A scalable and efficient pipeline for detecting and genotyping SNPs from next-generation sequencing data

June 22nd, 2011

Hyun Min Kang

Sequence Group Meeting

Outline

- Overview of Michigan SNP calling pipeline
- How to use the pipeline in practice
- How our pipeline performs in practice
- Some details on the underlying methods
- Current status and future directions

SNP calling pipelines at Michigan : A brief history

- Yun Li's ***thunder*** pipeline (2009.08-)
  - Major basis of current Michigan SNP calling pipeline
  - Used in 1000 Genomes Pilot SNP calls contributed by U of Michigan
  - http://genome.sph.umich.edu/wiki/Thunder

SNP calling pipelines at Michigan : A brief history

- Yun Li's ***thunder*** pipeline (2009.08-)
  - Major basis of current Michigan SNP calling pipeline
  - Used in 1000 Genomes Pilot SNP calls contributed by U of Michigan
  - http://genome.sph.umich.edu/wiki/Thunder
- Carlo Sidore's ***seqPipeline*** (2010.05-)
  - Better-automated & configurable version of Yun's pipeline
  - Implemented in python
  - Compatible with MOSIX and Sun Grid Engines
  - Used in 1000 Genomes May 2010 calls, GSK calls, and Sardinian SNP calls
  - Actively in sync with Hyun's ***umake*** pipeline
  - http://genome.sph.umich.edu/wiki/Variant_Call_Pipeline

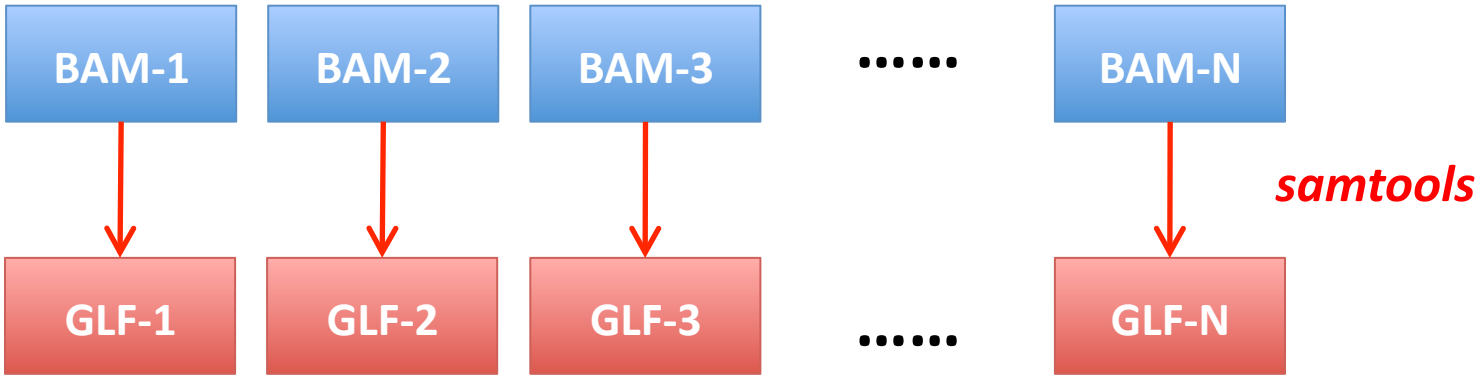
SNP calling pipelines at Michigan : A brief history

- Yun Li's ***thunder*** pipeline (2009.08-)
  - Major basis of current Michigan SNP calling pipeline
  - Used in 1000 Genomes Pilot SNP calls contributed by U of Michigan
  - http://genome.sph.umich.edu/wiki/Thunder
- Carlo Sidore's ***seqPipeline*** (2010.05-)
  - Better-automated & configurable version of Yun's pipeline
  - Implemented in python
  - Compatible with MOSIX and Sun Grid Engines
  - Used in 1000 Genomes May 2010 calls, GSK calls, and Sardinian SNP calls
  - Actively in sync with Hyun's ***umake*** pipeline
  - http://genome.sph.umich.edu/wiki/Variant_Call_Pipeline
- Hyun's ***umake*** pipeline (2010.08-)
  - Makefile-based automation of previous pipelines
  - Enhanced filtering scheme against potential false positive calls
  - Applied in recent call sets in 1000 Genomes, GoT2D, and Psoriasis RNA-seq
    - Goo's modified version (for exome) was applied in ESP calls (MS2500)
  - http://genome.sph.umich.edu/wiki/Umake (TBA)

