Variant calling

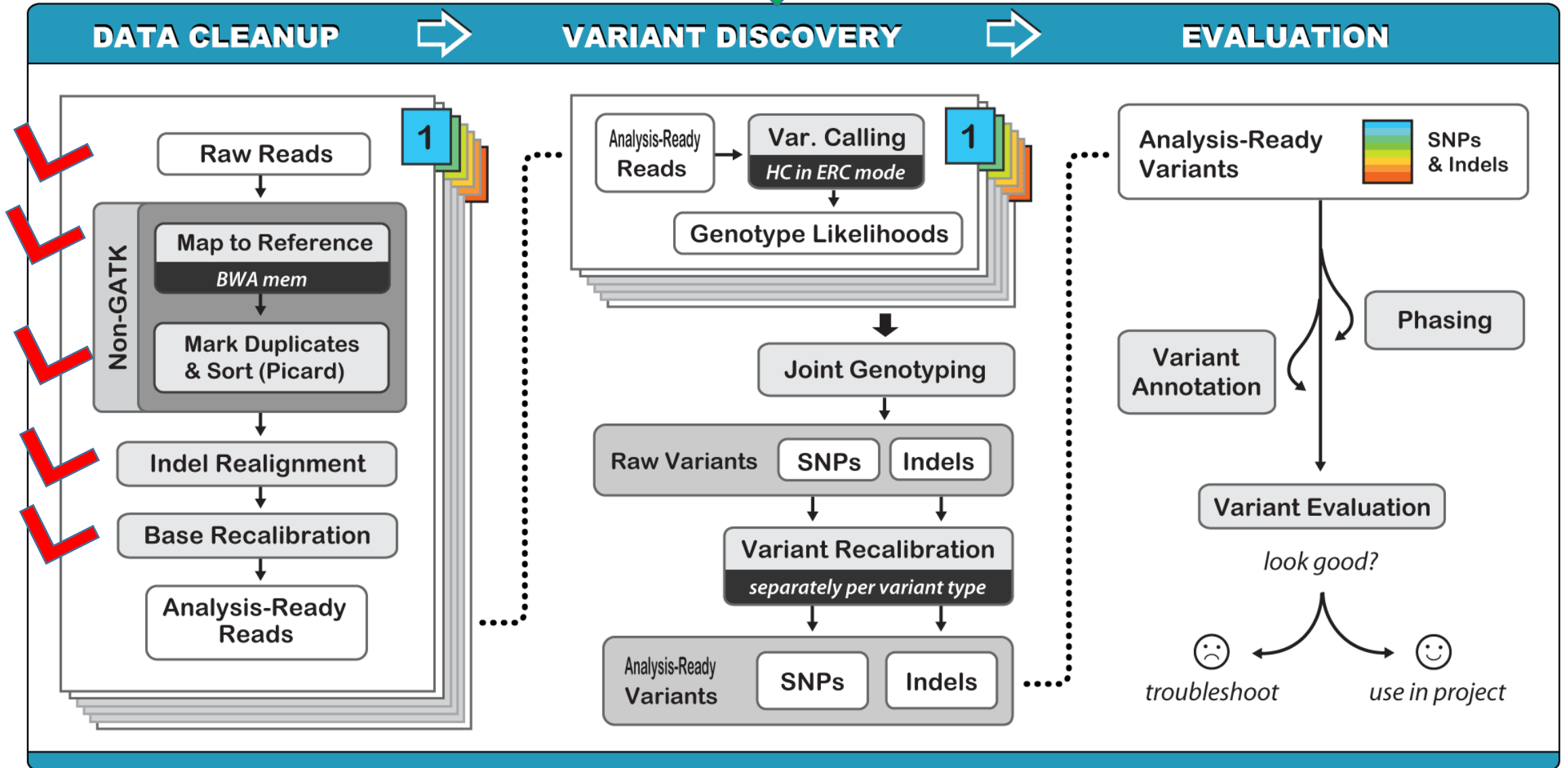Robert Bukowski, Qi Sun, Minghui Wang

Bioinformatics Facility

Institute of Biotechnology

Slides: http://cbsu.tc.cornell.edu/lab/doc/Variant_workshop_Part2.pdf

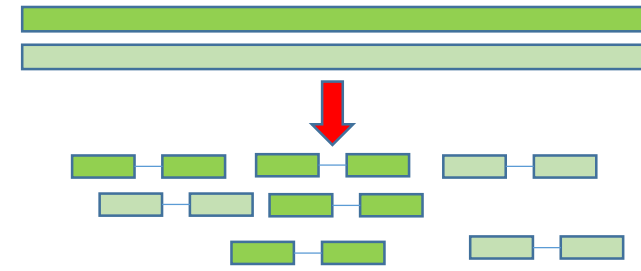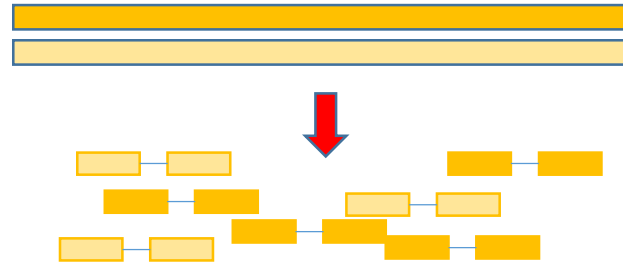Exercise instructions: http://cbsu.tc.cornell.edu/lab/doc/Variant_exercise2_2015.pdf

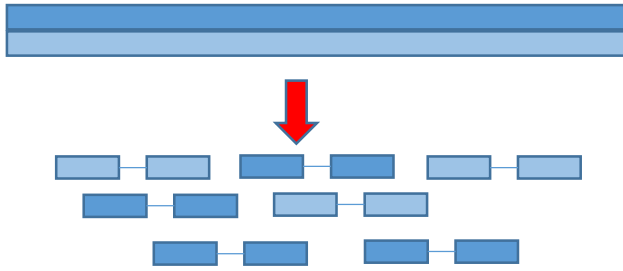# "Best Practices" for DNA-Seq variant calling
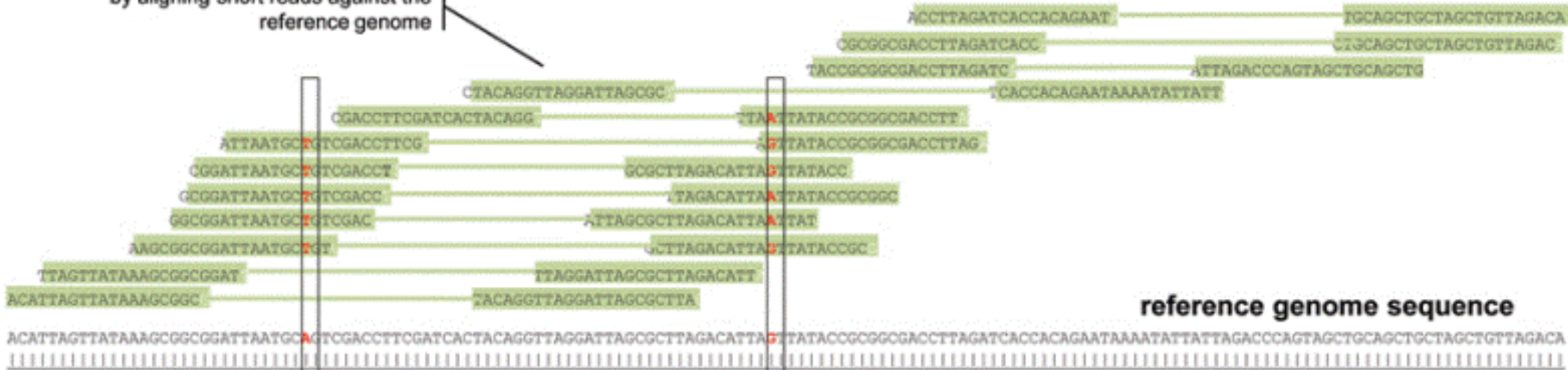
Individual 1   Individual 2   Individual 3

NGS

Align reads to a reference, do it for each sample

Genome of individual re-sequenced by aligning short reads against the reference genome

reference genome sequence

Individual is homozygous 'T' at this A/T polymorphism.

