Tumor Only Variant Detection Workflow

Genomics and Molecular Pathology Lab

UT Southwestern Medical Center

Author: Brandi Cantarel

Version: 1.0

Effective Date: 06/08/2017

1. Introduction

This SOP describes the Tumor Only Variant Detection Workflow used for NGS analysis in the UTSW Genomics and Molecular Pathology Lab (GMPL). This pipeline identifies single nucleotide variants (SNV), insertion and deletion (Indel) variants in genome/exome/target panel sequencing data.

2. Requirements

2.1. Data requirements

The pipeline requires as input BAM (alignment) files generated by the alignment workflow. The pipeline also requires the following reference data: (1) the fasta sequence of a reference genome, currently Human GRCh38; (2) A set of reference single nucleotide polymorphisms (SNPs) from the NCBI SNP database and (3) A set of reference indels from the 1000 genomes project.

3. Procedures

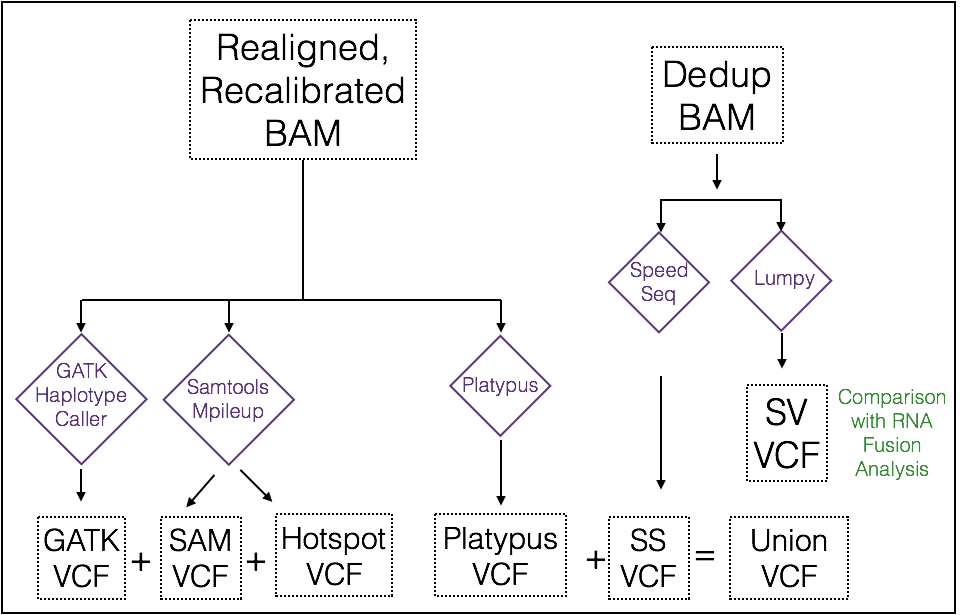


Figure 1: Procedure Overview

3.1. Calling Structural Variants

Identify structural variants.

