```
################################################################################
####kjkj
```

PATHS

```
################################################################################
####
ngsphyPATH="$HOME/git/ngsphy/"
SIMPHY_PROJECT_NAME="SimPhy_usecase"
CURRENT_DIR="/home/merly/test"
CASE_NAME="usecase3"
MYRANDOMNUM=2268317056
GATK="$HOME/apps/gatk/3.8-0-ge9d806836/GenomeAnalysisTK.jar"
PICARD="$HOME/apps/picard/picard.jar"
referenceFile="$CURRENT_DIR/${CASE_NAME}/reference/reference.fasta"
coverages=( "1x" "5x" "10x" "50x" "100x")
################################################################################
####
```

Data organization

```
################################################################################
####
echo "Creating test folder"
mkdir -p ${CURRENT_DIR}/${CASE_NAME}/settings/
${CURRENT_DIR}/${CASE_NAME}/files/ \
${CURRENT_DIR}/${CASE_NAME}/output/ ${CURRENT_DIR}/${CASE_NAME}/src/ \
$CURRENT_DIR/${CASE_NAME}/reference $CURRENT_DIR/${CASE_NAME}/img

echo "Gathering all the data in a single folder"
cp ${ngsphyPATH}/data/settings/ngsphy.settings.case1.1x.txt
${CURRENT_DIR}/${CASE_NAME}/settings/
cp ${ngsphyPATH}/data/settings/ngsphy.settings.case1.5x.txt
${CURRENT_DIR}/${CASE_NAME}/settings/
cp ${ngsphyPATH}/data/settings/ngsphy.settings.case1.10x.txt
${CURRENT_DIR}/${CASE_NAME}/settings/
cp ${ngsphyPATH}/data/settings/ngsphy.settings.case1.50x.txt
${CURRENT_DIR}/${CASE_NAME}/settings/
cp ${ngsphyPATH}/data/settings/ngsphy.settings.case1.100x.txt
```

```
${CURRENT_DIR}/${CASE_NAME}/settings/
cp ${ngsphyPATH}/data/settings/ngsphy.settings.case1.100x.rc.txt
${CURRENT_DIR}/${CASE_NAME}/settings/
cp ${ngsphyPATH}/data/indelible/control.case1.simphy.wrapper.txt
${CURRENT_DIR}/${CASE_NAME}/files/
cp ${ngsphyPATH}/data/trees/usecase.tree
${CURRENT_DIR}/${CASE_NAME}/files/original.tree
echo "Moving to the working directory"
cd ${CURRENT_DIR}/${CASE_NAME}
################################################################################
####
```

1. SimPhy sImulation 5 taxa/2 ind (haploids) per taxa

```
################################################################################
####
echo "Simulating gene/species trees with SimPhy"
simphy -rs 1 -rl f:1 -st f:100000 -sp f:10000 -sl f:4 -si f:4 -sb ln:-13.58,1.85 -su f:0.0000001 \
-hh ln:1.2,1 -hl ln:1.4,1 -hg f:400 \
-v 1 -o $SIMPHY_PROJECT_NAME -cs $MYRANDOMNUM \
-od 1 -op 1 -on 1 > $CURRENT_DIR/${CASE_NAME}/output/simphy.output
################################################################################
####
```

2. Sequence generator

```
################################################################################
####
echo "Downloading INDELIble_wrapper from SimPhy"
indelibleWrapperURL="https://raw.githubusercontent.com/adamallo/SimPhy/master/scripts/IND
ELIble_wrapper.pl"
if [[ $(uname -s) -eq "Linux" ]]; then
wget $indelibleWrapperURL
elif [[ $(uname -s) -eq "Darwin" ]]; then
curl -0 $indelibleWrapperURL
fi
mv INDELIble_wrapper.pl ${CURRENT_DIR}/${CASE_NAME}/src/
```

```
################################################################################
####
```