Starting Point : A set of BAM files

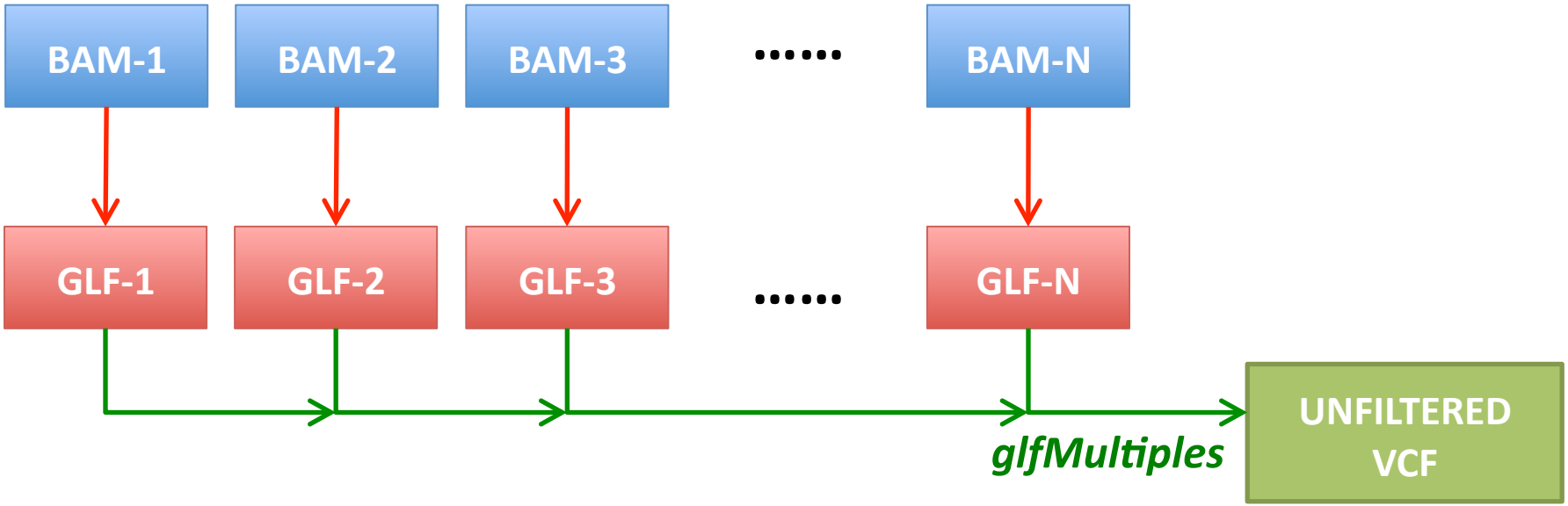
- Typically, one BAM file for each individual sample
  - Can handle multiple BAM files over multiple sequencing platforms
- Before SNP calling, BAM files are expected to be
  - Aligned from sequence reads (FASTQ –> BAM)
  - Merged across multiple lanes (read groups)
  - Duplicate-marked.
  - Base-quality recalibrated
  - Indel-realigned (optional)
    - Against known indels only (e.g. low-pass genome)
    - Against suspicious indels too (e.g. deep exome)

Computing genotype likelihoods from aligned reads



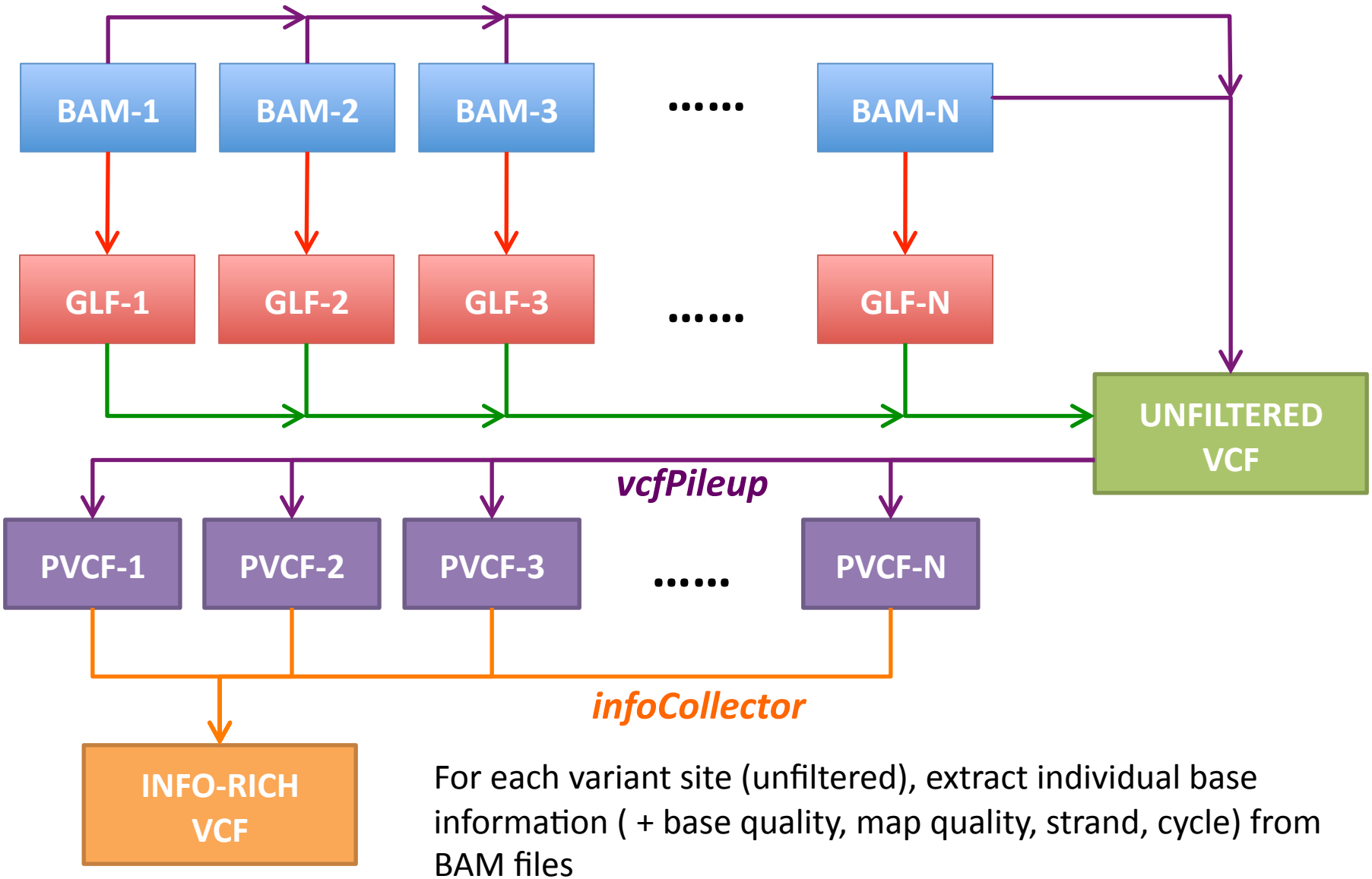
- Pileup using *samtools* under MAQ model
- For each genomic position
  - compute likelihood of observed bases
  - given all ${}_4H_2 = 10$ possible genotypes
    - A/A, A/C, A/G, A/T, C/C, C/G, C/T, G/G, G/T, and T/T
  - Per-base alignment quality (BAQ) adjustment is applied by default
- Store the pre-computed genotype likelihoods in GLF format
- Typically, GLF are generated for each genomic chunk (e.g. 5Mb) to facilitate parallelization