Individual is heterozygous at this G/A polymorphism.

Objective

For each site on the genome determine

- Whether or not there is **any variation** at this site (across all samples)
- If there is variation, **assign a genotype** (for diploids: pair of alleles) to each sample
- Report variant sites (positions, alleles, sample genotypes, …)

How it's done?

The old days:

Call variant base on thresholds (e.g., ratio of reference/non-reference bases)
Assign genotypes based on other thresholds

Now-days: probabilistic framework

- Calculate and report **probability** of a site being a variant (given read alignments)
- Calculate and report probabilities (or **likelihoods**) of various sample genotypes (given read alignments)
- Input: read **base quality scores**, preferably **recalibrated**

How to describe variants: Variant Call Format (VCF)

```
##fileformat=VCFv4.1
[HEADER LINES]
#CHROM    POS     ID    REF    ALT    QUAL     FILTER    INFO           FORMAT          ZW155                    ZW177
chr2R     2926    .     C      A      345.03   PASS      [ANNOTATIONS]  GT:AD:DP:GQ:PL  0/1:4,9:13:80:216,0,80   0/0:6,0:6:18:0,18,166
chr2R     9862    .     TA     T      180.73   .         [ANNOTATIONS]  GT:AD:DP:GQ:PL  1/1:0,5:5:15:97,15,0     1/1:0,4:4:12:80,12,0
chr2R     10834   .     A      ACTG   173.04   .         [ANNOTATIONS]  GT:AD:DP:GQ:PL  0/0:14,0:14:33:0,33,495  0/1:6,3:9:99:105,0,315
```

[HEADER LINES]: start with "##", describe all symbols found later on, e.g.,

```
##FORMAT=<ID=AD,Number=.,Type=Integer,Description="Allelic depths for the ref and alt alleles in the order listed">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Approximate read depth (reads with MQ=255 or with bad mates are filtered)">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
```

ID: some ID for the variant, if known (e.g., dbSNP)

REF, ALT: reference and alternative alleles (on forward strand of reference)

QUAL = -10*log(1-p), where p is the probability of variant being present given the read data

FILTER: whether the variant failed a filter (filters defined by the user or program processing the file)

How to describe variants: Variant Call Format (VCF)

```
[HEADER LINES]
#CHROM   POS    ID   REF    ALT     QUAL      FILTER     INFO              FORMAT                ZW155                      ZW177
chr2R    2926   .    C      A       345.03    PASS     [ANNOTATIONS]    GT:AD:DP:GQ:PL   0/1:4,9:13:80:216,0,80    0/0:6,0:6:18:0,18,166
chr2R    9862   .    TA     T       180.73     .       [ANNOTATIONS]    GT:AD:DP:GQ:PL   1/1:0,5:5:15:97,15,0      1/1:0,4:4:12:80,12,0
chr2R   10834   .    A     ACTG     173.04     .       [ANNOTATIONS]    GT:AD:DP:GQ:PL   0/0:14,0:14:33:0,33,495       ./.
```

GT (genotype):

      0/0   reference homozygote

      0/1   reference-alternative heterozygote

      1/1   alternative homozygote

      0/2, 1/2, 2/2, etc.   - possible if more than one alternative allele present

      ./.   missing data

AD: allele depths

DP: total depth (may be different from sum of AD depths, a the latter include only reads significantly supporting alleles)

PL: genotype likelihoods (phred-scaled), normalized to the best genotype, e.g.,

      PL(0/1) = -10*log[Prob(data|0/1) / Prob(data|best_genotype)]

GQ: genotype quality – this is just PL of the second-best genotype

How to describe variants: Variant Call Format (VCF)

```
[HEADER LINES]
#CHROM  POS    ID  REF   ALT   QUAL    FILTER    INFO           FORMAT            ZW155                      ZW177
chr2R   2926   .   C     A     345.03    PASS    [ANNOTATIONS]  GT:AD:DP:GQ:PL  0/1:4,9:13:80:216,0,80     0/0:6,0:6:18:0,18,166
chr2R   9862   .   TA    T     180.73      .     [ANNOTATIONS]  GT:AD:DP:GQ:PL  1/1:0,5:5:15:97,15,0       1/1:0,4:4:12:80,12,0
chr2R  10834   .   A    ACTG   173.04      .     [ANNOTATIONS]  GT:AD:DP:GQ:PL  0/0:14,0:14:33:0,33,495    0/1:6,3:9:99:105,0,315
```

[ANNOTATIONS]:   all kinds of quantities and flags that characterize the variant; supplied by the variant caller (different callers will do it differently)

Example:

AC=2;AF=0.333;AN=6;DP=16;FS=0.000;GQ_MEAN=16.00;GQ_STDDEV=10.54;MLEAC=2;MLEAF=0.333;MQ=25.00;MQ0=0;NCC=1;QD=22.51;SOR=3.611

All ANNOTATION parameters are defined in the **HEADER LINES** on to of the file

```
…
##INFO=<ID=AC,Number=A,Type=Integer,Description="Allele count in genotypes, for each ALT allele, in the same order as listed">
##INFO=<ID=AF,Number=A,Type=Float,Description="Allele Frequency, for each ALT allele, in the same order as listed">
##INFO=<ID=AN,Number=1,Type=Integer,Description="Total number of alleles in called genotypes">
##INFO=<ID=DP,Number=1,Type=Integer,Description="Approximate read depth; some reads may have been filtered">
##INFO=<ID=FS,Number=1,Type=Float,Description="Phred-scaled p-value using Fisher's exact test to detect strand bias">
##INFO=<ID=GQ_MEAN,Number=1,Type=Float,Description="Mean of all GQ values">
##INFO=<ID=MQ,Number=1,Type=Float,Description="RMS Mapping Quality">
##INFO=<ID=NCC,Number=1,Type=Integer,Description="Number of no-called samples">
##INFO=<ID=QD,Number=1,Type=Float,Description="Variant Confidence/Quality by Depth">
##INFO=<ID=SOR,Number=1,Type=Float,Description="Symmetric Odds Ratio of 2x2 contingency table to detect strand bias">
…
```

Sample-by-sample or joint (cohort-level) variant calling?

Genomes

Sample A

Sample B

Site 1   Site 2

If calling samples individually, Site 2 will be reported only for Sample B. Is Site 2 invariant in sample A, or is it just missing data (no read coverage)?