2.3 Modification of the gene tree

```
################################################################################
####
echo "Simulating DNA sequences from the gene/species trees above"
perl ${CURRENT_DIR}/${CASE_NAME}/src/INDELIble_wrapper.pl
$SIMPHY_PROJECT_NAME
"${CURRENT_DIR}/${CASE_NAME}/files/control.case1.simphy.wrapper.txt"
$MYRANDOMNUM 1 >> $CURRENT_DIR/${CASE_NAME}/output/wrapper.output
################################################################################
####
```

3. Allele count

```
################################################################################
####
gacmsa -i
${CURRENT_DIR}/${CASE_NAME}/${SIMPHY_PROJECT_NAME}/1/data_1_TRUE.fasta -o
$CURRENT_DIR/${CASE_NAME}/files/allele.count.txt
<1)
RR

################################################################################
####
```

4. Running NGSphy

```
################################################################################
####
echo "Running NGSphy - 100 replicates - Coverage 1x"
for replicate in $(seq 1 100); do ngsphy -s
${CURRENT_DIR}/${CASE_NAME}/settings/ngsphy.settings.case1.1x.txt; done
echo "Running NGSphy - 100 replicates - Coverage 5x"
for replicate in $(seq 1 100); do ngsphy -s
```

```
${CURRENT_DIR}/${CASE_NAME}/settings/ngsphy.settings.case1.5x.txt; done
echo "Running NGSphy - 100 replicates - Coverage 10x"
for replicate in $(seq 1 100); do ngsphy -s
${CURRENT_DIR}/${CASE_NAME}/settings/ngsphy.settings.case1.10x.txt; done
echo "Running NGSphy - 100 replicates - Coverage 50x"
for replicate in $(seq 1 100); do ngsphy -s
${CURRENT_DIR}/${CASE_NAME}/settings/ngsphy.settings.case1.50x.txt; done
echo "Running NGSphy - 100 replicates - Coverage 100x"
for replicate in $(seq 1 100); do ngsphy -s
${CURRENT_DIR}/${CASE_NAME}/settings/ngsphy.settings.case1.100x.txt; done
################################################################################
####
```

5. Reference selection

```
################################################################################
####
tail -n+3
${CURRENT_DIR}/${CASE_NAME}/${SIMPHY_PROJECT_NAME}/1/data_1_TRUE.fasta |
head -1 > ${CURRENT_DIR}/${CASE_NAME}/reference/reference.fasta
tail -n+3
${CURRENT_DIR}/${CASE_NAME}/${SIMPHY_PROJECT_NAME}/1/data_1_TRUE.fasta |
head -2 | tail -1 | tr -d " " >> ${CURRENT_DIR}/${CASE_NAME}/reference/reference.fasta
newRef=$(head -1 ${CURRENT_DIR}/${CASE_NAME}/reference/reference.fasta | tr "_" "," | tr
">" " " | tr -d " ")
echo "1,$newRef"> ${CURRENT_DIR}/${CASE_NAME}/files/my_reference_allele_file.case1.txt
################################################################################
####
```

6. Getting true variants

```
################################################################################
####
ngsphy -s ${CURRENT_DIR}/${CASE_NAME}/settings/ngsphy.settings.case1.100x.rc.txt
cp
${CURRENT_DIR}/${CASE_NAME}/NGSphy_case1_100x_RC/reads/no_error/REPLICATE_1/
data_1_1_NOERROR.vcf ${CURRENT_DIR}/${CASE_NAME}/files/true.vcf
vcftools --vcf ${CURRENT_DIR}/${CASE_NAME}/files/true.vcf --singletons --out
${CURRENT_DIR}/${CASE_NAME}/files/true
```

```
vcftools --vcf ${CURRENT_DIR}/${CASE_NAME}/files/true.vcf --extract-FORMAT-info GT --out
${CURRENT_DIR}/${CASE_NAME}/files/true
cat ${CURRENT_DIR}/${CASE_NAME}/files/true.vcf | grep -v "^#" | awk '{print $2}' >
${CURRENT_DIR}/${CASE_NAME}/files/true.variable.positions.txt
```