Calling variants from genotype likelihoods



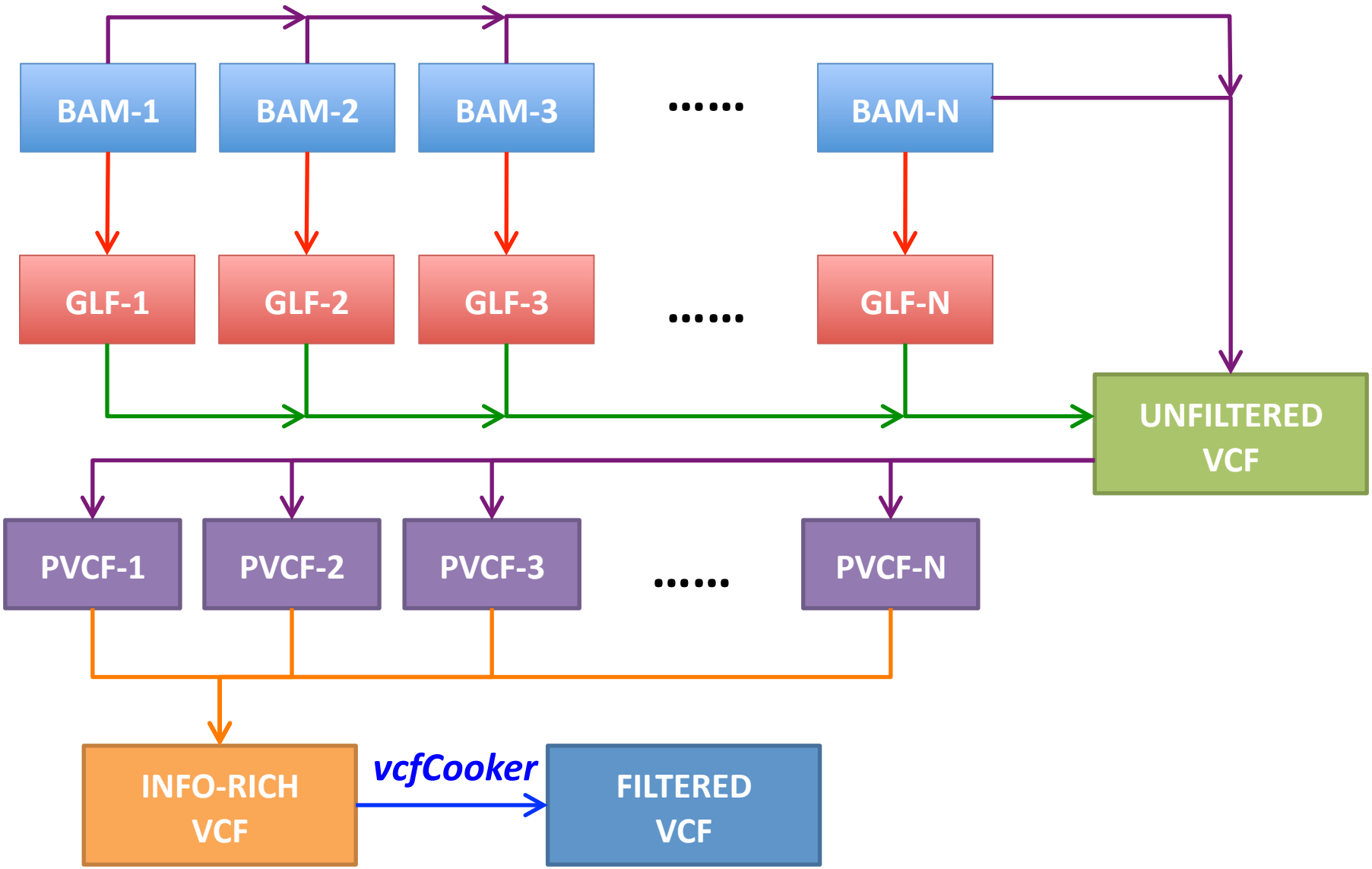
- Multi-sample SNP calling using *glfMultiples*
- A naïve Bayes model computing Pr(AlleleCount > 0 | Observed Data)
- Population-based prior
  - High power to detect shared variants among individuals at low-coverage
- Simple site frequency spectrum based on coalescent theory
  - Possible to incorporate alternative site frequency spectra

