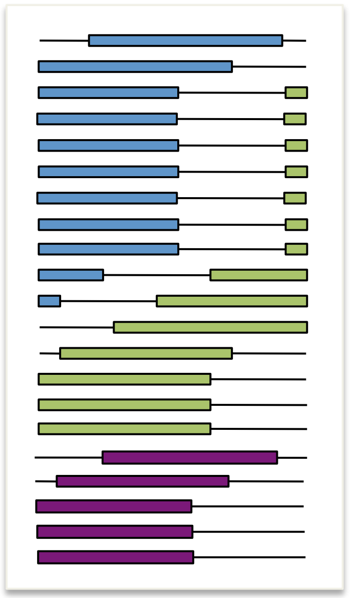


Overview of Picard pipeline

Reference genome



Enormous pile of short reads from NGS



We need to:

- Convert raw basecalls to SAM reads
- Map reads to reference with BWA
- Mark PCR duplicates
- Perform general SAM processing
  - Clean, validate, sort, merge
- Collect QC metrics

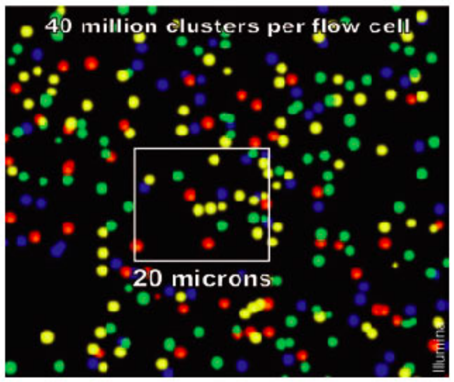
…Make it so, Picard!

BASECALLING (ILLUMINA)

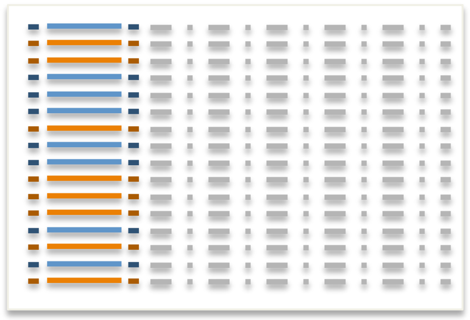
Best practice is to use **IlluminaBasecallsToSam**

- Validate basecalls with **CheckIlluminaDirectory**
  - Sequencing run creates a directory with basecalling (image) files
  - Organized by lane, cycle, etc.
- Create unmapped SAM file from basecalls with **IlluminaBasecallsToSam**
  - Unlike FASTQ, SAM can store file-level metadata (RG, LB, etc.)
- Low-level QC metrics to assess quality of run