singletons info

```
echo -e "SINGLETON/DOUBLETONS\tSINGLETONS\tDOUBLETONS" >
${CURRENT_DIR}/${CASE_NAME}/files/true.singletons.summary
ALL=$(tail -n+2 ${CURRENT_DIR}/${CASE_NAME}/files/true.singletons | awk '{print $3}' | wc -
l)
D=$(tail -n+2 ${CURRENT_DIR}/${CASE_NAME}/files/true.singletons | awk '{print $3}' | grep D
| wc -l)
S=$(tail -n+2 ${CURRENT_DIR}/${CASE_NAME}/files/true.singletons | awk '{print $3}' | grep S
| wc -l)
echo -e "$ALL\t$S\t$D" >> ${CURRENT_DIR}/${CASE_NAME}/files/true.singletons.summary
################################################################################
####
```

2. Indexing reference

```
################################################################################
####
bwa index $referenceFile
samtools faidx $referenceFile
java -jar -Xmx4G $PICARD CreateSequenceDictionary REFERENCE=$referenceFile
OUTPUT="$CURRENT_DIR/${CASE_NAME}/reference/reference.dict"
################################################################################
####
```

3. Mapping

```
################################################################################
####
```

Organizational purposes

```
mkdir -p $CURRENT_DIR/${CASE_NAME}/mappings/1x
${CURRENT_DIR}/${CASE_NAME}/mappings/5x
${CURRENT_DIR}/${CASE_NAME}/mappings/10x
${CURRENT_DIR}/${CASE_NAME}/mappings/50x
${CURRENT_DIR}/${CASE_NAME}/mappings/100x
bashFile=$CURRENT_DIR/${CASE_NAME}/src/mappings.sh
rm $bashFile
for ngsphyoutput in $(find ${CURRENT_DIR}/${CASE_NAME}/output -mindepth 1 -maxdepth 1
-type d); do
coverageFolder=$(basename ${ngsphyoutput})
for ngsphyreplicate in $(ls ${ngsphyoutput}| sort); do
numInds=$(cat
${ngsphyoutput}/${ngsphyreplicate}/ind_labels/${SIMPHY_PROJECT_NAME}.1.individuals.csv
| wc -l)
let numInds=numInds-2 # This file has a header
mkdir -p "$CURRENT_DIR/${CASE_NAME}/mappings/${coverageFolder}/${ngsphyreplicate}/"
for ind in $(seq 0 $numInds); do
echo "$ngsphyreplicate/$ind"
infile="${ngsphyoutput}/$ngsphyreplicate/reads/REPLICATE_1/LOCUS_1/${SIMPHY_PROJEC
T_NAME}1_1_data${ind}"
outfile="$CURRENT_DIR/${CASE_NAME}/mappings/${coverageFolder}/${ngsphyreplicate}/${n
gsphyreplicate}${ind}.sam"
RGID="${ngsphyreplicate}-I${ind}"
machine="HiSeq2500"
echo "bwa mem -M -t 4 -R
\"@RG\tID:${RGID}\tSM:${RGID}\tPL:Illumina\tLB:${RGID}\tPU:${machine}\" ${referenceFile}
${infile}R1.fq ${infile}R2.fq > $outfile" >> $bashFile
done
done
done
bash $bashFile
##############################################################################
####
```