Collecting individual read information for filtering

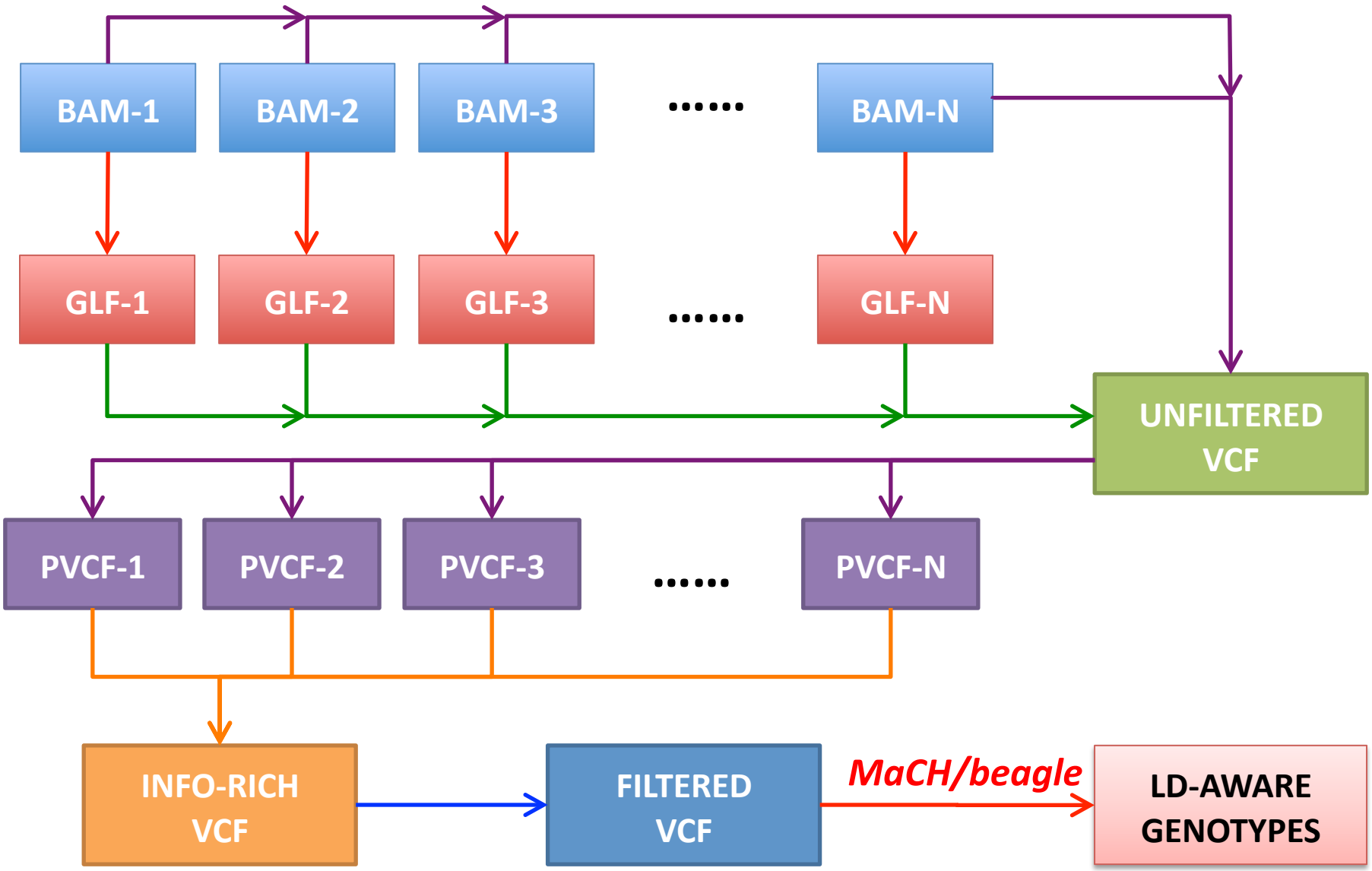


For each variant site (unfiltered), extract individual base information (+ base quality, map quality, strand, cycle) from BAM files

Filtering potential false positive SNPs



LD-aware genotype refinement



Umake : A scalable and efficient SNP calling pipeline

- Incremental framework for SNP calling
  - Pre-computed GLFs can be reused when calling across larger set.
  - No need to redundantly iterate BAM again
    - In current version, BAM is scanned once more in filtering step
    - We plan to modify current version to avoid such redundancy
- Scalability and efficiency
  - SNP calling time is linear to the number of individuals
  - Majority of disk I/O is sequential.
  - Practically scalable to thousands of genomes.
- Makefile-based framework for dependency management
  - Dependency between jobs are managed using Makefile
  - Easy parallelization using built-in option in GNU *make*
  - Convenient recovery from previous failure
    - Simply running *make* again will automatically resume from where it stopped
  - Recommended framework for other automated pipelines

How to use *umake* pipeline : Input data

1. A set of BAM files
   - Separated by individual
   - An individual may have multiple BAMs (e.g. different platforms)
   - Expected to be merged, dedupped, and recalibrated
2. Index of BAM files
   - A single text file mapping individuals to BAMs

     HG00096 ALL,EUR HG00096.ILLUMINA.GBR.bam
     NA07346 ALL,EUR NA07346.ILLUMINA.CEU.bam NA07346.LS454.CEU.bam
     NA20339 ALL,AFR NA20339.SOLID.ASW.bam
3. Configuration file containing
   - Location of software binaries and resources.
   - Series of steps to run (e.g. full pipeline vs pileup only)
   - Command line arguments of the software components such as filtering options, MOSIX nodes to use, or unit for genome chunks.

How to run *umake* pipeline

1. Create Makefile based on the configuration file

   umake.pl --conf [config-file.conf]

   [out-prefix].Makefile will be generated as a result

2. Run GNU make to kickstart SNP calling

   make –f [out-prefix].Makefile –j [# of parallel jobs] >& [out-prefix].log

3. If the job has failed for certain reason, debug the problem and repeat step 2 to resume from where it stopped.
   - Add –B option if you want to start over from a particular step.

4. Check whether filtered VCF file is generated to determine completion
   - At each step, [expected-output-file].OK is generated to confirm that the step has finished successfully.

Genotype refinement using *umake* pipeline

1. Run beagle to get initial genotypes

   umake-beagle.pl --conf [config-file.conf]

   make –f [out-prefix].beagle.Makefile –j [# of parallele jobs]

2. Run thunderVCF to further refine the genotypes

   umake-thunder.pl –conf [config-file.conf]

   make –f [out-prefix].thunder.Makefile –j [# of parallel jobs]

Additional guidance

- Toy example resources for low-coverage calling will available
  - 300kb region in chr20 across 30 individuals from the 1000 Genomes data
  - Example umake input files + tutorial will be provided
- For large-scale analysis, understanding of cluster systems is needed
  - All file paths should be visible by each cluster node
  - Setting adquate level of parallelism is important
  - For large-scale projects, we recommend to run the pipeline within mini-cluster so that the network and I/O traffic does not interfere across entire system.