Sequencer



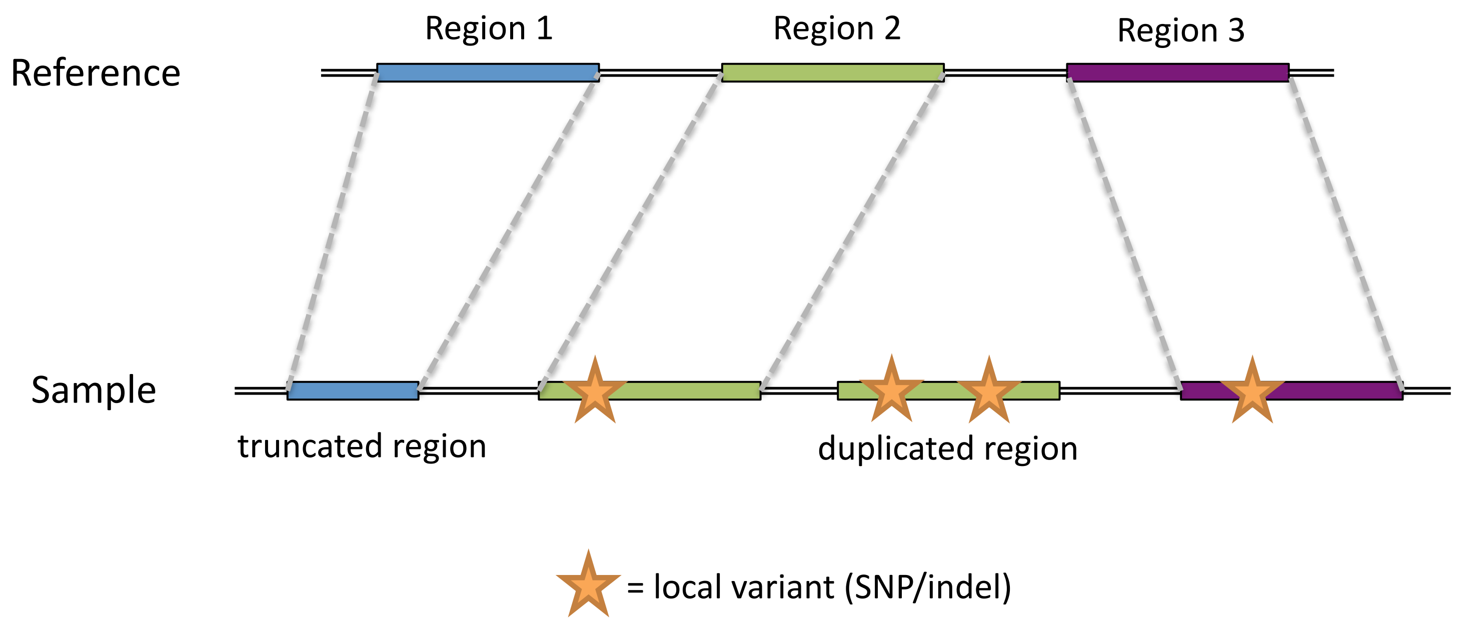
Basecalling (BCL) files


Unmapped SAM files

Images © Illumina, Inc.

MAPPING

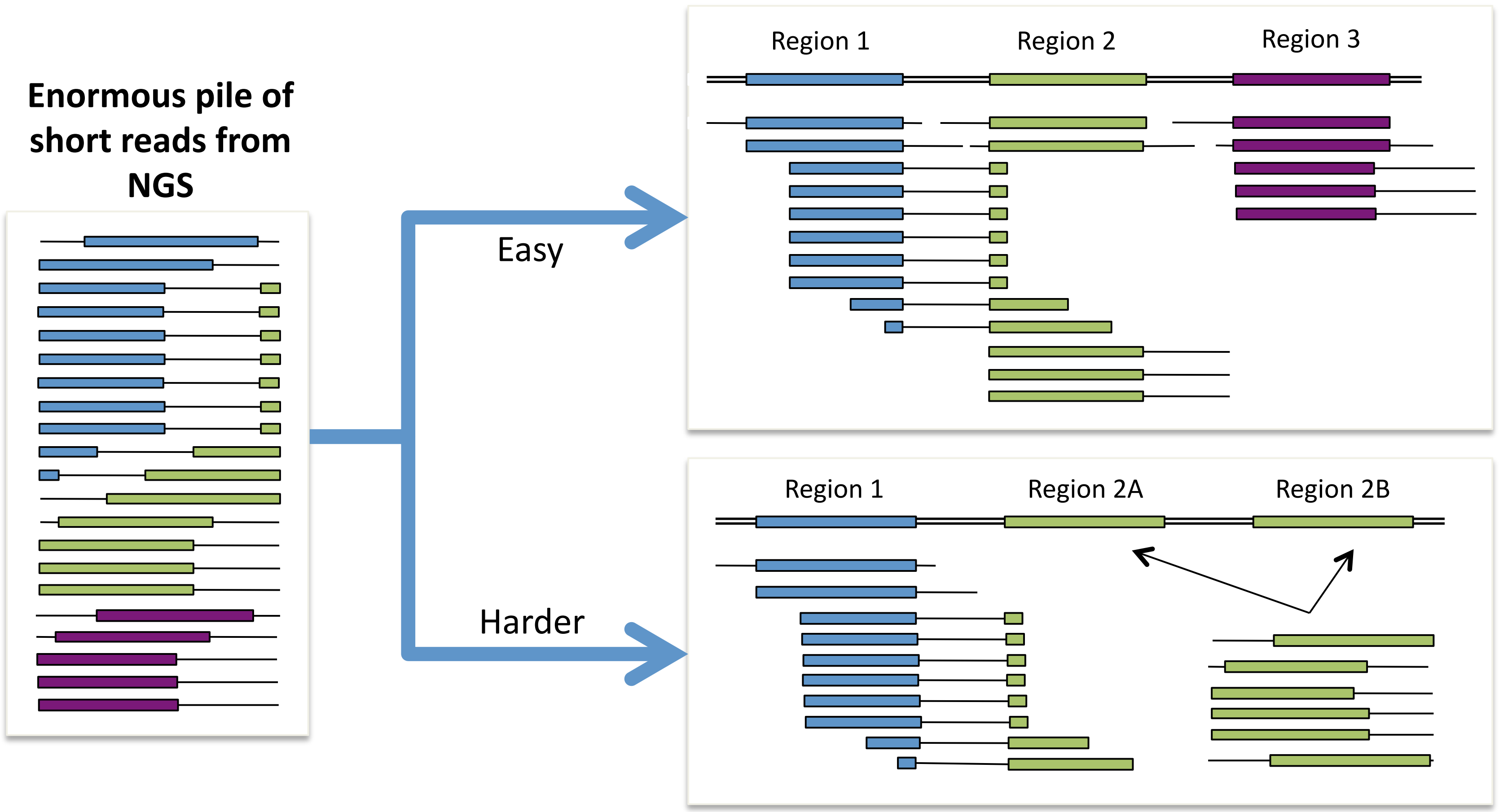
Ideally, we'd just map the sample genome to the reference genome