|  |  |
| --- | --- |
| executable | samtools/intel/1.3 bedtools/2.25.0 bcftools/intel/1.3 snpeff/4.2 speedseq/20160506 vcftools/0.1.14 vcf2bed.sv.pl |
| input | reference genome fasta, sorted-deduplicated bam, bed files with the gene and exon locations on the reference genome |
| output | structural variant vcf files and table of gene annotated structural variants |
| command | >/project/BICF/BICF\_Core/shared/seqprg/delly/src/delly call -t TRA -o delly\_translocations.bcf -q 30 -g ${reffa} ${ssbam}  >/project/BICF/BICF\_Core/shared/seqprg/delly/src/delly call -t DUP -o delly\_duplications.bcf -q 30 -g ${reffa} ${ssbam}  >/project/BICF/BICF\_Core/shared/seqprg/delly/src/delly call -t INV -o delly\_inversions.bcf -q 30 -g ${reffa} ${ssbam}  >/project/BICF/BICF\_Core/shared/seqprg/delly/src/delly call -t DEL -o delly\_deletion.bcf -q 30 -g ${reffa} ${ssbam}  >/project/BICF/BICF\_Core/shared/seqprg/delly/src/delly call -t INS -o delly\_insertion.bcf -q 30 -g ${reffa} ${ssbam}  >/project/BICF/BICF\_Core/shared/seqprg/delly/src/delly filter -t TRA -o delly\_tra.bcf -f germline delly\_translocations.bcf  >/project/BICF/BICF\_Core/shared/seqprg/delly/src/delly filter -t DUP -o delly\_dup.bcf -f germline delly\_translocations.bcf  >/project/BICF/BICF\_Core/shared/seqprg/delly/src/delly filter -t INV -o delly\_inv.bcf -f germline delly\_translocations.bcf  >/project/BICF/BICF\_Core/shared/seqprg/delly/src/delly filter -t DEL -o delly\_del.bcf -f germline delly\_translocations.bcf  >/project/BICF/BICF\_Core/shared/seqprg/delly/src/delly filter -t INS -o delly\_ins.bcf -f germline delly\_translocations.bcf  >bcftools concat -a -O v delly\_dup.bcf delly\_inv.bcf delly\_tra.bcf delly\_del.bcf delly\_ins.bcf | vcf-sort > ${pair\_id}.delly.vcf  >perl $baseDir/scripts/vcf2bed.sv.pl ${pair\_id}.delly.vcf > delly.bed  >bgzip ${pair\_id}.delly.vcf  >tabix ${pair\_id}.delly.vcf.gz  >sambamba sort -t \$SLURM\_CPUS\_ON\_NODE -n -o namesort.bam ${ssbam}  >sambamba view -h namesort.bam | samblaster -M -a --excludeDups --addMateTags --maxSplitCount 2 --minNonOverlap 20 -d discordants.sam -s splitters.sam > temp.sam  >gawk '{ if (\$0~"^@") { print; next } else { \$10="\*"; \$11="\*"; print } }' OFS="\\t" splitters.sam | samtools view -S -b - | samtools sort -o splitters.bam -  >gawk '{ if (\$0~"^@") { print; next } else { \$10="\*"; \$11="\*"; print } }' OFS="\\t" discordants.sam | samtools view -S -b - | samtools sort -o discordants.bam -  >speedseq sv -t \$SLURM\_CPUS\_ON\_NODE -o ${pair\_id}.sssv -R ${reffa} -B ${ssbam} -D discordants.bam -S splitters.bam -x ${index\_path}/exclude\_alt.bed  >java -jar \$SNPEFF\_HOME/SnpSift.jar filter "GEN[0].SU > 2" ${pair\_id}.sssv.sv.vcf.gz > lumpy.vcf  >perl $baseDir/scripts/vcf2bed.sv.pl lumpy.vcf > lumpy.bed  >bedtools intersect -v -a lumpy.bed -b delly.bed > lumpy\_only.bed  >bedtools intersect -header -b lumpy\_only.bed -a lumpy.vcf |bgzip > lumpy\_only.vcf.gz  >vcf-concat ${pair\_id}.delly.vcf.gz lumpy\_only.vcf.gz |vcf-sort -t temp > ${pair\_id}.sv.vcf  >perl $baseDir/scripts/vcf2bed.sv.pl ${pair\_id}.sv.vcf |sort -V -k 1,1 -k 2,2n | grep -v 'alt' |grep -v 'random' |uniq > svs.bed  >bedtools intersect -header -wb -a svs.bed -b ${index\_path}/gencode.exons.bed > exonoverlap\_sv.txt  >bedtools intersect -v -header -wb -a svs.bed -b ${index\_path}/gencode.exons.bed | >bedtools intersect -header -wb -a stdin -b ${index\_path}/gencode.genes.chr.bed > geneoverlap\_sv.txt  >perl $baseDir/scripts/annot\_sv.pl -r ${index\_path} -i ${pair\_id}.sv.vcf  >bgzip ${pair\_id}.sv.vcf |