3.2 Mappings relaxed

```
##############################################################################
####
```

Organizational purposes

```
mkdir -p $CURRENT_DIR/${CASE_NAME}/mappings-relaxed/1x
${CURRENT_DIR}/${CASE_NAME}/mappings-relaxed/5x
${CURRENT_DIR}/${CASE_NAME}/mappings-relaxed/10x
${CURRENT_DIR}/${CASE_NAME}/mappings-relaxed/50x
${CURRENT_DIR}/${CASE_NAME}/mappings-relaxed/100x
bashFileRelaxed=$CURRENT_DIR/${CASE_NAME}/src/mappings.relaxed.sh
rm $bashFileRelaxed
for ngsphyoutput in $(find ${CURRENT_DIR}/${CASE_NAME}/output -mindepth 1 -maxdepth 1
-type d); do
coverageFolder=$(basename ${ngsphyoutput})
for ngsphyreplicate in $(ls ${ngsphyoutput}| sort); do
numInds=$(cat
${ngsphyoutput}/${ngsphyreplicate}/ind_labels/${SIMPHY_PROJECT_NAME}.1.individuals.csv
| wc -l)
let numInds=numInds-2 # This file has a header
mkdir -p "$CURRENT_DIR/${CASE_NAME}/mappings-
relaxed/${coverageFolder}/${ngsphyreplicate}/"
for ind in $(seq 0 $numInds); do
echo "$ngsphyreplicate/$ind"
infile="${ngsphyoutput}/$ngsphyreplicate/reads/REPLICATE_1/LOCUS_1/${SIMPHY_PROJEC
T_NAME}1_1_data${ind}" outfile="$CURRENT_DIR/${CASE_NAME}/mappings-
relaxed/${coverageFolder}/${ngsphyreplicate}/${ngsphyreplicate}${ind}.sam"
RGID="${ngsphyreplicate}-I${ind}"
machine="HiSeq2500"
echo "bwa mem -M -t 4 -B 3 -R
\"@RG\tID:${RGID}\tSM:${RGID}\tPL:Illumina\tLB:${RGID}\tPU:${machine}\" ${referenceFile}
${infile}R1.fq ${infile}R2.fq > $outfile" >> $bashFileRelaxed
done
done
done
bash $bashFileRelaxed
################################################################################
####
```

4 BAMMING

```
################################################################################
```

```
####
rm $CURRENT_DIR/${CASE_NAME}/src/bamming.sh
for samFile in $(find ${CURRENT_DIR}/${CASE_NAME}/mappings -type f | grep sam$); do
echo $samFile
outputDIR=$(dirname $samFile)
outputFILE="$(basename $samFile .sam).sorted.bam"
echo "samtools view -bSh $samFile | samtools sort - -f $outputDIR/${outputFILE} -@ 4" >>
$CURRENT_DIR/${CASE_NAME}/src/bamming.sh
echo "samtools index $outputDIR/$outputFILE" >>
$CURRENT_DIR/${CASE_NAME}/src/bamming.sh
echo "rm $samFile" >> $CURRENT_DIR/${CASE_NAME}/src/bamming.sh
done
bash $CURRENT_DIR/${CASE_NAME}/src/bamming.sh

################################################################################
####
```

4.2 BAMMING - relaxed

```
################################################################################
####
rm $CURRENT_DIR/${CASE_NAME}/src/bamming.relaxed.sh
for samFile in $(find ${CURRENT_DIR}/${CASE_NAME}/mappings-relaxed -type f | grep
sam$); do
echo $samFile
outputDIR=$(dirname $samFile)
outputFILE="$(basename $samFile .sam).sorted.bam"
echo "samtools view -bSh $samFile | samtools sort - -f $outputDIR/${outputFILE} -@ 4" >>
$CURRENT_DIR/${CASE_NAME}/src/bamming.relaxed.sh
echo "samtools index $outputDIR/$outputFILE" >>
$CURRENT_DIR/${CASE_NAME}/src/bamming.relaxed.sh
echo "rm $samFile" >> $CURRENT_DIR/${CASE_NAME}/src/bamming.relaxed.sh
done
bash $CURRENT_DIR/${CASE_NAME}/src/bamming.relaxed.sh

################################################################################
####
```