"Seeing" reads from multiple samples (mapped to a region of reference genome) allows for smarter decisions about which alleles are real and which are sequencing or alignment errors...

Reads from many samples

More confidence in variant calling

Multiple samples data allow calling a variant even if individual sample calls are of low quality

Error?

Joint calling is better

Two approaches to variant calling

Call SNPs and indels separately by considering **each variant locus independently**

Fast, but less accurate, especially for indels, needs indel realignment

Implemented in GATK as
UnifiedGenotyper

Call SNPs, indels together **from haplotypes assembled** *de novo* in regions of interest (i.e., of high variability)

More accurate, but sloooow....
Indel realignment step not needed?

Implemented in GATK as
HaplotypeCaller

FreeBayes

(not GATK)

Garrison E, Marth G. Haplotype-based variant detection from short-read sequencing. arXiv preprint arXiv:1207.3907 [q-bio.GN] 2012

SNP and Indel calling is a large-scale Bayesian modeling problem

Prior of the genotype

Likelihood of the genotype

=1

Reported as PL in our VCF example

Diploid assumption

Bayesian model

$$\Pr\{G|D\} = \frac{\Pr\{G\}\Pr\{D|G\}}{\Sigma_i \Pr\{G_i\}\Pr\{D|G_i\}}, \quad [\text{Bayes' rule}]$$

$$\Pr\{D|G\} = \prod_j \left(\frac{\Pr\{D_j|H_1\}}{2} + \frac{\Pr\{D_j|H_2\}}{2}\right) \quad \text{where } G = H_1 H_2$$

$$\Pr\{D|H\} \quad \text{is} \quad \text{the haploid likelihood function}$$

- Inference: what is the genotype G of each sample given read data D for each sample?
- Calculate via Bayes' rule the probability of each possible G
- Product expansion assumes reads are independent
- Relies on a likelihood function to estimate probability of sample data given proposed haplotype

Haplotype likelihood function for UnifiedGenotyper:

$$\Pr\{D_j|H\} = \Pr\{D_j|b\}, \quad [\text{single base pileup}]$$

$$\Pr\{D_j|b\} = \begin{cases} 1 - \epsilon_j & D_j = b, \\ \epsilon_j & \text{otherwise.} \end{cases}$$

From base quality score

Substitution-specifc rates (confusion matrix) may also be used here
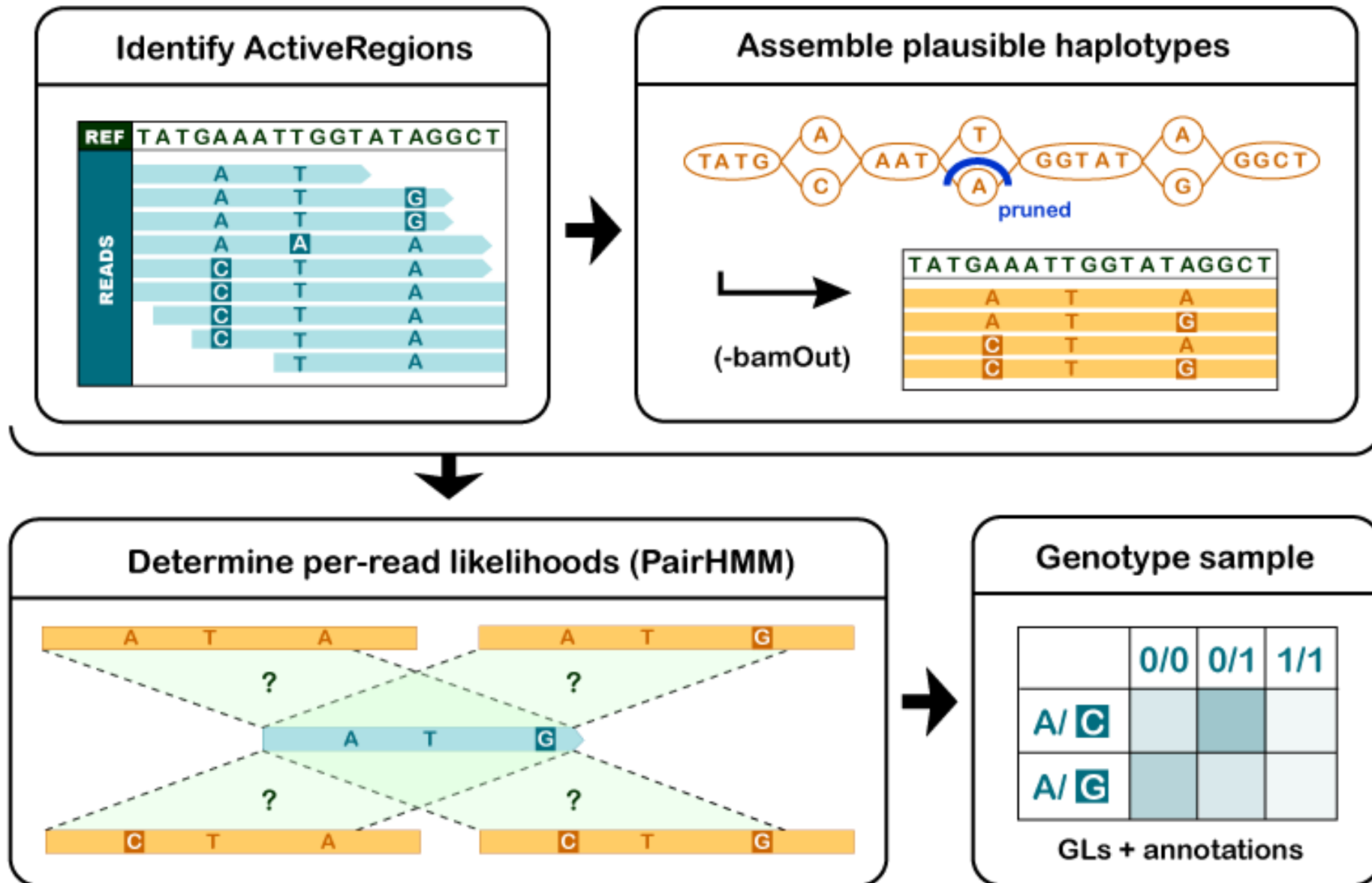
Multi-sample calling integrates per sample likelihoods to jointly estimate allele frequency of variation
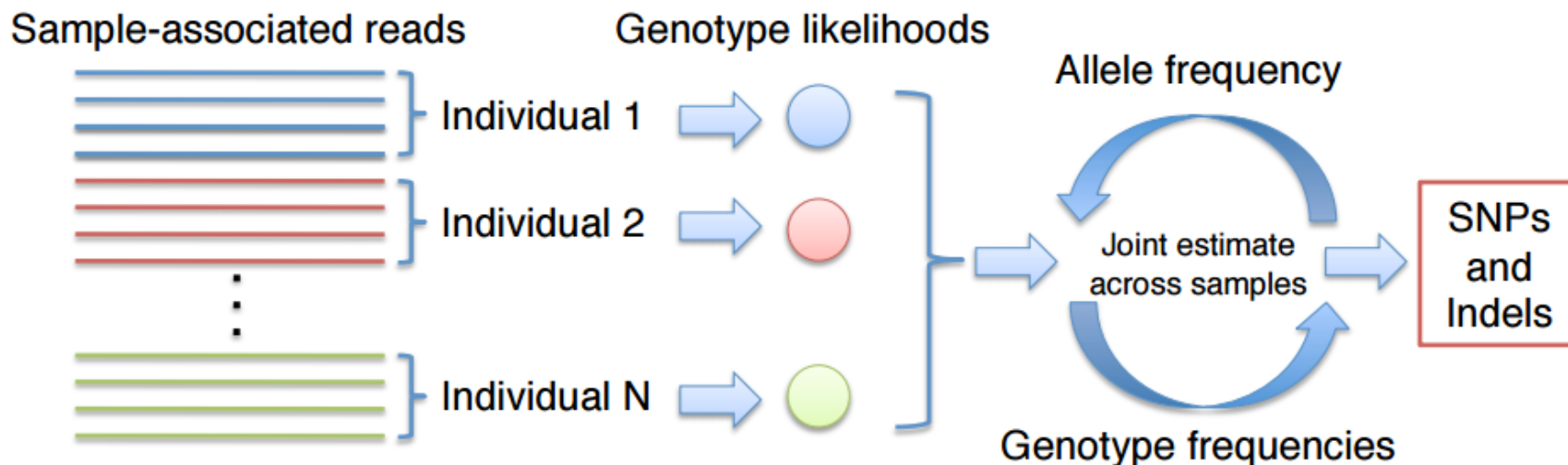
- Simultaneous estimation of:
  - Allele frequency (AF) spectrum Pr{AF = i | D}
  - The probability that a variant exists Pr{AF > 0 | D}
  - Assignment of genotypes to each sample