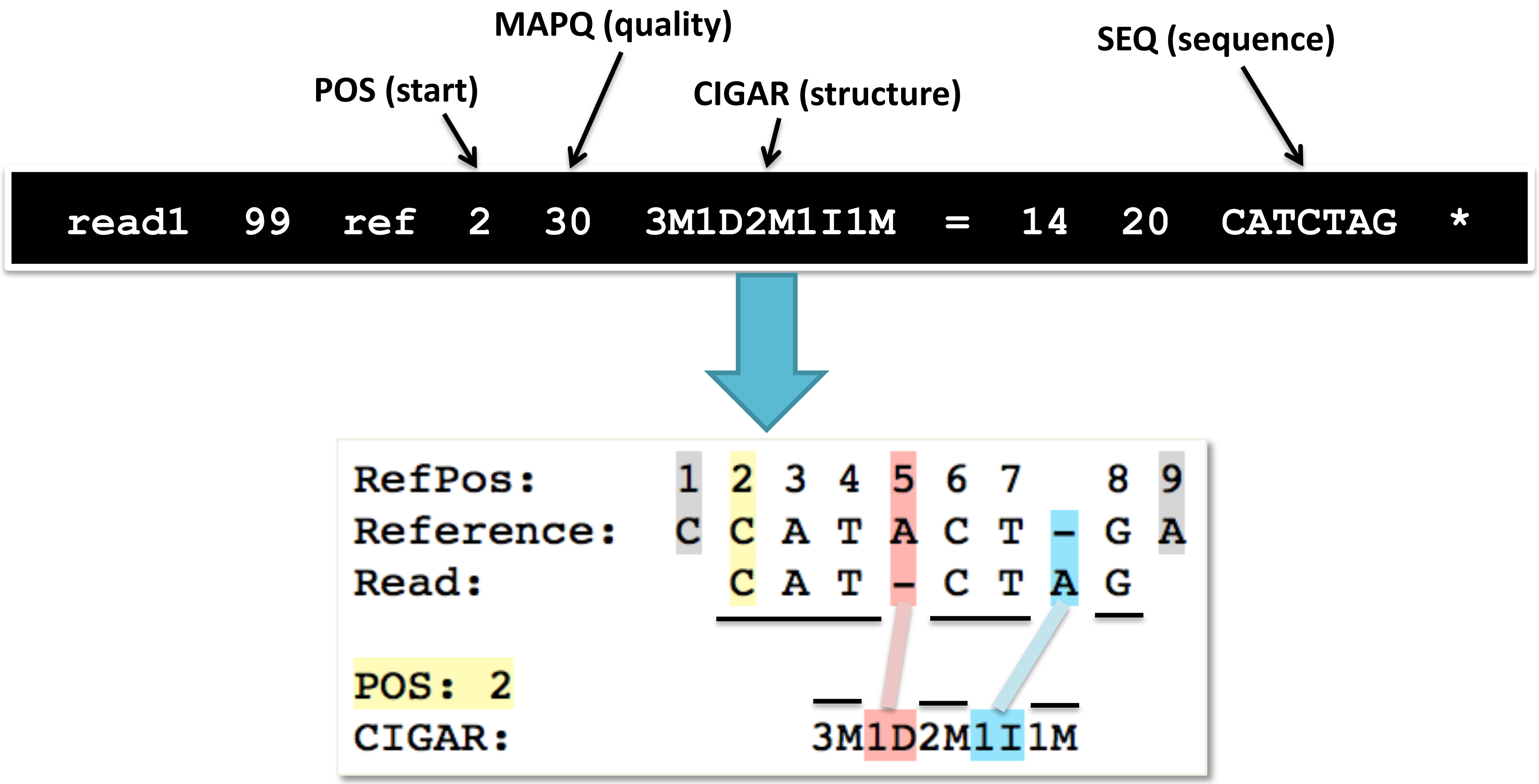
…But we don't have the whole sample in one piece.