4. Mark Duplicates

```
################################################################################
####
summaryFile="$CURRENT_DIR/${CASE_NAME}/files/duplicates.summary.txt"
rm $summaryFile
for bamFile in $(find ${CURRENT_DIR}/${CASE_NAME}/mappings -type f | grep sorted.bam$);
do
coverageFolder=$(basename $(dirname $(dirname $bamFile)))
outputDIR=$(dirname $bamFile)
values=($(basename $bamFile | tr "_" " " | tr "." " " ))
indID=${values[-4]}
repID=1
if [[ ${#values} -eq 6 ]]; then
repID=${values[-3]}
fi
dedupOutput="$outputDIR/$(basename $bamFile .sorted.bam).dedup.bam"
metricsOutput="$outputDIR/$(basename $bamFile .sorted.bam).metrics.txt"
histogramOutput="$outputDIR/$(basename $bamFile .sorted.bam).histogram.txt"
echo "picard MarkDuplicates I=$bamFile O=$dedupOutput M=$metricsOutput"
java -jar -Xmx4G $HOME/apps/picard/picard.jar MarkDuplicates INPUT=$bamFile
OUTPUT=$dedupOutput METRICS_FILE=$metricsOutput
```

```
header=$(head -7 $metricsOutput | tail -n+7)
summaryInfo=$(head -8 $metricsOutput | tail -n+8)
if [[ ! -f $summaryFile ]]; then
    echo -e "COVERAGE\tREPLICATE\tINDIVIDUAL_ID\tNUM_MAPPED_READS_SAMTOOLS\tNUM_REC
fi
numReads=$(samtools view -c $bamFile)
numRecords=$(samtools view $bamFile | wc -l)
numMappedReads=$(samtools view -F 0x4 $bamFile | cut -f 1 | sort | uniq | wc -l)
echo -e "$coverageFolder\t$repID\t$indID\t$numMappedReads\t$numRecords\t$numReads\t
tail -n+11 $metricsOutput > $histogramOutput
samtools index $dedupOutput
```

```
done
################################################################################
####
```

Mark duplicates for relaxed info

```
summaryFileRelaxed="$CURRENT_DIR/${CASE_NAME}/files/duplicates.summary.relaxed.txt"
rm $summaryFileRelaxed
```

for bamFile in $(find ${CURRENT_DIR}/${CASE_NAME}/mappings -type f | grep sorted.bam$);
do
coverageFolder=$(basename $(dirname $(dirname $bamFile)))
outputDIR=$(dirname $bamFile)
values=($(basename $bamFile | tr "_" " " | tr "." " "))
indID=${values[-4]}
repID=1
if [[${#values} -eq 6]]; then
repID=${values[-3]}
fi
dedupOutput="$outputDIR/$(basename $bamFile .sorted.bam).dedup.bam"
metricsOutput="$outputDIR/$(basename $bamFile .sorted.bam).metrics.txt"
histogramOutput="$outputDIR/$(basename $bamFile .sorted.bam).histogram.txt"
echo "picard MarkDuplicates I=$bamFile O=$dedupOutput M=$metricsOutput"
java -jar -Xmx4G $HOME/apps/picard/picard.jar MarkDuplicates INPUT=$bamFile
OUTPUT=$dedupOutput METRICS_FILE=$metricsOutput

```
header=$(head -7 $metricsOutput | tail -n+7)
summaryInfo=$(head -8 $metricsOutput | tail -n+8)
if [[ ! -f $summaryFileRelaxed ]]; then
    echo -e "COVERAGE\tREPLICATE\tINDIVIDUAL_ID\tNUM_MAPPED_READS_SAMTOOLS\tNUM_REC
fi
numReads=$(samtools view -c $bamFile)
numRecords=$(samtools view $bamFile | wc -l)
numMappedReads=$(samtools view -F 0x4 $bamFile | cut -f 1 | sort | uniq | wc -l)
echo -e "$coverageFolder\t$repID\t$indID\t$numMappedReads\t$numRecords\t$numReads\t
tail -n+11 $metricsOutput > $histogramOutput
samtools index $dedupOutput
```