Results in 1000 Genomes Project

- Summary of each contributing low coverage SNP call set -

Center	Total # variants	dbSNP% (129)	# novels	Novel ti/tv	Omni poly sensitivity	Omni mono false discovery
Broad	**36.59M**	22.71%	28.28M	2.167	96.51% 2.04M / 2.12M	5.45% 3,253 / 59,721
Sanger	**34.81M**	22.88%	26.84M	2.176	96.13% 2.04M / 2.12M	4.94% 2,948 / 59,721
UMich	**34.52M**	24.36%	26.11M	2.161	98.03% 2.08M / 2.12M	2.77% 1,655 / 59,721
Baylor	**34.14M**	21.79%	26.70M	2.131	93.77% 1.99M / 2.12M	1.43% 856 / 59,721
BC	**33.32M**	23.89%	25.36M	2.098	94.86% 2.01M / 2.12M	9.72% 5,808 / 59,721
NCBI	**30.69M**	25.67%	22.81M	2.329	94.60% 2.00M / 2.12M	10.47% 6,254 / 59,721

Michigan call set has highest sensitivity and second lowest FDR among the 6 contributing sets

Slide by Ryan Poplin at Broad Institute

GoT2D : Chr20 Variant Calls from 1,514 Genomes (4x)

Category	#SNPs	%dbSNP	Known Ts/Tv	Novel Ts/Tv	Overall Ts/Tv
UNFILTERED	468,162	34.2%	2.29	1.92	2.04
FAIL	85,092	12.0%	1.56	1.01	1.06
PASS	**383,070**	**39.1%**	**2.35**	**2.42**	**2.39**

Category	HM3-ALL Sensitivity	OMNI-FIN Sensitivity	OMNI-GBR Sensitivity	Exome* Sensitivity	Exome* Sens.(AF>1%)
UNFILTERED	90.3%	99.3%	98.4%	62.4%	N/A
FAIL-FILTER	0.8%	0.8%	0.9%	2.7%	N/A
PASS-FILTER	**89.5%**	**98.3%**	**97.6%**	**59.7%**	**97.4%**

* Note that low Exome sensitivity is mainly due to excessive numbers of singletons (>35%)
+ 15% Exome samples do not exist in the genome samples

ESP : Variant calls from 2,520 Exomes

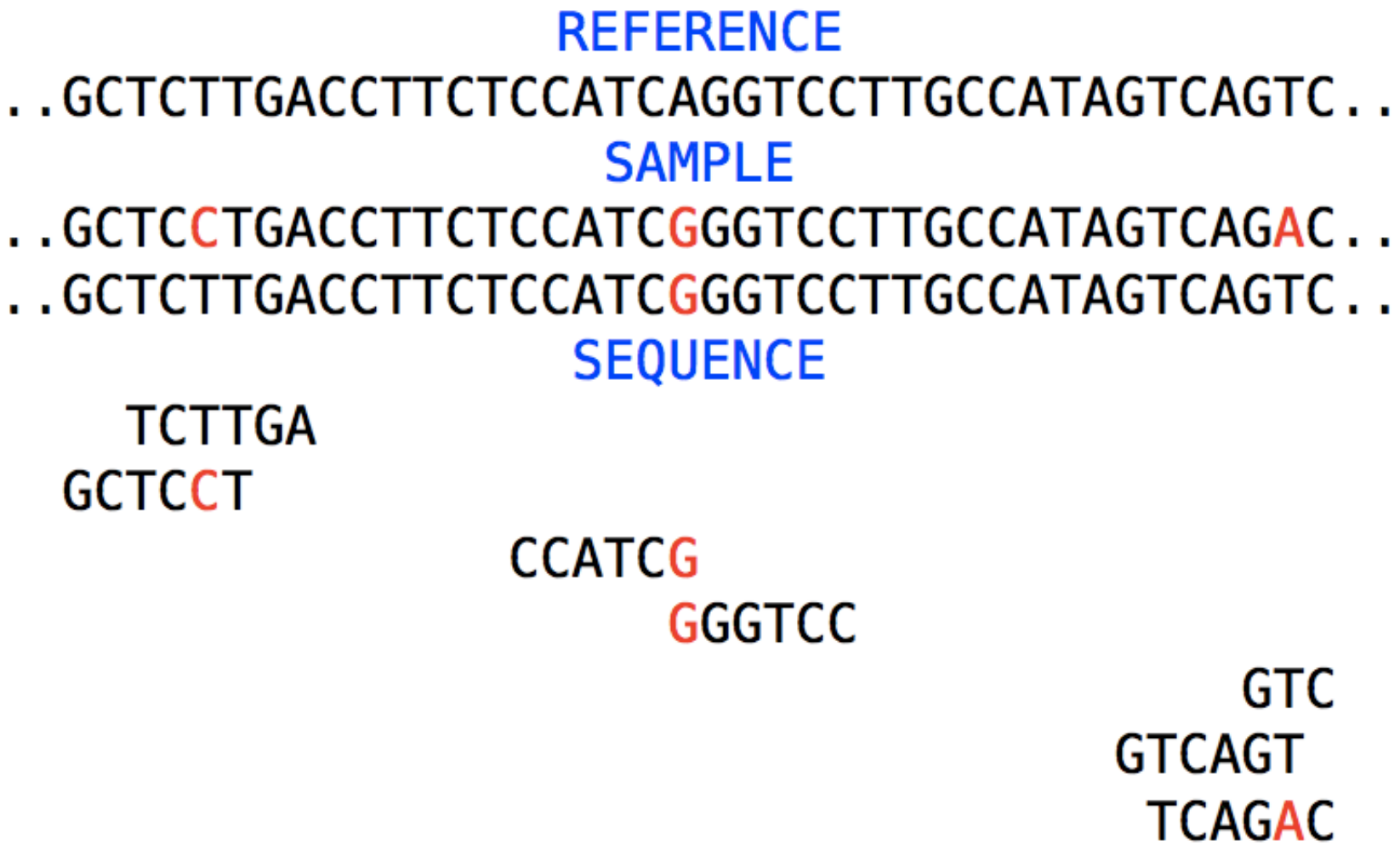
- Variants are called jointly across 2,520 BAMs processed by Broad (1,136) and U of Washington (1,384)
- 1,377 European Americans and 1,143 African Americans