Reported in **QUAL** field of VCF

Excellent mathematical treatment at http://lh3lh3.users.sourceforge.net/download/samtools.pdf

HaplotypeCaller: what does it do?

P{D|H} determined from scores of reads alignments to haplotypes (based on base qualities)

Haplotype caller: what does it do?

1. **Define active regions**
The program determines which regions of the genome it needs to operate on, based on the presence of significant evidence for variation.

2. **Determine haplotypes by re-assembly of the active region**
For each ActiveRegion, the program builds a De Bruijn-like graph to reassemble the ActiveRegion, and identifies what are the possible haplotypes present in the data. The program then realigns each haplotype against the reference haplotype using the Smith-Waterman algorithm in order to identify potentially variant sites.

3. **Determine likelihoods of the haplotypes given the read data**
For each ActiveRegion, the program performs a pairwise alignment of each read against each haplotype using the PairHMM algorithm. This produces a matrix of likelihoods of haplotypes given the read data. These likelihoods are then marginalized to obtain the likelihoods of alleles for each potentially variant site given the read data.

4. **Assign sample genotypes**
For each potentially variant site, the program applies Bayes' rule, using the likelihoods of alleles given the read data to calculate the likelihoods of each genotype per sample given the read data observed for that sample. The most likely genotype is then assigned to the sample.

Artifact SNPs and small indels caused by large indel is only recovered by local assembly
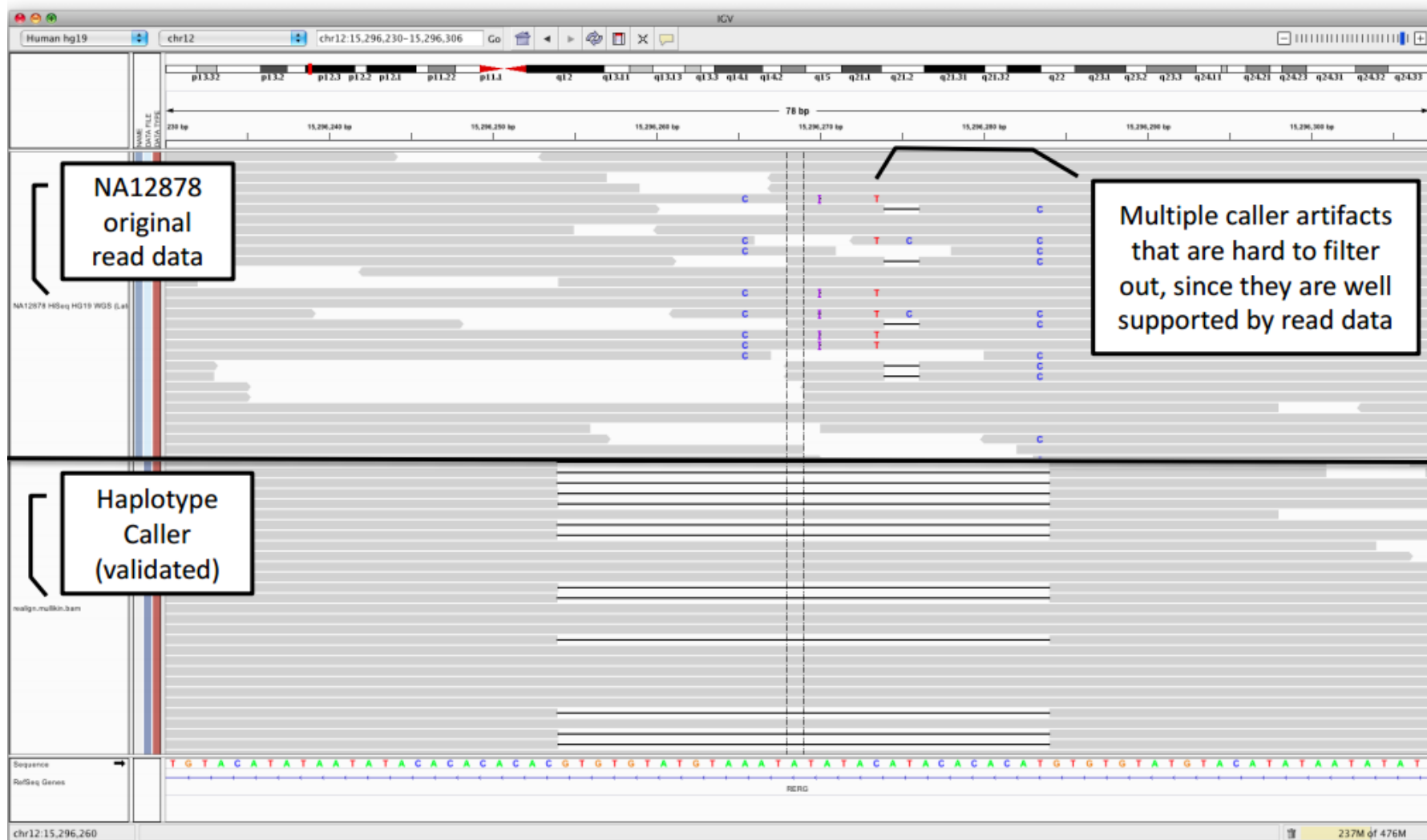
FreeBayes: haplotype-based an alternative to GATK
Erik Garrison et al., https://github.com/ekg/freebayes

Local realignment around indels not needed.
- It is accomplished internally, discordant alignments are dramatically reduced through the direct detection of <u>haplotypes</u>.
- True also for GATK's **HaplotypeCaller**

Base quality recalibration not needed
- Sequencing platform errors tend to cluster (e.g. at the ends of reads), and generate unique, non-repeating haplotypes at a given locus, which can be discarded
- Should be true also for GATK's **HaplotypeCaller**