done

##
####

Coverage information per position for all datafiles in input order

##
####
find ${CURRENT_DIR}/${CASE_NAME}/mappings -type f | grep dedup.bam$ | grep 1x >

```
${CURRENT_DIR}/${CASE_NAME}/files/list.files.bam.coverage.1x.txt
find ${CURRENT_DIR}/${CASE_NAME}/mappings -type f | grep dedup.bam$ | grep 5x >
${CURRENT_DIR}/${CASE_NAME}/files/list.files.bam.coverage.5x.txt
find ${CURRENT_DIR}/${CASE_NAME}/mappings -type f | grep dedup.bam$ | grep 10x >
${CURRENT_DIR}/${CASE_NAME}/files/list.files.bam.coverage.10x.txt
find ${CURRENT_DIR}/${CASE_NAME}/mappings -type f | grep dedup.bam$ | grep 50x >
${CURRENT_DIR}/${CASE_NAME}/files/list.files.bam.coverage.50x.txt
find ${CURRENT_DIR}/${CASE_NAME}/mappings -type f | grep dedup.bam$ | grep 100x >
${CURRENT_DIR}/${CASE_NAME}/files/list.files.bam.coverage.100x.txt

samtools depth -f ${CURRENT_DIR}/${CASE_NAME}/files/list.files.bam.coverage.1x.txt >
files/coverage.distro.1x.txt
samtools depth -f ${CURRENT_DIR}/${CASE_NAME}/files/list.files.bam.coverage.5x.txt >
files/coverage.distro.5x.txt
samtools depth -f ${CURRENT_DIR}/${CASE_NAME}/files/list.files.bam.coverage.10x.txt >
files/coverage.distro.10x.txt
samtools depth -f ${CURRENT_DIR}/${CASE_NAME}/files/list.files.bam.coverage.50x.txt >
files/coverage.distro.50x.txt
samtools depth -f ${CURRENT_DIR}/${CASE_NAME}/files/list.files.bam.coverage.100x.txt >
files/coverage.distro.100x.txt

##########################################################################
####
```

Mappign Quality at diferent levels

samtools view $SEED | awk '{print $1\",\"$5}'

```
##########################################################################
####

##########################################################################
####
```

5. INDEL REALIGNMENT

##
####
for bamFile in $(find ${CURRENT_DIR}/${CASE_NAME}/mappings -type f | grep dedup.bam$);
do
echo "$bamFile"
outputDIR=$(dirname $bamFile)
mkdir -p $outputDIR/target/
targetOutput="$outputDIR/target/$(basename $bamFile .dedup.bam).target.intervals.list"
realignedBam="$outputDIR/$(basename $bamFile .dedup.bam).realigned.bam"
java -jar -Xmx4g $GATK \
-T RealignerTargetCreator \
-R $referenceFile \
-nt 4
-I $bamFile \
-o $targetOutput

```
java -jar -Xmx4g $GATK \
    -T IndelRealigner \
    -R $referenceFile \
    -nt 4
    -I $bamFile \
    -targetIntervals $targetOutput \
    -o $realignedBam
samtools index $realignedBam
```

done

stopped here

$ find ${CURRENT_DIR}/${CASE_NAME}/mappings -type f | grep dedup.bam$ | grep -n "/home/merly/test/usecase3/mappings/100x/NGSphy_case1_100x_71/NGSphy

_case1_100x_71_5.dedup.bam"

##
####

6. BASE QUALITY RECALIBRATION: need true variants

https://software.broadinstitute.org/gatk/documentation/article?id=2801

##
####

7. GATK - single call joint genotyping

##
####

```
for bamFile in $(find ${CURRENT_DIR}/${CASE_NAME}/mappings -type f | grep
realigned.bam$ | grep NGSphy_case1_100x/); do
echo "$bamFile"
outputDIR=$(dirname $bamFile)
mkdir -p $outputDIR/vcf-singlevc-joint-gt/
OUTPUTVCF="$outputDIR/vcf-singlevc-joint-gt/$(basename $bamFile .realigned.bam).g.vcf"
{ time java -jar -Xmx4g $GATK \
-T HaplotypeCaller \
-R $referenceFile \
-I $bamFile \
-ERC GVCF \
-o $OUTPUTVCF; } 2>>
${CURRENT_DIR}/${CASE_NAME}/files/time.gatk.HaplotypeCaller.g.vcf.txt
done

coverages=( "1x" "5x" "10x" "50x" "100x")
for coverageLevel in ${coverages[*]}; do
```

```
coverageFolder="${CURRENT_DIR}/${CASE_NAME}/mappings/$coverageLevel"
for replicate in $(ls $coverageFolder); do
echo $replicate
individuals=""
replicateFolder="${CURRENT_DIR}/${CASE_NAME}/mappings/$coverageLevel/$replicate"
for indFile in $(find ${CURRENT_DIR}/${CASE_NAME}/mappings/$coverageLevel/$replicate -
type f | grep .g.vcf$); do
individuals+=" -V $indFile"
done
OUTPUTVCF="$replicateFolder/vcf-singlevc-joint-gt/$replicate.vcf"
{ time java -jar -Xmx4g $GATK \
-T GenotypeGVCFs \
-R $referenceFile \
-newQual \
$individuals \
-o $OUTPUTVCF ;} 2>> ${CURRENT_DIR}/${CASE_NAME}/files/time.gatk.genotypeGVCF.txt
done
done
##############################################################################
####
```