Mapping reads to reference is simple in principle…

…but is complicated by mismatches (true mutations or sequencing errors), indels, duplicated regions etc.



Anatomy of a SAM alignment



```
read1   99   ref   2   30   3M1D2M1I1M   =   14   20   CATCTAG   *
```

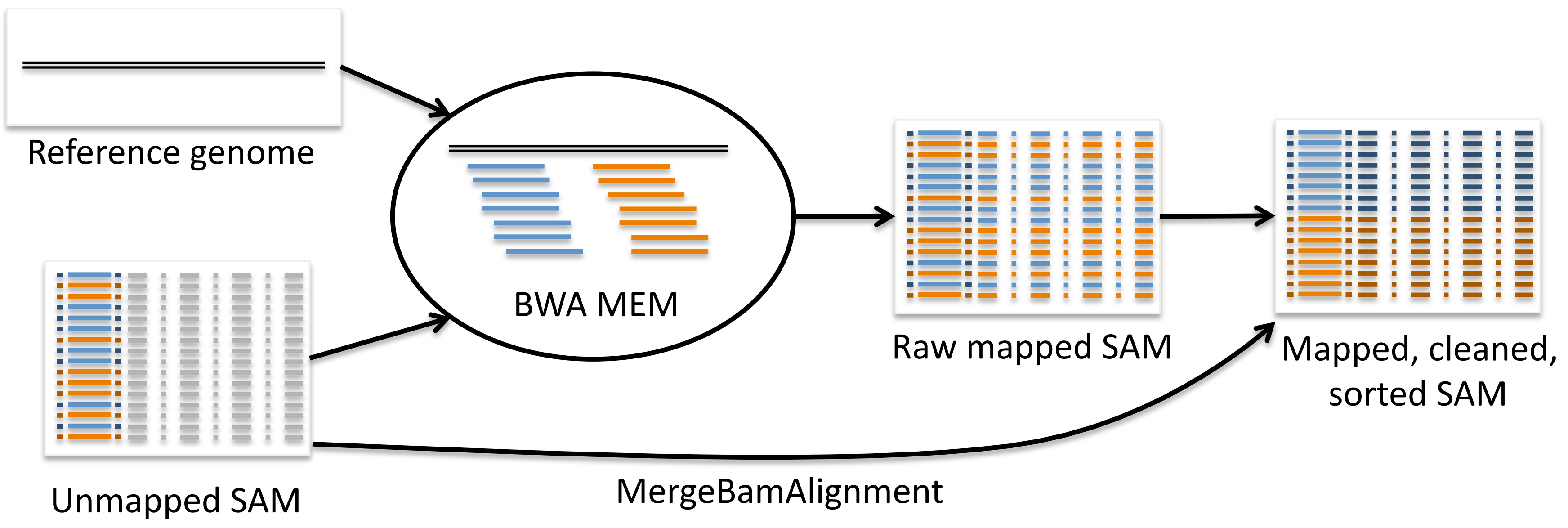
```
RefPos:       1 2 3 4 5 6 7   8 9
Reference:    C C A T A C T - G A
Read:           C A T - C T A G

POS: 2
CIGAR:          3M1D2M1I1M
```

See also:

- SAM format spec: http://samtools.github.io/hts-specs/SAMv1.pdf
- Explain SAM flags: http://broadinstitute.github.io/picard/explain-flags.html

Best practices mapping using BWA + Picard

- Use **BWA MEM** algorithm
- Raw BWA output may not be up to our quality standards
  - Alignments clipping end of reference, missing mates...
- Combine output with original unmapped SAM using **MergeBamAlignment**
  - Fixes up various mapping issues to make downstream processing easier
  - Sorts reads by mapping position + adds read group info as well



Hey, what about RNAseq?

- Use STAR aligner instead of BWA
- Additional post-mapping step (SplitNCigarReads)