**Reported variant quality more reliable**
- Better (than GATK's) Bayesian model, directly incorporating a number of metrics, such as read placement bias and allele balance
- No variant quality recalibration of complex variant filtering needed

In our tests – order of magnitude faster than GATK HaplotypeCaller (+ savings on BAM preparation)!
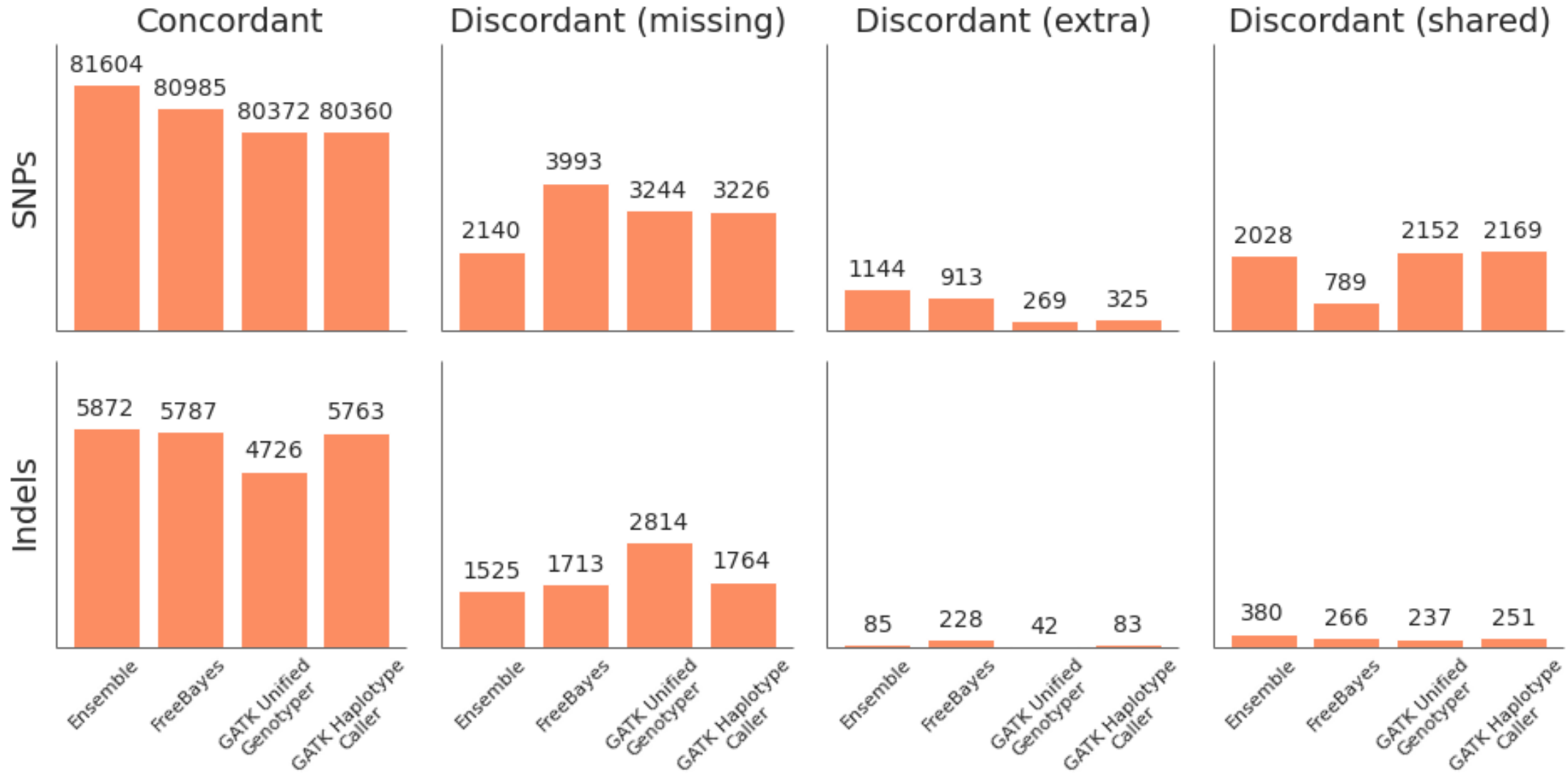
Still suffers from "N+1" problem

Comparison of GATK and FreeBayes (how many known NA12878 SNPs/indels are called correctly/incorrectly)