3.2. Variant Calling and Filtering using GATK

Identify variants using GATK

|  |  |
| --- | --- |
| executable | gatk/3.5 python/2.7.x-anaconda bedtools/2.25.0 snpeff/4.2 vcftools/0.1.14 |
| input | reference genome fasta, realigned/recalibrated bam, dbsnp vcf |
| output | gvcf and vcf |
| command | >java -Xmx32g -jar \$GATK\_JAR -R ${reffa} -D ${dbsnp} -T HaplotypeCaller -stand\_call\_conf 30 -stand\_emit\_conf 10.0 -A FisherStrand -A QualByDepth -A VariantType -A DepthPerAlleleBySample -A HaplotypeScore -A AlleleBalance -variant\_index\_type LINEAR -variant\_index\_parameter 128000 --emitRefConfidence GVCF -I ${gbam} -o ${pair\_id}.gatk.g.vcf -nct 2  >java -Xmx32g -jar \$GATK\_JAR -R ${reffa} -D ${dbsnp} -T GenotypeGVCFs -o gatk.vcf -nt 4 --variant ${pair\_id}.gatk.g.vcf  vcf-annotate -n --fill-type gatk.vcf | bcftools norm -c s -f ${reffa} -w 10 -O z -o ${pair\_id}.gatk.vcf.gz -  >tabix ${pair\_id}.gatk.vcf.gz |

3.3. Variant Calling and Filtering using SAMtools

Call variants with mapping quality >= 50 and base calling quality >= 20, and filter them based on the following cutoffs: QUAL >= 10 & MQ >= 20 & DP >= 10. Only capturing panel variants will be remained.

|  |  |
| --- | --- |
| executable | python/2.7.x-anaconda samtools/intel/1.3 bedtools/2.25.0 bcftools/intel/1.3 snpeff/4.2 vcftools/0.1.14 |
| input | reference genome fasta, realigned/recalibrated bam, dbsnp vcf |
| output | vcf |
| command | >samtools mpileup -t 'AD,DP,INFO/AD' -ug -Q20 -C50 -f ${reffa} ${gbam} | bcftools call -vmO z -o ${pair\_id}.sam.ori.vcf.gz  >vcf-sort ${pair\_id}.sam.ori.vcf.gz | vcf-annotate -n --fill-type | bcftools norm -c s -f ${reffa} -w 10 -O z -o ${pair\_id}.sam.vcf.gz - |

3.4. Hotspot Variant Identification

List COSMIC variants with variant allele fraction > 0.01 and depth > 50.

|  |  |
| --- | --- |
| executable | python/2.7.x-anaconda samtools/intel/1.3 bedtools/2.25.0 bcftools/intel/1.3 snpeff/4.2 vcftools/0.1.14 |
| input | reference genome fasta, realigned/recalibrated bam, dbsnp vcf, cosmic.vcf.gz |
| output | hotspot vcf |
| command | >samtools mpileup -d 99999 -t 'AD,DP,INFO/AD' -uf ${reffa} ${gbam} > ${pair\_id}.mpi  >bcftools filter -i "AD[1]/DP > 0.01" ${pair\_id}.mpi | bcftools filter -i "DP > 50" | bcftools call -m -A |vcf-annotate -n --fill-type | bcftools norm -c s -f /project/shared/bicf\_workflow\_ref/GRCh38/genome.fa -w 10 -O z -o ${pair\_id}.lowfreq.vcf.gz -  >java -jar \$SNPEFF\_HOME/SnpSift.jar annotate ${index\_path}/cosmic.vcf.gz ${pair\_id}.lowfreq.vcf.gz | java -jar \$SNPEFF\_HOME/SnpSift.jar filter "(CNT[\*] >0)" - |bgzip > ${pair\_id}.hotspot.vcf.gz |

3.5. Variant Calling and Filtering using SpeedSeq

Detect variants using SpeedSeq and filter with the following cutoffs: QUAL >= 10 & DP >= 10. Only on-target variants will be remained.

|  |  |
| --- | --- |
| executable | python/2.7.x-anaconda samtools/intel/1.3 bedtools/2.25.0 bcftools/intel/1.3 snpeff/4.2 speedseq/20160506 vcftools/0.1.14 |
| input | reference genome fasta, realigned/recalibrated bam |
| output | vcf |
| command | >speedseq var -t \$SLURM\_CPUS\_ON\_NODE -o ssvar ${reffa} ${gbam}  >vcf-annotate -n --fill-type ssvar.vcf.gz| bcftools norm -c s -f ${reffa} -w 10 -O z -o ${pair\_id}.ssvar.vcf.gz - |