Category	#SNPs	%dbSNP (build 129)	Known Ts/Tv	Novel Ts/Tv	Overall Ts/Tv
ALL SNPs	996,392	14.5	3.01	3.11	3.09
CODING	**620,240**	**14.6**	**3.48**	**3.50**	**3.50**
SILENT	247,645	18.7	5.61	6.18	6.06
MISSENSE	364,586	12.0	2.36	2.67	2.63
NONSENSE	7,896	6.2	2.18	2.39	2.38

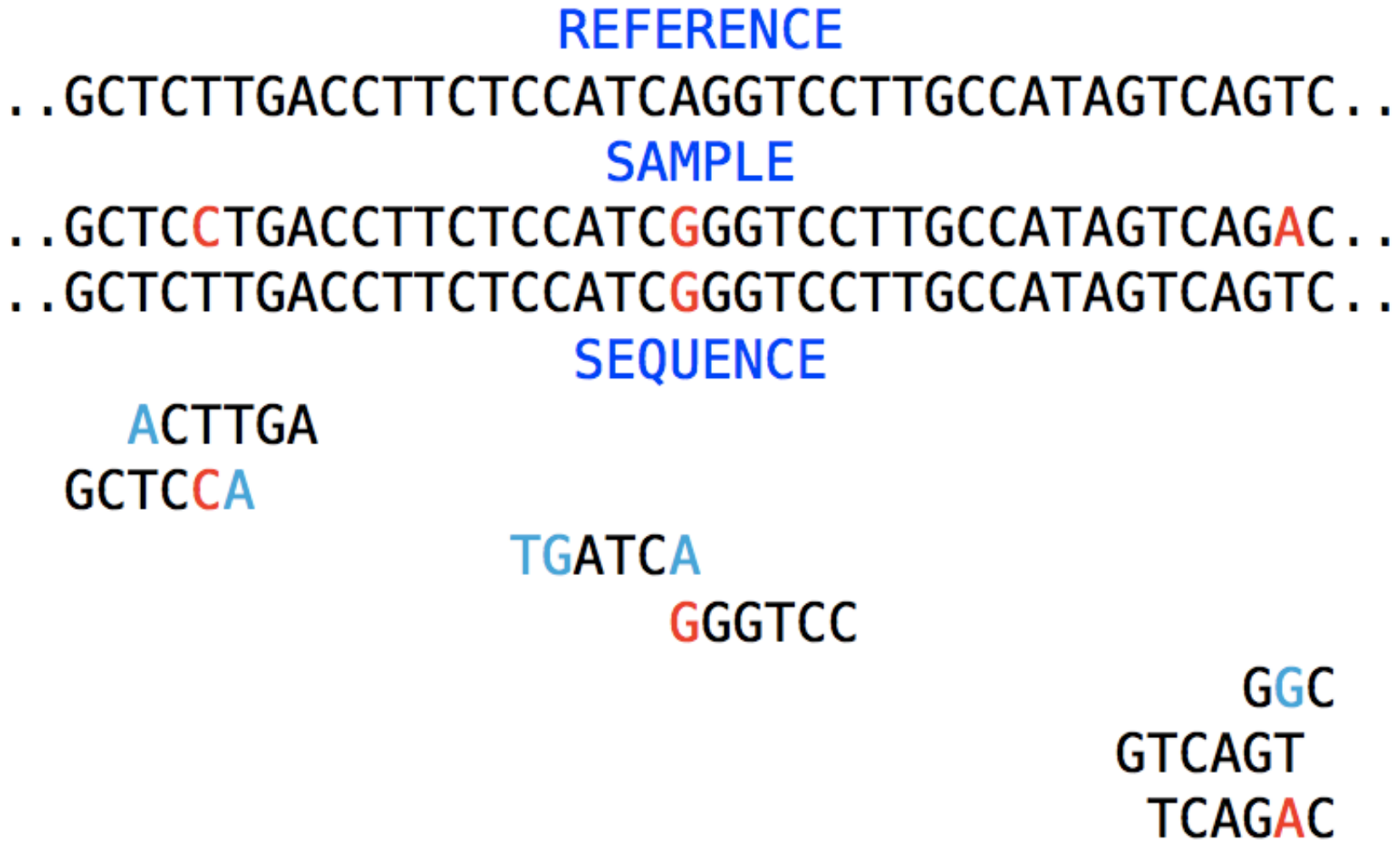
Understanding the details of each component..

- Calculating genotype likelihood from the aligned sequence reads
- Calling SNPs across multiple-samples based on genotype likelihoods
- Filtering against potential false positive calls.