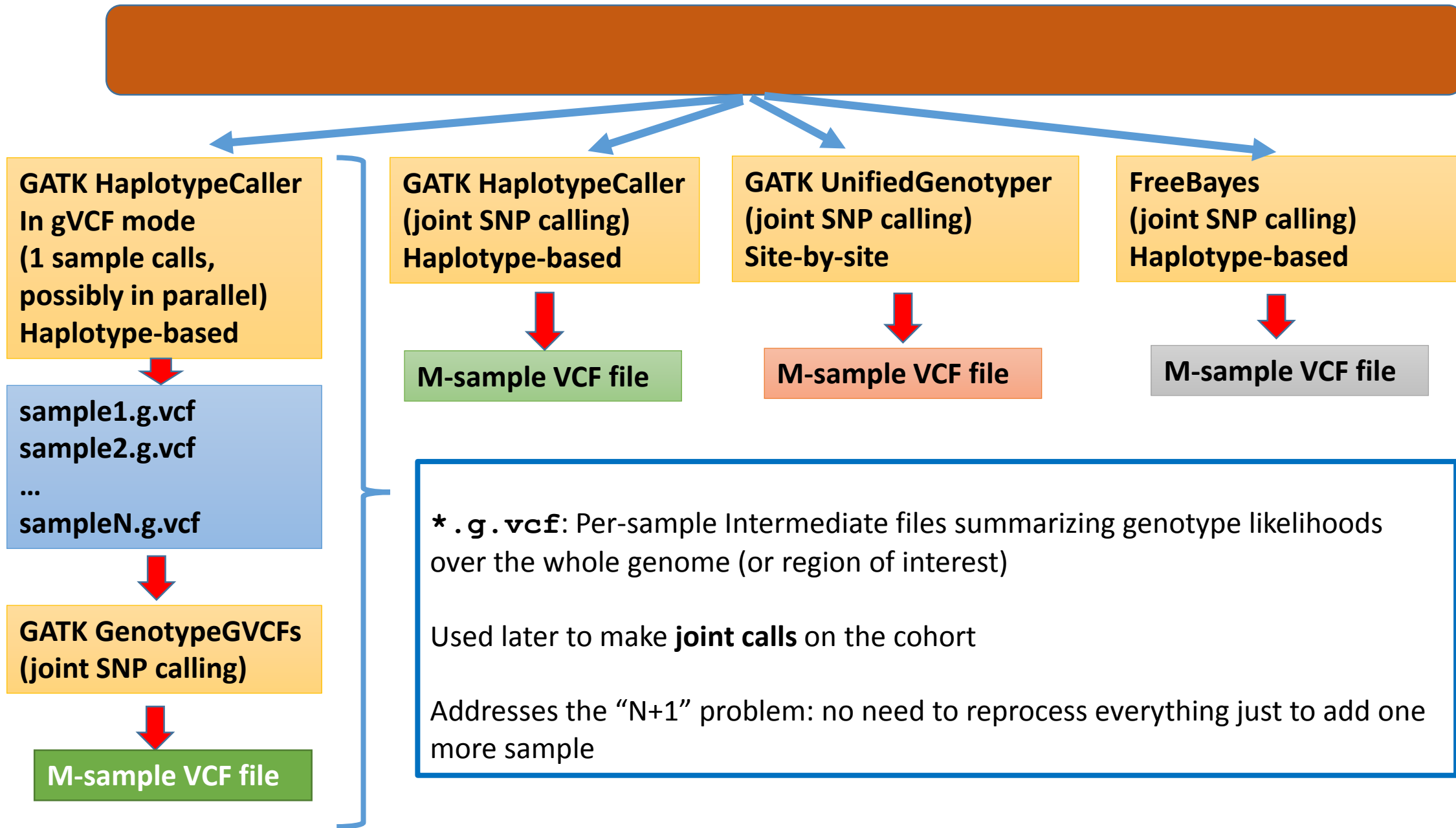GATK best-practice BAM preparation (recalibration, realignment)
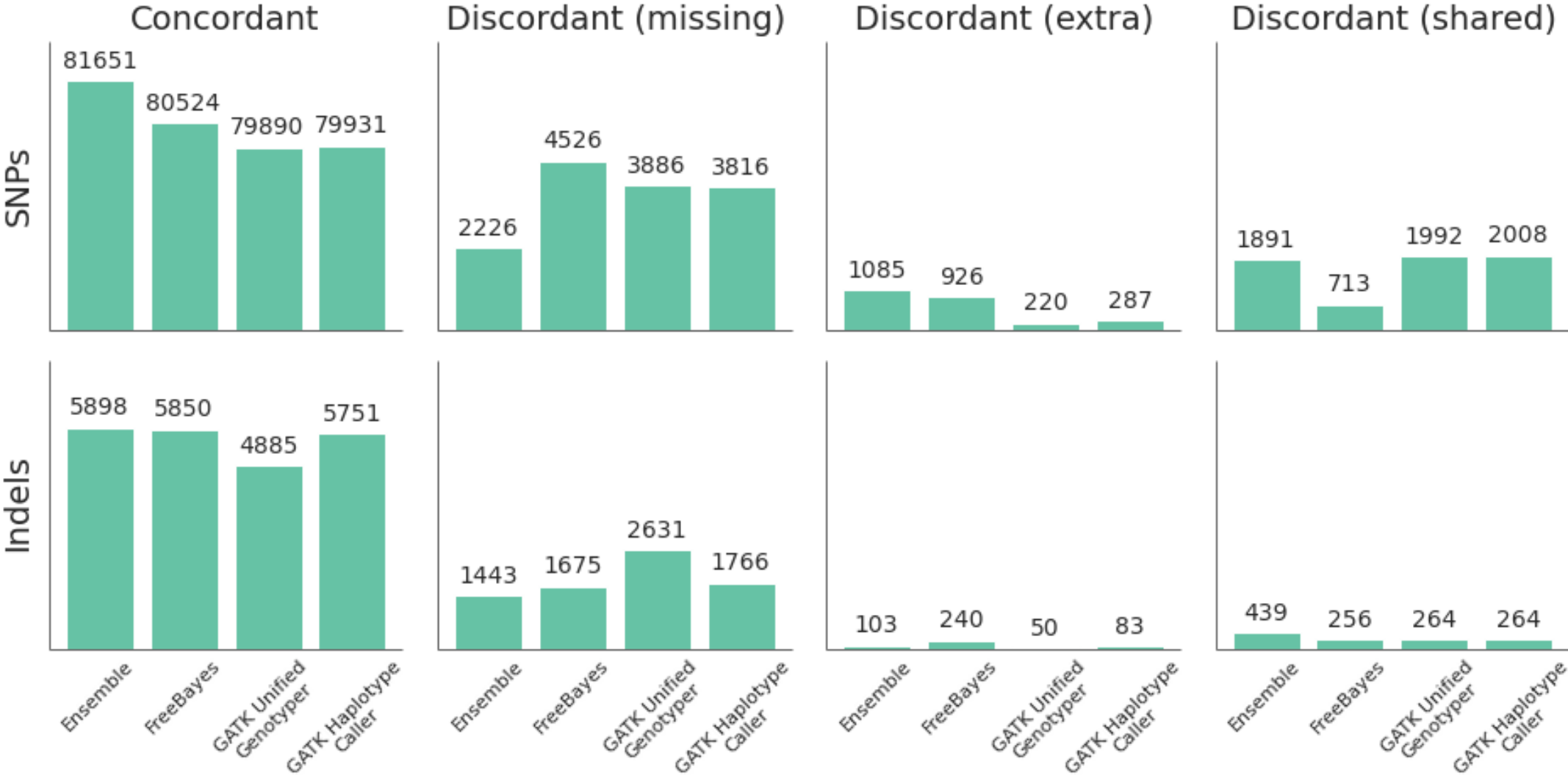
Comparison of GATK and FreeBayes (how many known NA12878 SNPs/indels are called correctly/incorrectly)

Minimal BAM preparation (samtools de-duplication only)

From sample alignments (BAM files) files to raw variants (VCF file)

Running HaplotypeCaller in gVCF mode

Run for each sample (on a multi-CPU machine, **run a few simultaneously**)

<div style="border:1px solid #000; background:#d9d9d9; padding:8px;">

```
java -jar GenomeAnalysisTK.jar \
    -T HaplotypeCaller \
    -R genome.fa \
    -I sample1.sorted.dedup.realigned.fixmate.recal.bam  \
    --emitRefConfidence GVCF \
    --variant_index_type LINEAR \
    --variant_index_parameter 128000 \
    -o sample1.g.vcf
```
</div>

Slow

GATK HaplotypeCaller In gVCF mode (1 sample calls, possibly in parallel) Haplotype-based

⬇

sample1.g.vcf sample2.g.vcf ... sampleN.g.vcf

⬇

Run once after all *.g.vcf files are obtained

GATK GenotypeGVCFs (joint SNP calling)

<div style="border:1px solid #000; background:#d9d9d9; padding:8px;">

```
java –Xmx2g -jar GenomeAnalysisTK.jar \
    -T GenotypeGVCFs \
    -R genome.fa \
    --variant sample1.g.vcf \
    --variant sample2.g.vcf \
    --variant sample3.g.vcf \
    --variant sample4.g.vcf \
    -stand_call_conf 30 \
    -stand_emit_conf 10 \
    -o 4samples.vcf
```
</div>

Fast

⬇

M-sample VCF file

Running HaplotypeCaller (variant-only mode)

GATK HaplotypeCaller (joint SNP calling) Haplotype-based

M-sample VCF file

```
java –jar GenomeAnalysisTK.jar \
    -T HaplotypeCaller \
    -R genome.fa \
    -I sample1.sorted.dedup.realigned.fixmate.recal.bam  \
    -I sample2.sorted.dedup.realigned.fixmate.recal.bam  \
    -I sample3.sorted.dedup.realigned.fixmate.recal.bam  \
    -I sample4.sorted.dedup.realigned.fixmate.recal.bam  \
    -L chr2R \
    -stand_call_conf 30 \
    -stand_emit_conf 10 \
    -o 4samples_joint_call.chr2R.vcf
```

May be parallelized by genome region (using –L option)
i.e., different regions run on different processors

Note:

Haplotype assembly uses reads from all samples (rather than one at a time), and so…