3.6. Variant Calling and Filtering using Platypus

Detect variants using Platypus and filter with the following cutoffs: QUAL >= 10 & QC > 2. Only on-target variants will be remained.

|  |  |
| --- | --- |
| executable | python/2.7.x-anaconda bedtools/2.25.0 snpeff/4.2 platypus/gcc/0.8.1 bcftools/intel/1.3 samtools/intel/1.3 vcftools/0.1.14 |
| input | reference genome fasta, realigned/recalibrated bam |
| output | vcf |
| command | >Platypus.py callVariants --minMapQual=10 --mergeClusteredVariants=1 --nCPU=\$SLURM\_CPUS\_ON\_NODE --bamFiles=${gbam} --refFile=${reffa} --output=platypus.vcf  >bgzip platypus.vcf  >tabix platypus.vcf.gz  >vcf-sort platypus.vcf.gz| vcf-annotate -n --fill-type -n | bcftools norm -c s -f ${reffa} -w 10 -O z -o ${pair\_id}.platypus.vcf.gz - |

3.7. generate union

Integrate result variant lists generated by GATK, SAMtools, Hotspot algorithm, SpeedSeq, and Platypus variant callers. The priority order for ambiguous variants is SAMtools - SpeedSeq - GATK - Platypus - Hotspot.

|  |  |
| --- | --- |
| executable | gatk/3.5 python/2.7.x-anaconda bedtools/2.25.0 snpeff/4.2 bcftools/intel/1.3 samtools/intel/1.3 vcftools/0.1.14 |
| input | reference genome fasta, GATK, SAMtools, Hotspot, SpeedSeq, Platypus gzip vcf |
| output | union vcf |
| command | >bedtools multiinter -i ${gatk} ${sam} ${ss} ${plat} ${hs} -names gatk sam ssvar platypus hotspot |cut -f 1,2,3,5 | bedtools sort -i stdin | bedtools merge -c 4 -o distinct > ${fname}\_integrate.bed  >bedtools intersect -header -v -a ${hs} -b ${sam} |bedtools intersect -header -v -a stdin -b ${gatk} | bedtools intersect -header -v -a stdin -b ${ss} | >bedtools intersect -header -v -a stdin -b ${plat} | bgzip > ${fname}.hotspot.nooverlap.vcf.gz  >vcf-sort ${sam} |bgzip > sam.vcf.gz  >tabix ${fname}.hotspot.nooverlap.vcf.gz  >tabix ${gatk}  >tabix sam.vcf.gz  >tabix ${ss}  >tabix ${plat}  >java -Xmx32g -jar \$GATK\_JAR -R ${reffa} -T CombineVariants --filteredrecordsmergetype KEEP\_UNCONDITIONAL --variant:gatk ${gatk} --variant:sam sam.vcf.gz --variant:ssvar ${ss} --variant:platypus ${plat} --variant:hotspot ${fname}.hotspot.nooverlap.vcf.gz -genotypeMergeOptions PRIORITIZE -priority sam,ssvar,gatk,platypus,hotspot -o ${fname}.int.vcf  >perl $baseDir/scripts/uniform\_integrated\_vcf.pl ${fname}.int.vcf  >bgzip ${fname}\_integrate.bed  >tabix ${fname}\_integrate.bed.gz  >bgzip ${fname}.uniform.vcf  >tabix ${fname}.uniform.vcf.gz  >bcftools annotate -a ${fname}\_integrate.bed.gz --columns CHROM,FROM,TO,CallSet -h ${index\_path}/CallSet.header ${fname}.uniform.vcf.gz | bgzip > ${fname}.union.vcf.gz |