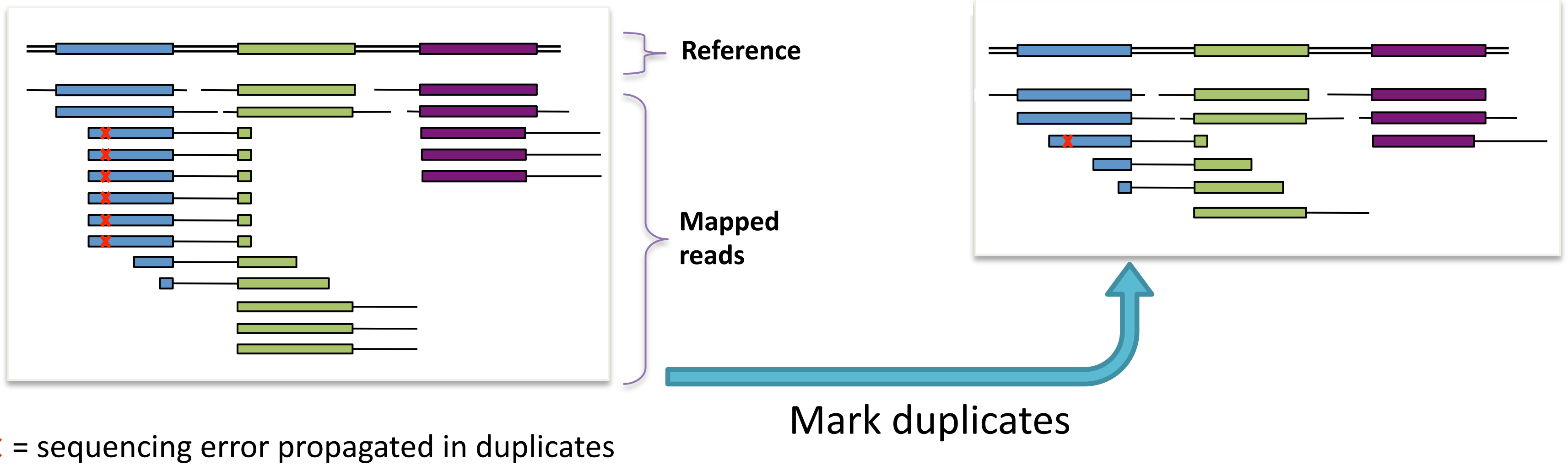
- But MarkDuplicates + other Picard steps are unchanged

- Stay tuned for RNAseq presentation…

MARKING DUPLICATES

Why mark duplicates?

- They’re **non-independent measurements** of a sequence
  - Sampled from the exact same template of DNA
  - Violates assumptions of variant calling
- What’s more, errors in sample/library prep will get propagated to *all* the duplicates
  - Just pick the “best” copy – mitigate the effects of errors



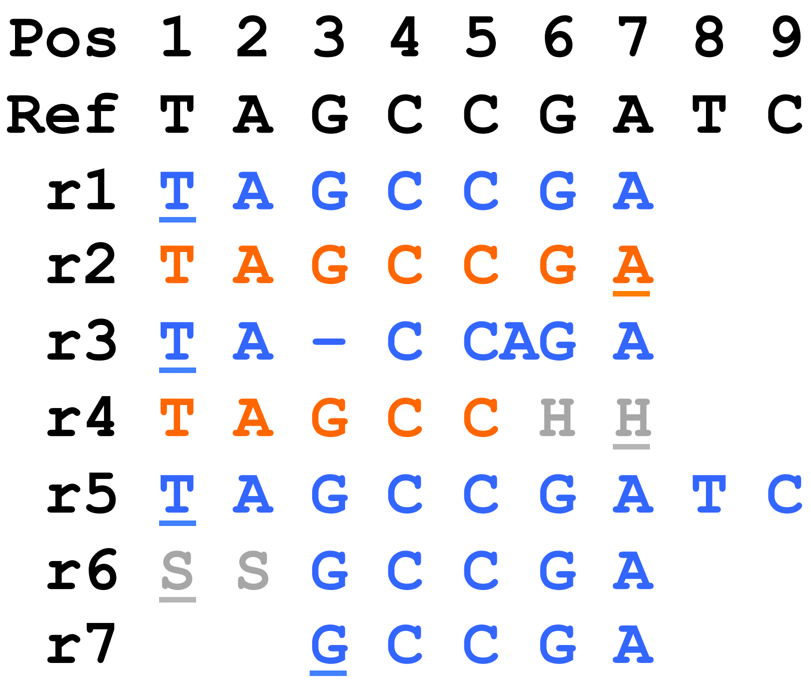
How do we identify duplicates?

- Dupes come from the same input DNA template…so should have same start position on reference!
  - “Where was the first base that was sequenced?”
  - For PE reads, same start for both ends
- Identify duplicate sets, then choose representative read based on base quality scores and other criteria

But there’s a catch (or two)…

- BWA sometimes “clips” bases from the ends of the alignment
- Fragments mapped to the reverse strand are specified by their 3’ position, instead of 5’
- Need to use SAM flags + CIGAR string to determine the unclipped 5’ end

Identify duplicates using orientation + “unclipped” 5’ position



Pos	1	2	3	4	5	6	7	8	9
Ref	T	A	G	C	C	G	A	T	C
r1	T	A	G	C	C	G	A		
r2	T	A	G	C	C	G	A		
r3	T	A	–	C	CAG		A		
r4	T	A	G	C	C	H	H		
r5	T	A	G	C	C	G	A	T	C
r6	S	S	G	C	C	G	A		
r7			G	C	C	G	A		

Blue maps to forward strand
Orange maps to reverse strand
Grey bases are clipped

Underlined is the expected 5’ start of the read, given the mapping

So…what are the duplicate sets?