8 - Count discovered variants

```
##############################################################################
####
mkdir ${CURRENT_DIR}/${CASE_NAME}/varsites/
numVariantsSummary="${CURRENT_DIR}/${CASE_NAME}/files/numvariants.summary.txt"
echo -e "COVERAGE\tREPLICATE\tNUM_VARIANTS" >
${CURRENT_DIR}/${CASE_NAME}/files/numvariants.summary.txt
for coverageLevel in ${coverages[]}; do for vcffile in $(find
${CURRENT_DIR}/${CASE_NAME}/mappings/$coverageLevel -name "*.vcf"| grep -v g.vcf |
grep vcf-singlevc-joint-gt); do
base=$(basename $vcffile)
repID=$(echo $base |tr "_" " " | tr "." " " | awk '{print $4}' )
if [[ repID -eq "vcf" ]]; then
repID=1
fi
numVariants=$(cat $vcffile | grep -v "^#" |wc -l)
mkdir -p ${CURRENT_DIR}/${CASE_NAME}/varsites/$coverageLevel/
cat $vcffile | grep -v "^#" | awk '{print $2}' >
${CURRENT_DIR}/${CASE_NAME}/varsites/$coverageLevel/${base}.varsites
```

```
echo -e "$coverageLevel\t$repID\t$numVariants" >> $numVariantsSummary
done
done
################################################################################
####
```

9. get information per coverage on the varibale sites

```
################################################################################
####
for coverageLevel in ${coverages[]}; do find
${CURRENT_DIR}/${CASE_NAME}/varsites/$coverageLevel -name ".varsites" >
${CURRENT_DIR}/${CASE_NAME}/files/varsites.$coverageLevel.files
done

################################################################################
####
```

10.DECOMPOSING MNPs

```
################################################################################
####
for coverageLevel in ${coverages[]}; do for vcffile in $(find
${CURRENT_DIR}/${CASE_NAME}/mappings/$coverageLevel -name ".vcf"| grep -v g.vcf |
grep vcf-singlevc-joint-gt); do
base=$(basename $vcffile)
dir=$(dirname $vcffile)
newname="$dir/$(basename $base .vcf).decomposed.vcf"
echo "vt decompose_blocksub $vcffile -o $newname" >>
${CURRENT_DIR}/${CASE_NAME}/src/decomposed.vcf.sh
done
done
```

```
bash ${CURRENT_DIR}/${CASE_NAME}/src/decomposed.vcf.sh

================================================================================

decomposedNumVariants="${CURRENT_DIR}/${CASE_NAME}/files/numvariants.decomposed
.summary.txt"
echo -e "COVERAGE\tREPLICATE\tNUM_VARIANTS" > $decomposedNumVariants
for coverageLevel in ${coverages[*]}; do mkdir -p
${CURRENT_DIR}/${CASE_NAME}/decomposed-varsites/$coverageLevel/ for vcffile in $(find
${CURRENT_DIR}/${CASE_NAME}/mappings/$coverageLevel -type f -name "*.vcf"| grep -v
g.vcf | grep vcf-singlevc-joint-gt | grep decomposed.vcf); do
base=$(basename $vcffile)
repID=$(echo $base |tr "_" " " | tr "." " " | awk '{print $4}' )
if [[ repID -eq "decomposed" ]]; then
repID=1
fi
numVariants=$(cat $vcffile | grep -v "^#" | wc -l)
echo -e "$coverageLevel\t$repID\t$numVariants"
cat $vcffile | grep -v "^#" | awk '{print $2}' > ${CURRENT_DIR}/${CASE_NAME}/decomposed-
varsites/$coverageLevel/${base}.varsites
echo -e "$coverageLevel\t$repID\t$numVariants" >> $decomposedNumVariants
done
done
for coverageLevel in ${coverages[*]}; do
find ${CURRENT_DIR}/${CASE_NAME}/decomposed-varsites/$coverageLevel -type f >
${CURRENT_DIR}/${CASE_NAME}/files/varsites.decomposed.$coverageLevel.files
done

################################################################################
####
```