Example of aligned sequence reads



Allowing errors



Genotype likelihood (simple version)

$$\Pr(b_i, Q_i | G_1, G_2) = \begin{cases} 1 - 10^{-Q_i/10} & G_1 = G_2 = b_i \\ \frac{1}{2} - \frac{1}{3}10^{-Q_i/10} & G_1 \neq G_2, b_i = G_1 \text{ (OR } G_2) \\ \frac{1}{3}10^{-Q_i/10} & b_i \neq G_1, b_i \neq G_2 \end{cases}$$

$$\Pr(\mathbf{b}, \mathbf{Q} | G_1, G_2) = \prod_{i=1}^{n} \Pr(b_i, Q_i | G_1, G_2)$$

- Q_i is phred-scale quality of each base
- Simple model under the assumption of independent errors
- Revisied MAQ models correlation between the errors
  - Detailed description available in samtools website

Key model under glfMultiples

- Prior of a site being SNP

$$\Pr(\text{SNP}) = \theta \sum_{i=1}^{2n} \frac{1}{i}, \quad \theta = 10^{-3}$$

- Maximum-likelihood estimate of allele frequency

$$p = \arg\max_{p \in [0,1]} \Pr(\text{Data}|p, A, B) \Pr(A, B)$$

$$\Pr(\text{Data}|p, A, B) = \prod_{i=1}^{m} p^2 \Pr(\mathbf{b}_i, \mathbf{Q}_i|A, A) + 2p(1-p) \Pr(\mathbf{b}_i, \mathbf{Q}_i|A, B) + (1-p)^2 \Pr(\mathbf{b}_i, \mathbf{Q}_i|B, B)$$

- Posterior probability of SNP

$$\Pr(\text{SNP}|\text{Data}) = \max_p \sum_{(A,B) \neq (R,R)} \Pr(\text{Data}|p, A, B) \Pr(A, B)$$

SNP filtering

- Implicit assumption under likelihood-based SNP calling model
  - All the reads are correctly aligned
    - Alignment errors do exist
    - False alignment can create false SNPs
  - Each site is either a monomorphic site or a SNP site
    - Ignores the possibility of being INDELs or SVs
    - Nearby INDELs can create spurious SNP calls
  - The base quality reflects the true error rate of each base
    - Some sites are more prone to errors than others
    - There is reference bias in SOLiD data : higher errors at nonREF sites
  -

Useful features for filtering SNPs

- Total Depth
  - If depth is too high or too low compared to other sites, it can be deletions or duplicated regions
- Strand Bias
  - 2×2 table between (refBase,altBase) × (fwdStrand,revStrand)
  - Departure from null distribution suggests potential strand-specific bias in alignment or base calls
- Allele Balance
  - Individuals with heterozygous alleles are expected to have 50:50 ratio between refBase and altBase.
  - Departure from 50:50 suggest either
    - Alignment artifacts may have contributed to small fraction of non-ref base
    - The region might be duplicated regions where only fraction of them carries actual variants
- Cycle Bias
  - If non-ref bases are enriched at the end of the reads, could be artifacts due to nearby INDELs

Useful features for filtering SNPs

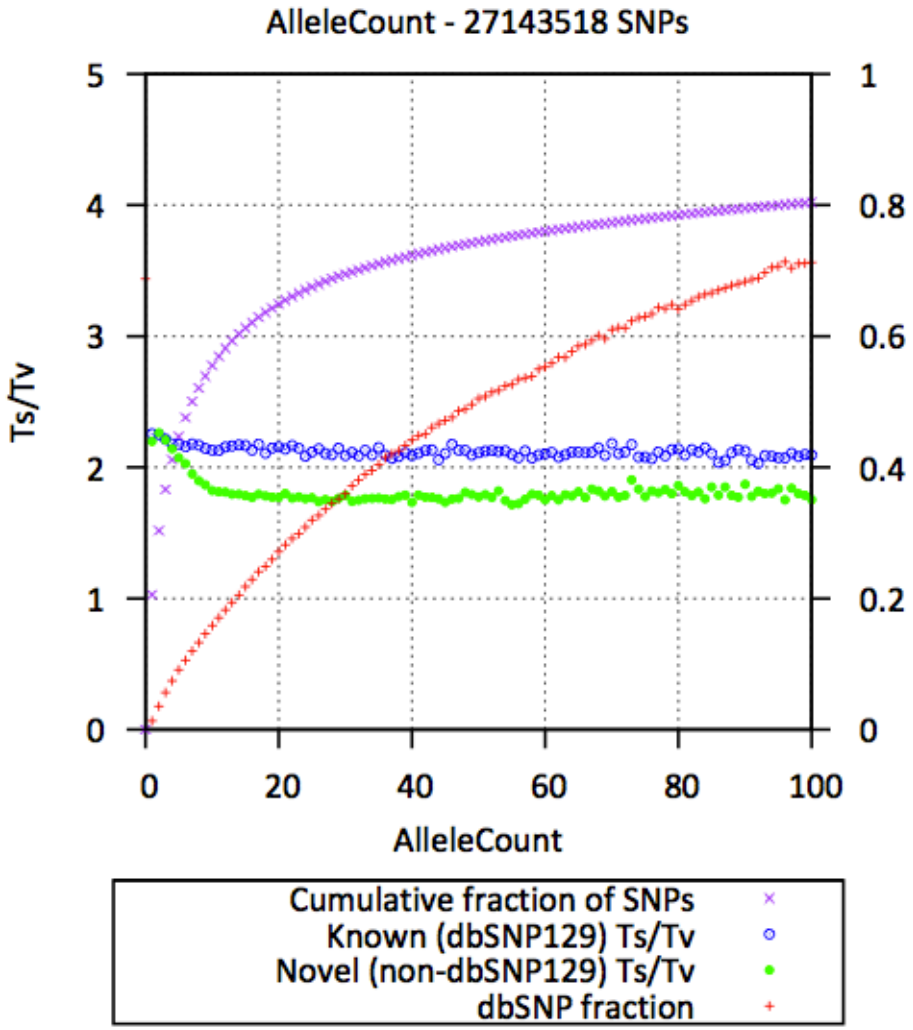
- Low mapping quality
  - If RMS mapping quality is low, or a large % of reads have very low mapping quality, it could create false positive SNP calls
- Nearby INDELs
  - If the SNP is located in close proximity with known INDELs, the SNP is more likely to be affected by the INDELs
- Base-quality over-recalibration
  - If the number of non-ref, not-alt bases are excessive compared to what is expected by base quality, the base quality might have been over-recalibrated, creating false SNPs