3.8. Annotate VCF

Annotate VCF with publicly available genes

|  |  |
| --- | --- |
| executable | python/2.7.x-anaconda bedtools/2.25.0 snpeff/4.2 bcftools/intel/1.3 samtools/intel/1.3 |
| input | reference genome fasta, union vcf |
| output | annotated vcf |
| command | >bcftools annotate -Oz -a ${index\_path}/ExAC.vcf.gz -o ${fname}.exac.vcf.gz --columns CHROM,POS,AC\_Het,AC\_Hom,AC\_Hemi,AC\_Adj,AN\_Adj,AC\_POPMAX,AN\_POPMAX,POPMAX ${unionvcf}  >tabix ${fname}.exac.vcf.gz  >bcftools annotate -Oz -a ${index\_path}/dbSnp.vcf.gz -o ${fname}.dbsnp.vcf.gz --columns CHROM,POS,ID,RS ${fname}.exac.vcf.gz  >tabix ${fname}.dbsnp.vcf.gz  >bcftools annotate -Oz -a ${index\_path}/clinvar.vcf.gz -o ${fname}.clinvar.vcf.gz --columns CHROM,POS,CLNSIG,CLNDSDB,CLNDSDBID,CLNDBN,CLNREVSTAT,CLNACC ${fname}.dbsnp.vcf.gz  >tabix ${fname}.clinvar.vcf.gz  >bcftools annotate -Oz -a ${index\_path}/utswv2\_artifact.bed.gz -o ${fname}.utswbl.vcf.gz -m "UTSWBlacklist" -c CHROM,FROM,TO ${fname}.clinvar.vcf.gz  >tabix ${fname}.utswbl.vcf.gz  >java -Xmx10g -jar \$SNPEFF\_HOME/snpEff.jar -no-intergenic -lof -c \$SNPEFF\_HOME/snpEff.config ${snpeff\_vers} ${fname}.utswbl.vcf.gz | java -jar \$SNPEFF\_HOME/SnpSift.jar annotate ${index\_path}/cosmic.vcf.gz - | java -Xmx10g -jar \$SNPEFF\_HOME/SnpSift.jar dbnsfp -v -db ${index\_path}/dbNSFP.txt.gz - | java -Xmx10g -jar \$SNPEFF\_HOME/SnpSift.jar gwasCat -db ${index\_path}/gwas\_catalog.tsv - |bgzip > ${fname}.annot.vcf.gz  >tabix ${fname}.annot.vcf.gz  >bcftools stats ${fname}.annot.vcf.gz > ${fname}.stats.txt  >plot-vcfstats -s -p ${fname}.statplot ${fname}.stats.txt |

4. Implementation

The Workflow used in this SOP can be downloaded here: <https://git.biohpc.swmed.edu/brandi.cantarel/clinseq_workflows>

Usage:

>nextflow -C /project/PHG/PHG\_Clinical/clinseq\_workflows/nextflow.config run -with-timeline -w $workdir /project/PHG/PHG\_Clinical/clinseq\_workflows/alignment.nf --design $outdir\/design.txt --capture $capture --input $outdir --output $outnf &> $outnf\/nextflow.log

where the work\_directory is the folder where the work files for each step will be deposited, the input directory is the name of the folder where the bam files reside, the output directory is the name of the folder where the output files will be written, the design.txt is a tab delimited file that contains the sample id and bamfiles, and the nextflow.log is the file that keeps the log of the run and tracks workflow errors.