Identify duplicates using orientation + “unclipped” 5’ position

Pos	1	2	3	4	5	6	7	8	9
Ref	T	A	G	C	C	G	A	T	C
r1	T	A	G	C	C	G	A		
r2	T	A	G	C	C	G	A		
r3	T	A	–	C	CAG		A		
r4	T	A	G	C	C	H	H		
r5	T	A	G	C	C	G	A	T	C
r6	S	S	G	C	C	G	A		
r7			G	C	C	G	A		

Blue maps to forward strand
Orange maps to reverse strand
Grey bases are clipped

Underlined is the expected 5’ start of the read, given the mapping

So…what are the duplicate sets?
☞ **r1, r3, r5, r6** (start at position 1)

Identify duplicates using orientation + “unclipped” 5’ position

```
Pos 1 2 3 4 5 6 7 8 9
Ref T A G C C G A T C
 r1 T A G C C G A
 r2 T A G C C G A
 r3 T A – C CAG A
 r4 T A G C C H H
 r5 T A G C C G A T C
 r6 S S G C C G A
 r7     G C C G A
```

Blue maps to forward strand
Orange maps to reverse strand
Grey bases are clipped

Underlined is the expected 5’ start of the read, given the mapping

So…what are the duplicate sets?

☞ **r1, r3, r5, r6** (start at position 1)

☞ **r2, r4** (start at position 7)

Identify duplicates using orientation + “unclipped” 5’ position

Pos	1	2	3	4	5	6	7	8	9
Ref	T	A	G	C	C	G	A	T	C
r1	T	A	G	C	C	G	A		
r2	T	A	G	C	C	G	A		
r3	T	A	–	C	CAG		A		
r4	T	A	G	C	C	H	H		
r5	T	A	G	C	C	G	A	T	C
r6	S	S	G	C	C	G	A		
r7			G	C	C	G	A		

Blue maps to forward strand
Orange maps to reverse strand
Grey bases are clipped

Underlined is the expected 5’ start of the read, given the mapping

So…what are the duplicate sets?

☞ **r1, r3, r5, r6** (start at position 1)
☞ **r2, r4** (start at position 7)
☞ **r7** (starts at position 3)

PICARD BEST PRACTICES PIPELINE

Pipeline overview

Library Prep

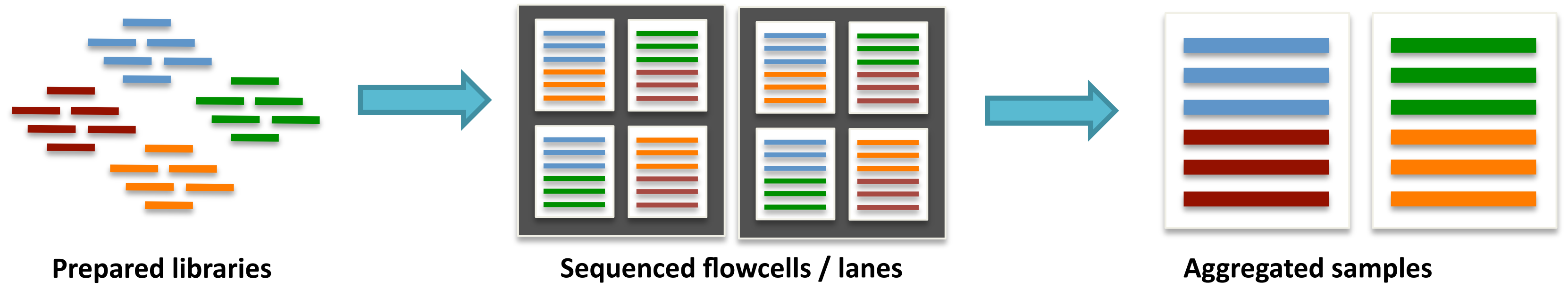
- Sample may have multiple libraries
- Libraries might be split across multiple flowcells

Flowcell-level processing

- Convert basecalls to SAM
- *Map with BWA*
- Sorting + cleaning
- Mark duplicates
- Collect QC metrics