Example : 1KG chr20 SNP call summary

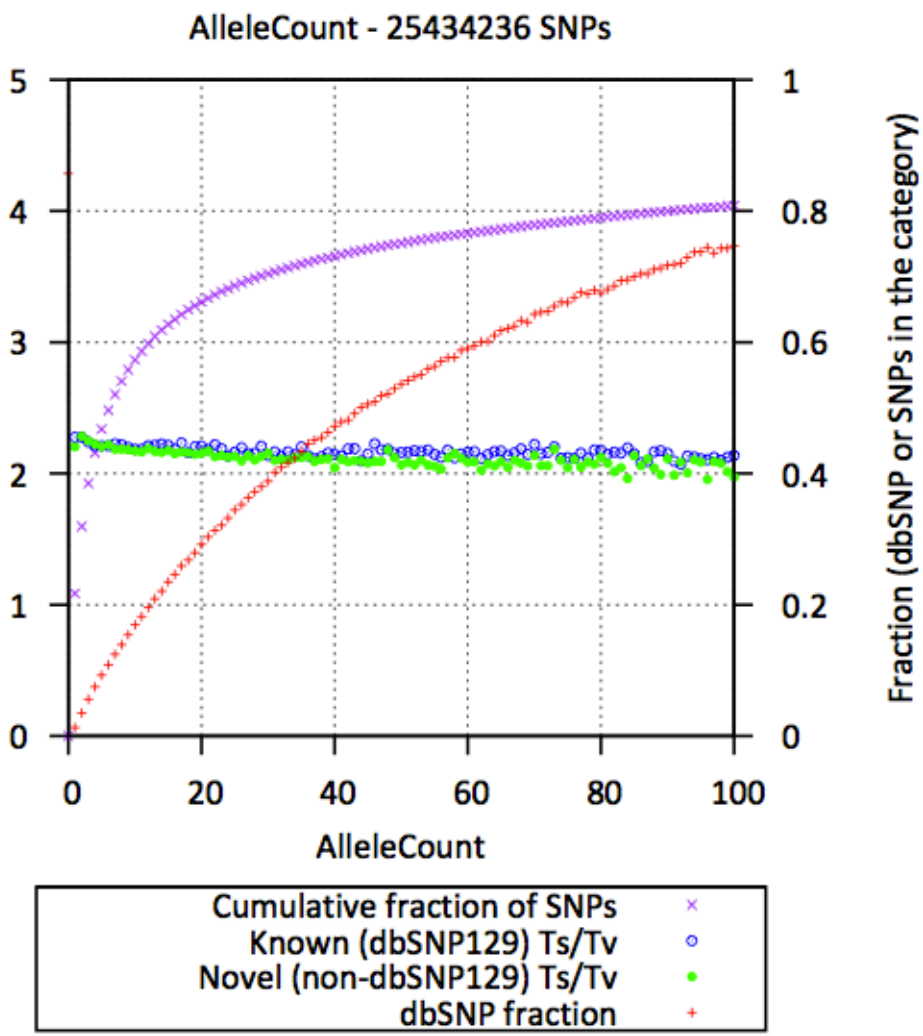
FILTERs	# SNPs	#dbSNP (b129)	%dbSNP (b129)	Known Ts/Tv	Novel Ts/Tv	Overall Ts/Tv	%HM3 Redis-covery
AB>0.67	50,863	3,392	6.7	1.46	0.87	0.90	0.000
DP>20x	5,472	934	17.1	1.30	0.98	1.03	0.000
INDEL 5bp	6,920	1,897	27.4	1.29	1.17	1.20	0.197
STR>0.15	16,199	829	5.1	1.62	0.51	0.54	0.014
DP<0.5x	1,226	179	14.6	1.13	1.10	1.10	0.000
QUAL<5	23,272	451	1.9	1.78	1.91	1.91	0.019
STR<0.15	16,925	1,237	7.3	1.64	0.54	0.59	0.027
PASS	**721,250**	**190,877**	**26.5**	**2.33**	**2.35**	**2.35**	**98.664**
FAIL	**94,462**	**7,770**	**8.2**	**1.47**	**1.04**	**1.07**	**0.254**
TOTAL	**815,712**	**198,647**	**24.4**	**2.29**	**2.08**	**2.12**	**98.918**

Effect of filtering on quality of SNPs

- Before filtering



- After filtering



Current status and future directions

- Where we are
  - Our SNP calling pipeline is producing high-quality variants across many large-scale projects
  - The pipeline will be publicly released with detailed documentation within 1-2 weeks
- Future directions
  - Avoiding redundant disk I/O for reading BAMs for filtering
    - During pileup, collect additional site-level statistics together
  - Improved filtering
    - Need to properly account for overlapping read pairs.
    - Haplotype consistency per individual level
  - Applying machine-learning approaches for automated variant filtering
  - Empirical calibration of the genotype likelihood
    - Accounting for contamination or reference bias
  - Extension of the pipeline to call INDELs

Acknowledgements

- Goncalo Abecasis
- Carlo Sidore
- Goo Jun
- Mary Kate Trost
- Adrian Tan
- Yun Li
- Tom Blackwell
- Matthew Flickinger
- Matthew Snyder
- Alex Tsoi