5. Related Documents & References

* <http://www.bioinformatics.babraham.ac.uk/projects/fastqc/>
  + <http://www.bioinformatics.babraham.ac.uk/projects/trim_galore/>
* <http://bio-bwa.sourceforge.net/>
  + Li H, Durbin R. Fast and accurate short read alignment with Burrows-Wheeler transform. Bioinformatics. 2009 Jul 15;25(14):1754-60.
  + <https://github.com/lh3/seqtk>
  + <https://github.com/lh3/bwa/tree/master/bwakit>
* <http://samtools.sourceforge.net/>
  + Li H, Handsaker B, Wysoker A, Fennell T, Ruan J, Homer N, Marth G, Abecasis G, Durbin R; 1000 Genome Project Data Processing Subgroup. The Sequence Alignment/Map format and SAMtools. Bioinformatics. 2009 Aug 15;25(16):2078-9.
  + Li H. A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. Bioinformatics. 2011 Nov 1;27(21):2987-93.<http://samtools.sourceforge.net/>
* <http://lomereiter.github.io/sambamba/>
  + Tarasov A, Vilella AJ, Cuppen E, Nijman IJ, Prins P. Sambamba: fast processing of NGS alignment formats. Bioinformatics. 2015 Jun 15;31(12):2032-4.<http://lomereiter.github.io/sambamba/>
* <https://broadinstitute.github.io/picard/>
* <https://github.com/hall-lab/speedseq>
  + Chiang C, Layer RM, Faust GG, Lindberg MR, Rose DB, Garrison EP, Marth GT, Quinlan AR, Hall IM. SpeedSeq: ultra-fast personal genome analysis and interpretation. Nat Methods. 2015 Oct;12(10):966-8.<https://github.com/hall-lab/speedseq>
* <https://software.broadinstitute.org/gatk/>
  + McKenna A, Hanna M, Banks E, Sivachenko A, Cibulskis K, Kernytsky A, Garimella K, Altshuler D, Gabriel S, Daly M, DePristo MA. The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. Genome Res. 2010 Sep;20(9):1297-303.
  + DePristo MA, Banks E, Poplin R, Garimella KV, Maguire JR, Hartl C, Philippakis AA, del Angel G, Rivas MA, Hanna M, McKenna A, Fennell TJ, Kernytsky AM, Sivachenko AY, Cibulskis K, Gabriel SB, Altshuler D, Daly MJ. A framework for variation discovery and genotyping using next-generation DNA sequencing data. Nat Genet. 2011 May;43(5):491-8.
  + Van der Auwera GA, Carneiro MO, Hartl C, Poplin R, Del Angel G, Levy-Moonshine A, Jordan T, Shakir K, Roazen D, Thibault J, Banks E, Garimella KV, Altshuler D, Gabriel S, DePristo MA. From FastQ data to high confidence variant calls: the Genome Analysis Toolkit best practices pipeline. Curr Protoc Bioinformatics. 2013;43:11.10.1-33.
  + <http://gatkforums.broadinstitute.org/gatk>
* <http://bedtools.readthedocs.io/en/latest/>
  + Quinlan AR, Hall IM. BEDTools: a flexible suite of utilities for comparing genomic features. Bioinformatics. 2010 Mar 15;26(6):841-2.<http://bedtools.readthedocs.io/en/latest/>
* <http://vcftools.sourceforge.net/index.html>
  + Danecek P, Auton A, Abecasis G, Albers CA, Banks E, DePristo MA, Handsaker RE, Lunter G, Marth GT, Sherry ST, McVean G, Durbin R; 1000 Genomes Project Analysis Group. The variant call format and VCFtools. Bioinformatics. 2011 Aug 1;27(15):2156-8.<http://vcftools.sourceforge.net/index.html>
* <http://snpeff.sourceforge.net/>
  + Cingolani P, Platts A, Wang le L, Coon M, Nguyen T, Wang L, Land SJ, Lu X, Ruden DM. A program for annotating and predicting the effects of single nucleotide polymorphisms, SnpEff: SNPs in the genome of Drosophila melanogaster strain w1118; iso-2; iso-3. Fly (Austin). 2012 Apr-Jun;6(2):80-92.
  + Cingolani P, Patel VM, Coon M, Nguyen T, Land SJ, Ruden DM, Lu X. Using Drosophila melanogaster as a Model for Genotoxic Chemical Mutational Studies with a New Program, SnpSift. Front Genet. 2012 Mar 15;3:35.<http://snpeff.sourceforge.net/>
* <http://www.well.ox.ac.uk/platypus>
  + Rimmer A, Phan H, Mathieson I, Iqbal Z, Twigg SR; WGS500 Consortium, Wilkie AO, McVean G, Lunter G. Integrating mapping-, assembly- and haplotype-based approaches for calling variants in clinical sequencing applications. Nat Genet. 2014 Aug;46(8):912-8.<http://www.well.ox.ac.uk/platypus>

6. Revision History