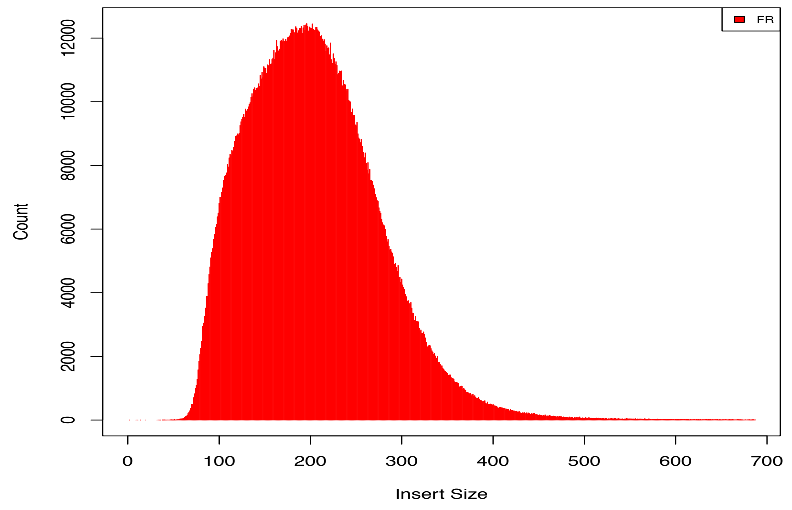
Aggregation-level processing

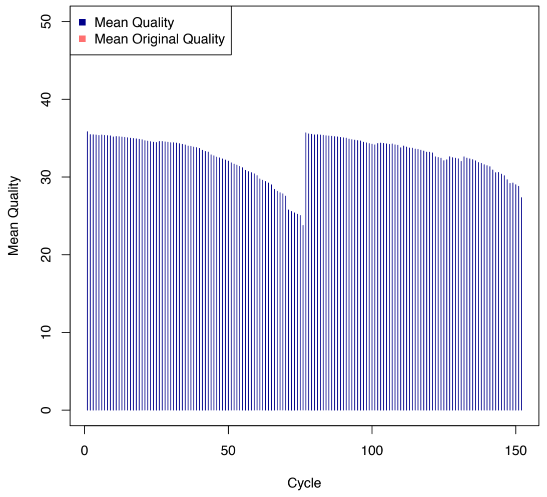
- Merge sample across flowcells
- Mark duplicates (again)
- *Indel realignment*
- *BQSR*
- Collect QC metrics

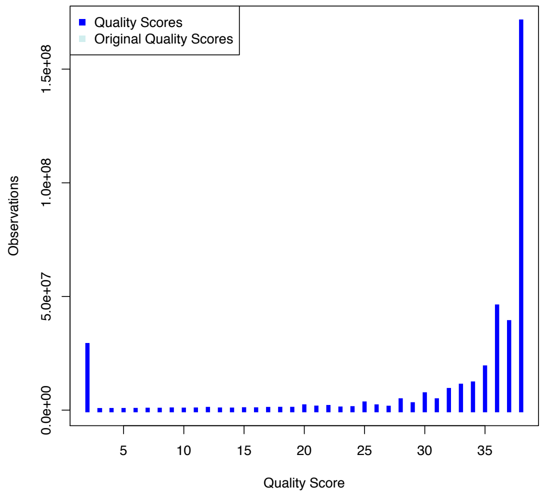


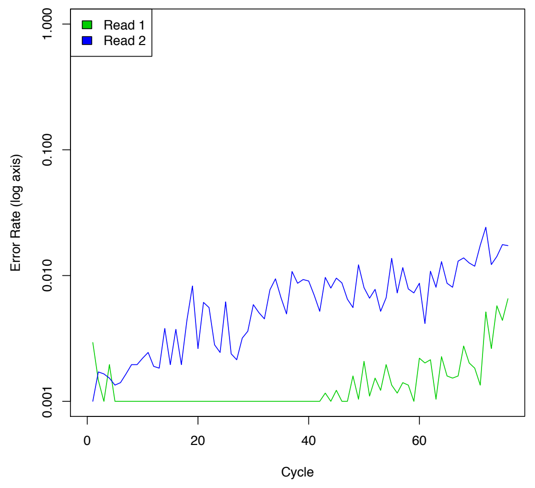
Collect QC metrics

- Base qualities
- Insert sizes
- Mapping qualities
- Coverage
- GC bias
- PF (purity filter) pass rate
- …and more!









Many more useful Picard tools

- If working with FASTQ data (non-Broad), you may have to annotate your SAMs manually
  - **SortSam**
  - **AddOrReplaceReadGroups**
- Validate with **ValidateSamFile**
- Fix inconsistencies with **RevertSam**
- Many other useful tools for manipulating SAMs (and VCFs, and FASTQs) in Picard!

