MB-GATK-SGE pipeline

For GATK best practices: classic UG / v3.x HC / MuTect
Matthew Bashton

Classic Unified Genotyper workflow

BAM files merged using Picard threading used to off-load (de)compression/IO, shell script takes path/*.bam as input from command line

End of sample level workflow

Merge BAM files `Merge_BAM` `MergeBAM.sh`

Reads are realigned around indels, two stages:
i) Realignment Target Creation,
ii) Indel Realignment

2nd realignment of all reads `Realn_merged` `Merged_RTC.sh` `Merged_IDR.sh`

Q scores for each base are recalibrated using machine learning. Two stages i) build model ii) apply it and "print" a

Base Q Score Recal `BQSR_merged` `Merged_BaseRecal.sh` `Merged_PrintReads.sh`

Variants called on all samples simultaneously, using Unified Genotyper, calls SNPs and indels separately owing to size of

Gaussian mixture model trained using 1000G, HapMap, dbSNP and Omni array data. Recalibrated variants are filtered at a desired truth level, SNPs and

Unified Genotyper `UG_merged` `UG_snps.sh` `UG_indels.sh`

Varient Q Score Recalibration `VQSR_UG` `VQSR_snps_UG.sh` `VQSE_indels_UG.sh`

`ApplyRecalibration_snps_UG.sh`
`ApplyRecalibration_indels_UG.sh`

Common per-sample processing

Per sample level workflow

START Raw Reads `FASTQ`

Input is Raw .fastq.gz ideally reads should not be trimmed

Align Reads `BWA_MEM` `BWA.sh` `Submit_BWA.pl`

BWA gives SAM output, -R used to

SAM to sorted index BAM `SamToSortedBam` `SamToSortedBam.sh`

SAM converted to sorted index BAM in one step using

Find PCR duplicates `MarkDuplicates` `MarkDuplicates.sh`

Duplicates marked not removed, using Picard: MarkDuplicates, this is more effective at finding duplicates than SAMtools

Sample level realignment `1stRealn` `1st_RTC.sh` `1st_IDR.sh`

Reads are realigned around indels, two stages:
i) Realignment Target Creation,
ii) Indel Realignment

New Haplotype Caller workflow

`BaseRecal.sh` `PrintReads.sh`

Base Q Score Recal `BQSR_sample_lvl`

Q scores for each base are recalibrated using machine learning. Two stages i) build model ii) apply it and "print" a

`HC.sh`

Haplotype Caller per sample `HC_sample_lvl`

Variants called in new genomic VCF mode at sample level, this is quick, GATK 3.x uses AVX accelerated PairHMM on new CPUs

End of sample level workflow

`GenotypeGVCFs.sh`

Genotype and fuse gVFCs to a single VCF `GenotypeGVCFs`

Individual gVCF files are fused here to make a single VCF, downstream analysis is as before

`VQSR_snps_HC.sh`
`VQSE_indels_HC.sh`

Variant Q Score Recalibration `VQSR_HC`

Gaussian mixture model trained using 1000G, HapMap, dbSNP and Omni array data. Recalibrated variants are filtered at a desired truth level, SNPs and indels should not undergo VQSR together

`ApplyRecalibration_snps_HC.sh`
`ApplyRecalibration_indels_HC.sh`

MuTect 1.x somatic variant calling

`MT.sh`
`Submit_MT.pl`

Call tumor / normal pairs using MuTect `MuTect`

MuTect subtracts the normal (germline) variants from the tumor (somatic) variants. It also reports if SNPs are novel i.e. not in COSMIC or dbSNP

MuTect handels heterogeneous and impure

Perl script submits MuTect

Recalibrated variant filtering

`SelectRecaledVariants_snps.sh`
`SelectRecaledVariants_indels.sh`

Select variants: passing recalibration, VQSlod >= 0, VQSlod >= 3 `Filt_Recaled_VCF`

Recalibrated variants can be filtered for those passing recalibration via the PASS flag. This can be further filtered via the VQSlod log odds ratio which is the likelihood of being a true variant versus being false under the trained Gaussian mixture model. VQSlod >= 0 tends to be a better subset with >= 3 being even better in terms of variant quality

`HardFilt_both_HC.sh`
`HardFilt_indels_UG.sh`
`HardFilt_snps_UG.sh`

Hard Filter variants if VQSR fails `Hard_filt`

Should recalibration fail (owing to lack of bad variants) then hard (i.e. preset) filters can be applied to both the SNPs and indels