…resulting VCF file will not be exactly equivalent to that obtained from gVCF mode runs followed by GenotypeGVCFs

Running UnifiedGenotyper

GATK UnifiedGenotyper (joint SNP calling) Site-by-site

M-sample VCF file

```
java -Djava.io.tmpdir=$TMP -jar GenomeAnalysisTK.jar \
    -T UnifiedGenotyper \
    -R genome.fa \
    -I sample1.sorted.dedup.realigned.fixmate.recal.bam  \
    -I sample2.sorted.dedup.realigned.fixmate.recal.bam  \
    -I sample3.sorted.dedup.realigned.fixmate.recal.bam  \
    -I sample4.sorted.dedup.realigned.fixmate.recal.bam  \
    -L chr2R \
    -stand_call_conf 30 \
    -stand_emit_conf 10 \
    -o 4samples.UG.chr2R.vcf
```

May be parallelized by genome region (using –L option)
i.e., different regions run on different processors

Broad recommends running HaplotypeCaller-based pipeline instead if this

However, still recommended for
Pooled sample cases
High ploidity cases

Options to pay attention to in HaplotypeCaller, GenotypeVCFs, and UnifiedGenotyper

`-stand_emit_conf [number]`

Variants with quality score (QUAL) less than `[number]` will not be written to VCF file. Good to set this low – better have too many raw variants than too few. Can always filter VCF file later. Default 30.

`-stand_call_conf [number]`

Variants with QUAL< `[number]` will be marked as **LowQual** in FILTER field of VCF. Default: 30

`-dcov [int]`

Read depth at any locus will be capped at `[number]`; the goal is to provide even distribution of read start positions while removing excess coverage. For truly unbiased down-sampling, use `-dfrac`
Defaults are usually high (250) – can be reduced to speed things up.

What to do with a freshly obtained set of called variants?

Useful tool: VariantFiltration – hard filtering on various criteria

Example:

```
java -jar GenomeAnalysisTK.jar  \
-T VariantFiltration \
-R genome.fa \
-filter "MQ0 >= 4 && ((MQ0 / (1.0 * DP)) > 0.1)"  \
-filter "FS>=10.0"    \
-filter "AN>=4" \
-filter "DP>100 || DP<4" \
-filterName HARD_TO_VALIDATE  \
-filterName SNPSBFilter \
-filterName SNPNalleleFilter  \
-filterName SNPDPFilter  \
-cluster 3 \
-window 10  \
--variant chr2R.4samples.vcf \
-o chr2R.4samples.filtered.vcf
```

Filtering options for SNPs may be different than for indels (see exercise)

Whenever any of the "-filter" conditions satisfied, the corresponding "-filterName" will be added to the FILTER field in VCF.

Commonly used filtering parameters

DP
Total depth of read coverage at the site (shouldn't be too low)

MQ0
Number of zero mapping quality reads spanning the site (should be low)

MQ
RMS mapping quality of reads spanning the site

FS
P-value (phred-scaled) of the strand bias contingency table (should be low)