How do I use Picard?

- broadinstitute.github.io/picard
- Fully open source, anyone can contribute
- Java-based command line interface, much like GATK

- Example 1: sort by genomic coordinate

```
java –jar picard.jar SortSam INPUT=unsorted.sam OUTPUT=sorted.sam \
      SORT_ORDER=coordinate
```

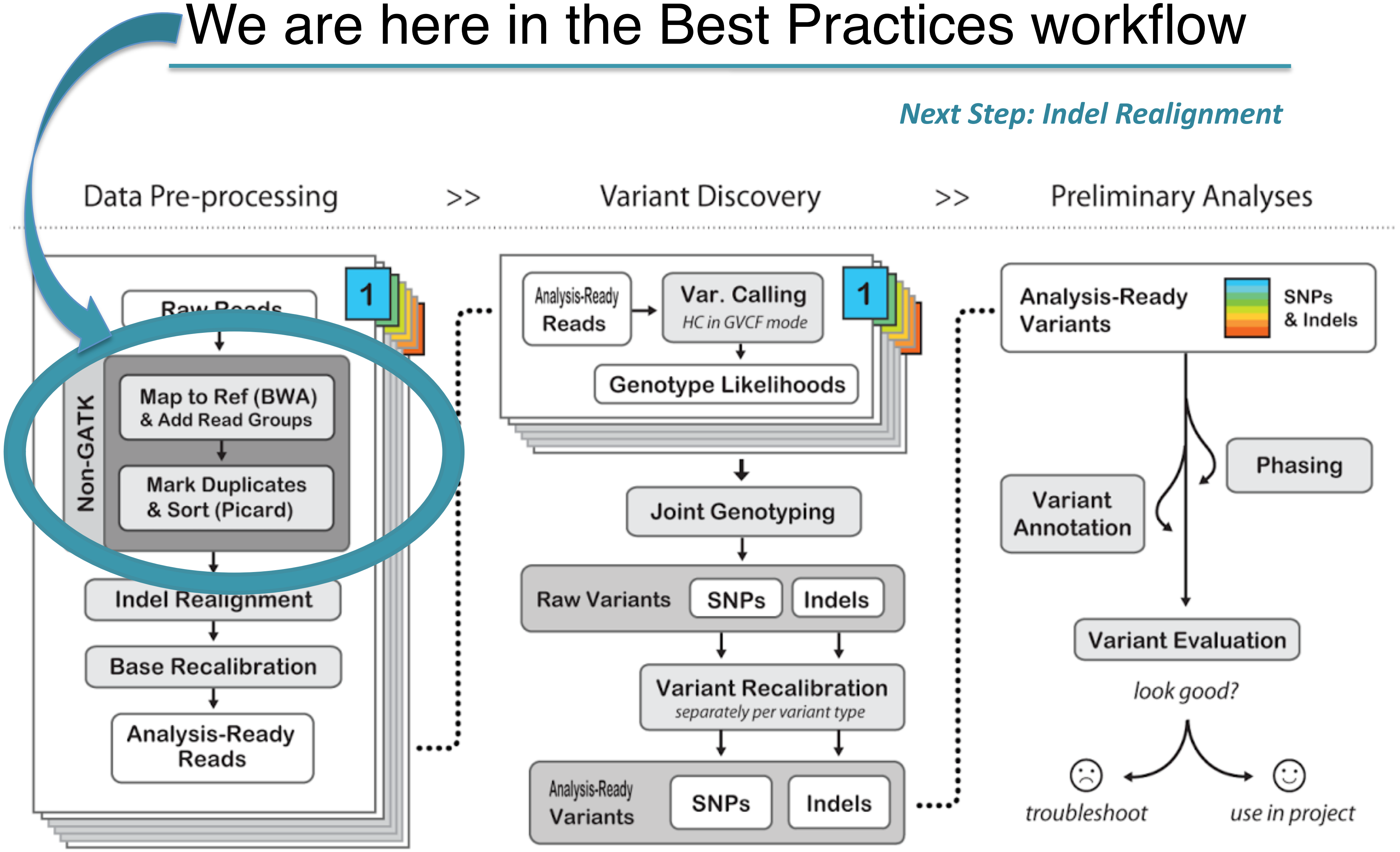
- Example 2: mark duplicates

```
java –jar picard.jar MarkDuplicatesWithMateCigar \
      INPUT=unmarked.sam OUTPUT=marked.sam
```

TO CONCLUDE

We are here in the Best Practices workflow

Next Step: Indel Realignment



Further reading

http://www.broadinstitute.org/gatk/guide/best-practices

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