Supplemental Whole Exome Sequencing Analysis Methods
Somatic mutation detection; adapted from Teer etal “Evaluating somatic tumor mutation detection without matched normal samples” Hum Genomics 2017. < PMC5584341>

Settings were initially informed by 1000 Genomes phase 2 and GATK best practices:
ftp://ftp-trace.ncbi.nih.gov/1000genomes/ftp/README.alignment_data
https://www.broadinstitute.org/gatk/guide/pdfdocs/GATK_GuideBook_2.3-9.pdf
GATK_Lite 2.2-16 was used - settings may be different for other versions.

Step 0: Trim sequence reads

- Remove adapters from raw FASTQ sequence reads with cutadapt 1.16.

```
cutadapt \
       -m 30      \
       -a <ADPT_R1> \
       -A <ADPT_R2> \
       -o <FASTQ.1> \
       -p <FASTQ.2> \
       --trim-n  \
       --cores 4 \
       <IN_FASTQ.1> \
       <IN_FASTQ.2>
```

Step 1: Sequence Alignment

- Align with BWA 0.7.7 (paired-end):

```
bwa aln -q 15 <reference>[1] <FASTQ.1> -f <out.1.sai>
bwa aln -q 15 <reference> <FASTQ.2> -f <out.2.sai>
bwa sampe -a <max_insert_size>[2] \
  -r “@RG\tID:${NAME}\tSM:${NAME}\tPL:ILLUMINA\tLB:${NAME}_lib" \
  <reference> \
  <out.1.sai> \
  <out.2.sai> \
  <FASTQ.1> \
  <FASTQ.2> \
  -f <out.sam>
```

[1] hs37d5 was used in this study.
[2] A value of 600 was used in this study.

Step 2: SAM to BAM, sort, fixmate, add MD

- Sort, correct with samtools 0.1.18:

```
samtools view -bSu <out.sam>  | \
  samtools sort -n -o -m 3000000000 - <out.sort.tmp> | \
  samtools fixmate /dev/stdin /dev/stdout   | \
  samtools sort -o -m 3000000000 - <out.csort.tmp> | \
  samtools fillmd -b - <reference.fa> \
  > <out.fixed.bam>
```

Step 3: Mark duplicates

- Mark duplicates with Picard 1.82:

```
java -Xmx6g -jar MarkDuplicates.jar \
  INPUT=<out.fixed.bam> \
  OUTPUT=<out.dup.bam> \
  ASSUME_SORTED=TRUE \
  VALIDATION_STRINGENCY=LENIENT \
  METRICS_FILE=<out.dup.metrics> \
  CREATE_INDEX=TRUE
```

Step 4: Realign around indels

- low_coverage and mills_devine indel VCFs from GATK bundle
- Indel Realignment with GATK Lite 2.2-16:

```
java -Xmx6g -jar GenomeAnalysisTK.jar \
  -T RealignerTargetCreator \
  -R <reference.fa> \
  -I <out.dup.bam> \
  -o <out.intervals> \
  -known <low_coverage_indels.vcf> \
  -known <mills_devine_indels.vcf> \

java -Xmx6g -jar GenomeAnalysisTK.jar \
  -T IndelRealigner \
  -R <reference.fa> \
  -I <out.dup.bam> \
  -targetIntervals <out.intervals> \
  -o <out.realign.bam> \
  -known <low_coverage_indels.vcf> \
  -known <mills_devine_indels.vcf> \
  -LOD 4.0 \
  -model USE_READS
```

Step 5: Base quality recalibration

- dbsnp.vcf from GATK bundle
- BQSR with GATK:

```
java -Xmx6g -jar GenomeAnalysisTK.jar \
  -T BaseRecalibrator \
  -l INFO \
  -L <target_region> \
  -R <reference.fa> \
  -I <out.realign.bam> \
  -knownSites <dbsnp.vcf> \
  --disable_indel_quals \
  -cov ReadGroupCovariate \
  -cov QualityScoreCovariate \
  -cov CycleCovariate \
  -cov ContextCovariate \
  -o <out.recal_data>

java -Xmx6g -jar GenomeAnalysisTK.jar \
  -T PrintReads \
  -l INFO \
  -R <reference.fa> \
  -I <out.realign.bam> \
  -o <out.recal.bam> \
  --disable_indel_quals \
  -BQSR <out.recal_data>
```

Step 6: Add MD tag and index final BAM

- Add MD tag with samtools:

```
samtools calmd -Erb <out.recal.bam> <reference.fa> \
  > <out.bam>
samtools index <out.bam>
```

Step 7: Collect metrics

- Get alignment metrics with Picard

```
java -Xmx6g -jar CollectMultipleMetrics.jar \
  INPUT=<out.bam> \
  REFERENCE_SEQUENCE=<reference.fa> \
  OUTPUT=<out.stats> \
  VALIDATION_STATUS=LENIENT
```