QD
QUAL/(depth of non-reference reads) – should be large (e.g, >2)
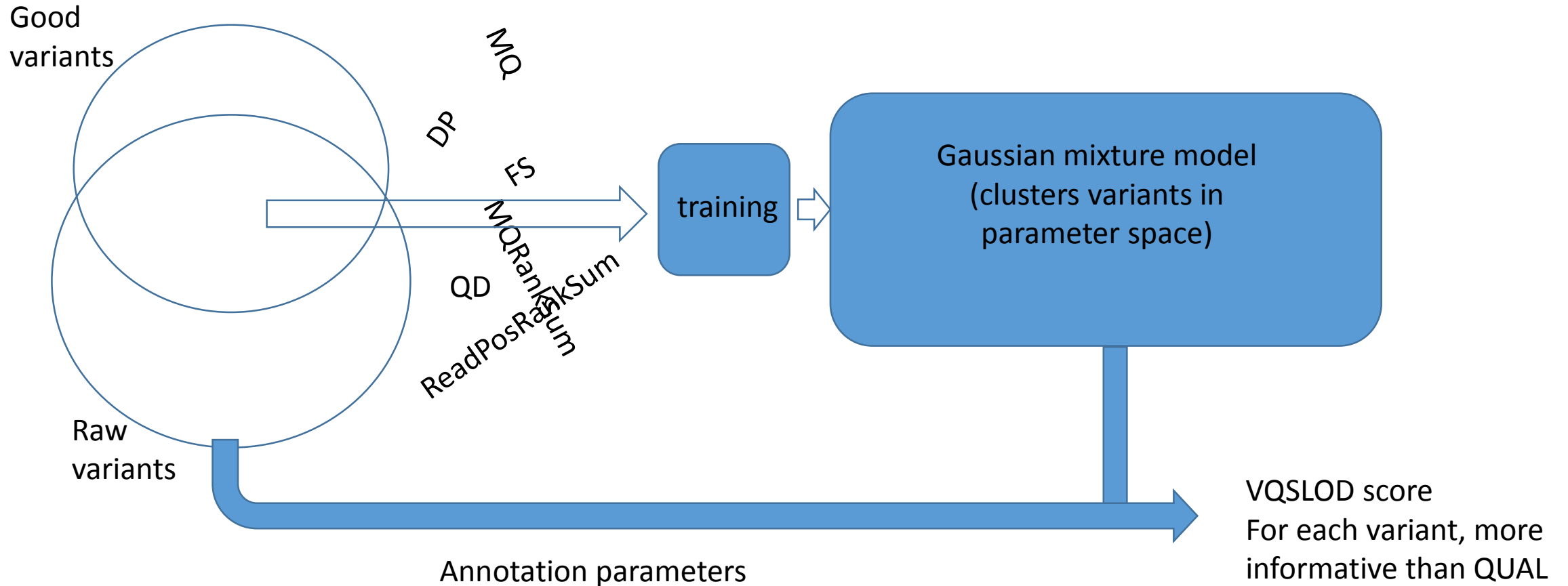
ReadPosRankSum
Parameter showing how close the variant site is to ends of reads (typically more positive for good variants) – available only for heterozygous sites
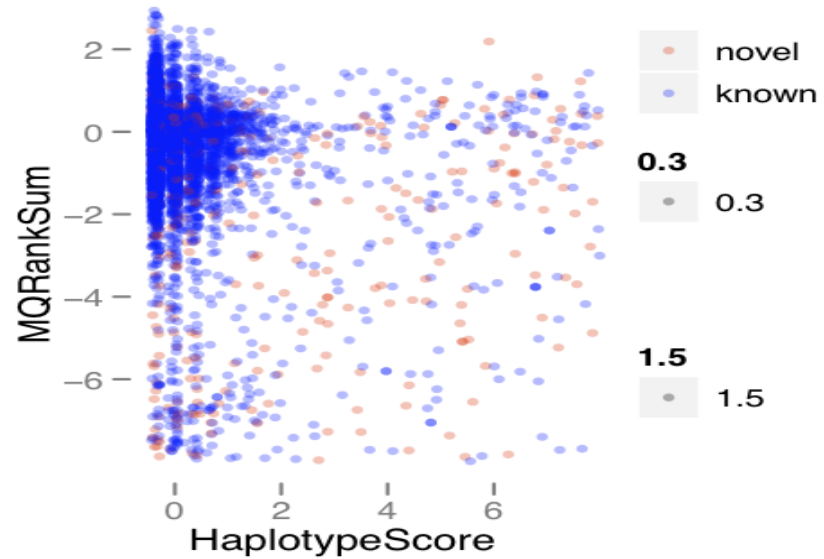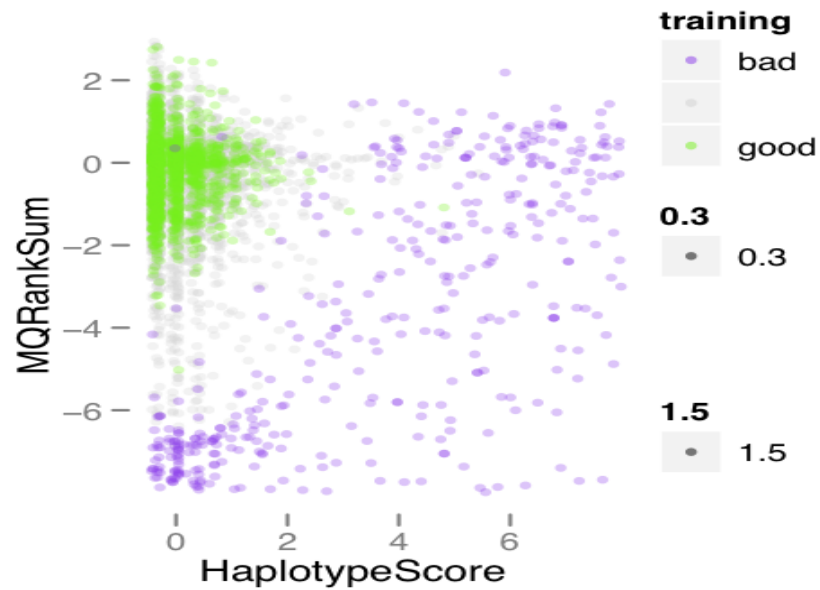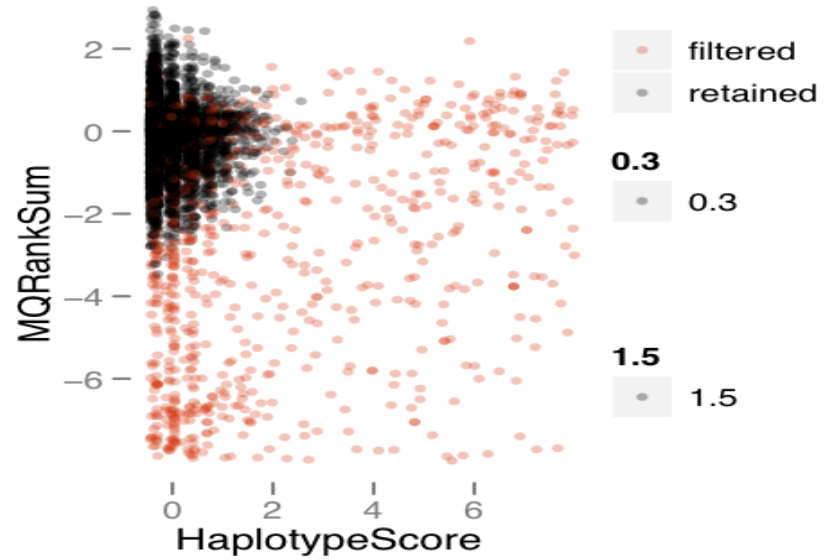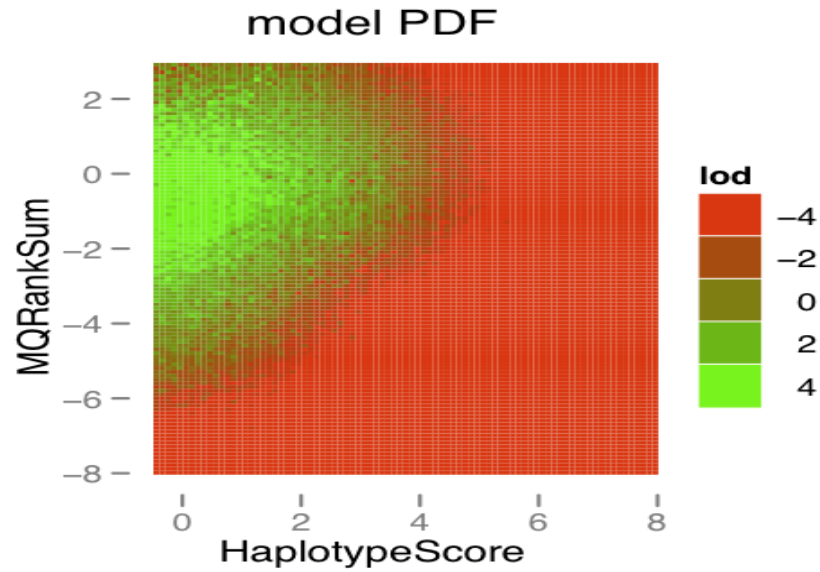
MQRankSum
Parameter comparing mapping qualities of reads carrying an alternative allele to reference reads – available only for heterozygous sites (typically more positive for good variants).

Variant Quality Score Recalibration (VSQR)

Recommended instead of hard (threshold-based) filtering when a set of true, reliable variants is available.

Good variants

MQ

DP

FS

MQRankSum

QD

ReadPosRankSum

training

Gaussian mixture model (clusters variants in parameter space)

Raw variants

Annotation parameters

VQSLOD score
For each variant, more informative than QUAL

2D cross-section though cluster of variants in multi-D parameters space

Useful tool: VariantEval – summary stats and comparison of callsets

```
java -Xmx2g -jar GenomeAnalysisTK.jar \
    -R genome.fa \
    -T VariantEval \
    -o file1.file2.comp.gatkreport \
    --eval:set1 file1.vcf  \
    --comp file2.vcf
```

Will summarize various properties of variants in file1.vcf
- Classes of variants
- Indel characteristics
- Ti/Tv
- Multi-allelic variants
- ….

Will compare to variants in file2.vcf
- Common variants and extra variants in file1.vcf (compared to file2.vcf)
- Concordance rate

Other VCF analysis and manipulation package: vcftools

vcftools (A. Auton, A. Amrcketta, http://vcftools.sourceforge.net/)

Obtain basis VCF statistics (number of samples and variant sites):

Extract subset of variants (chromosome chr2R, between positions 1M and 2M) and write tem a new VCF file

Get allele frequencies for all variants and write them to a file

Compare two VCF files (will print out various kinds of compare info in files `hc.ug.compare.*`):

Vcftools can also compute
- LD statistics
- Fst between populations

Word of caution

All call optimization effort in GATK directed towards detection and removal of sequencing errors and small alignment errors

Reference genome assumed to be adequate (similar to those pf re-sequenced individuals), i.e., reads assumed to be decently mapped to right locations possibly with small alignment ambiguities

Elaborate GATK pipeline will not help in case of massive misalignments (reads mapping to completely wrong locations) resulting from large diversity

What to do then?

Filter raw set of variants (most of them wrong) based on data for a large population (if you have one)

       Identity by Descent (IBD):   exploit local identity within pairs of samples
       local Linkage Disequilibrium (LD):   true variant should be in LD with nearby ones