Step 8a: Somatic mutation calling with Strelka 1.0.13 and Tabix 0.2.5

```
## Strelka with more sensitive settings
## (reduce snv and indel noise levels 10x: in config.ini)
cat strelka_config_bwa_exome.ini \
  | sed -e 's/ssnvNoise = 0.0000005/ssnvNoise = 0.00000005/' \
        -e 's/sindelNoise = 0.000001/sindelNoise = 0.0000001/' \
        > config.ini

# Configure strelka run
configureStrelkaWorkflow.pl \
  --normal=<normal.bam> \
  --tumor=<tumor.bam> \
  --ref=<reference.fa> \
  --config=config.ini \
  --output-dir=<sample>

# Run in <sample> directory
make -j 4

# Add genotypes to “all” and “pass” outputs
for type in all passed;
do

  bgzip results/${type}.somatic.snvs.vcf
  tabix -p vcf results/${type}.somatic.snvs.vcf.gz

  zgrep '^##[^IF]' results/${type}.somatic.snvs.vcf.gz \
    > ${type}.somatic.vcf
  sed -e 's/BCNoise/BCNoise_indel/g' -e 's/DP/DP_indel/g' \
    results/${type}.somatic.indels.vcf \
    > ${type}.somatic.indels.format.vcf
  zgrep --no-filename '^##[IF]' results/${type}.somatic.snvs.vcf.gz \
    ${type}.somatic.indels.format.vcf | sort \
    | uniq >> ${type}.somatic.vcf
  zgrep '^#CHROM' results/${type}.somatic.snvs.vcf.gz | sed -e \
    "s/NORMAL/<normal_name>/" -e "s/TUMOR/<tumor_name>/" \
    >> ${type}.somatic.vcf
  zgrep --no-filename -v '^#' results/${type}.somatic.snvs.vcf.gz \
    ${type}.somatic.indels.format.vcf \
    | sort -S 20G -k 1,1n -k 2,2n \
    | perl -a -F"\t" -nle
        'if ($F[0] =~ /([MXY]T?)/){
          push @{$c{$1}}, $_
        }elsif($F[0] =~ /^[0-9]+$/){
          print( join "\t", @F )
        };
        END{
          for my $k ("X","Y","MT"){
            foreach (@{$c{$k}}) {print $_}
          }
        }'  >> ${type}.somatic.vcf
  bgzip ${type}.somatic.vcf
  perl strelka_add_genotype_vcf.pl[3] --vcf ${type}.somatic.vcf.gz \
    > unmerged.${type}.tumor.vcf
  perl strelka_merge_genotypes_vcf.pl[3] unmerged.${type}.tumor.vcf \
    | bgzip -c > ${type}.<output_name>.tumor.vcf.gz
  tabix -p vcf ${type}.<output_name>.tumor.vcf.gz

done
```

[3] See supplemental code files.

Step 8b: Somatic mutation calling with MuTect 1.1.4 and Tabix 0.2.5

```
# Run MuTect to get SNVs
java \
  -Xmx10G \
  -jar mutect.jar \
  --analysis_type MuTect \
  --reference_sequence <reference.fa> \
  --cosmic <cosmic_coding_mutations.vcf>\
  --dbsnp <dbsnp.vcf> \
  --intervals <target_regions> \
  --input_file:normal <normal.bam> \
  --input_file:tumor  <tumor.bam> \
  --out <sample.out> \
  --vcf <sample.snv.vcf> \
  --enable_extended_output \
  --max_alt_alleles_in_normal_count 3 \
  --max_alt_allele_in_normal_fraction 0.05 \
  --coverage_file <sample.cov.wig>

# Run MuTect to get Indels with GATK Lite 2.2-16
java \
  -Xmx10G \
  -jar GenomeAnalysisTK.jar \
  --analysis_type SomaticIndelDetector \
  --reference_sequence <reference.fa> \
  --intervals <target_regions> \
  --input_file:normal <normal.bam> \
  --input_file:tumor  <tumor.bam> \
  --out <sample.indel.vcf>
# merge SNV and Indel mutations
perl mutect_correct_vcf.pl[3] \
  --snv_vcf <sample.snv.vcf> \
  --indel_vcf <sample.indel.vcf> \
  --out <sample.out> \
  --tumor_sample <tumor_name> \
  --normal_sample <normal_name> \
  --tumor_only \
  --pass_only

(grep '^##' sample_tumor.snv.vcf; \
  grep -P \
'^##.*(ID=MM|ID=MQS|ID=NQSBQ|ID=NQSMM|ID=REnd|ID=RStart|ID=SC)|^##Somat
icIndelDetector' \
  sample_tumor.indel.vcf) > sample.tumor.vcf
grep '^#CHROM' sample_tumor.snv.vcf >> sample.tumor.vcf

grep -hv '^#' sample_tumor.snv.vcf sample_tumor.indel.vcf \
  | sort -S 10G -k 1,1n -k 2,2n \
  | perl -a -F"\t" -nle
      'if ($F[0] =~ /([MXY]T?)/){
        push @{$c{$1}}, $_
      }elsif($F[0] =~ /^[0-9]+$/){
        print( join "\t", @F )
      };
      END{
        for my $k ("X","Y","MT"){
          foreach (@{$c{$k}}) {print $_}
        }
      }'  >> sample.tumor.vcf

bgzip sample.tumor.vcf
tabix -p vcf sample.tumor.vcf.gz
```

Step 9: Merge MuTect and Strelka outputs using vcftools 0.1.15

```
vcf-merge \
  -s -t -c any \
  ${mutectVCFs} \
  > mutect_tumor.vcf; \
bgzip mutect_tumor.vcf; \
tabix -p vcf mutect_tumor.vcf.gz

vcf-merge \
  -s -t -c any \
  ${strelkaVCFs} \
  > strelka_tumor.vcf; \
bgzip strelka_tumor.vcf; \
tabix -p vcf strelka_tumor.vcf.gz

# merge MuTect and Strelka VCF files together
```

```
perl merge_strelka_mutect_asyn.pl \
  --strelka_vcf strelka_tumor.vcf.gz \
  --mutect_vcf mutect_tumor.vcf.gz \
  > strelka_mutect.vcf
```

Step 10: Annotate with ANNOVAR
- Add predicted protein alterations
- Add 1000 Genomes allele frequencies

Step 11: Filter mutations bases on quality and context
- Mutations pass IF (called in Strelka as PASS) OR (called in Strelka as any AND called in MuTect)
- Mutations pass IF 1000 Genomes allele frequency is less than 1%.
- Mutations pass if they are predicted to alter protein sequence
- Passing mutations are summarized in a spreadsheet.