SINGLETONS

```
################################################################################
####
rm src/bash.singletons.sh
for coverageLevel in ${coverages[*]}; do mkdir -p
```

```
*${CURRENT_DIR}/${CASE_NAME}/singletons/$coverageLevel/ for vcffile in $(find*
*${CURRENT_DIR}/${CASE_NAME}/mappings/$coverageLevel -type f -name "*.vcf"| grep -v
g.vcf | grep -v decomposed); do
echo "vcftools --vcf $vcffile --singletons --out
${CURRENT_DIR}/${CASE_NAME}/singletons/$coverageLevel/$(basename $vcffile vcf)full"
>> src/bash.singletons.sh
done
done
bash src/bash.singletons.sh
rm src/bash.singletons.decomposed.sh
for coverageLevel in ${coverages[*]}; do mkdir -p
*${CURRENT_DIR}/${CASE_NAME}/singletons/$coverageLevel/ for vcffile in $(find*
*${CURRENT_DIR}/${CASE_NAME}/mappings/$coverageLevel -type f -name "*.vcf"| grep
decomposed); do
echo "vcftools --vcf $vcffile --singletons --out
${CURRENT_DIR}/${CASE_NAME}/singletons/$coverageLevel/$(basename $vcffile
vcf.decomposed.vcf)decomposed" >> src/bash.singletons.decomposed.sh
done
done
bash src/bash.singletons.decomposed.sh

singletonsNums="${CURRENT_DIR}/${CASE_NAME}/files/numvariants.singletons.txt"
echo -e "COVERAGE\tREPLICATE\tNUM_VARIANTS" > $singletonsNums
for coverageLevel in ${coverages[*]}; do
*${CURRENT_DIR}/${CASE_NAME}/singletons/$coverageLevel/ for vcffile in $(find*
*${CURRENT_DIR}/${CASE_NAME}/singletons/$coverageLevel -type f -name "*.singletons" |
grep full ); do
base=$(basename $vcffile)
repID=$(echo $base |tr "_" " " | tr "." " " | awk '{print $4}' )
if [[ repID -eq "vcf" ]]; then
repID=1
fi
echo $repID
num=$(cat "${CURRENT_DIR}/${CASE_NAME}/singletons/$coverageLevel/$(basename
$vcffile)" | tail -n+2 | awk '{print $2}' | sort | uniq | wc -l)
echo -e "$coverageLevel\t$repID\t$num">> $singletonsNums
done
done

singletonsDNums="${CURRENT_DIR}/${CASE_NAME}/files/numvariants.singletons.decompos
ed.summary.txt"
echo -e "COVERAGE\tREPLICATE\tNUM_VARIANTS" > $singletonsDNums
```

```
for coverageLevel in ${coverages[]}; do
${CURRENT_DIR}/${CASE_NAME}/singletons/$coverageLevel/ for vcffile in $(find
${CURRENT_DIR}/${CASE_NAME}/singletons/$coverageLevel -type f -name "*.singletons" |
grep decomposed ); do
base=$(basename $vcffile)
repID=$(echo $base |tr "_" " " | tr "." " " | awk '{print $4}' )
if [[ repID -eq "vcf" ]]; then
repID=1
fi
echo $repID
num=$(cat "${CURRENT_DIR}/${CASE_NAME}/singletons/$coverageLevel/$(basename
$vcffile)" | tail -n+2 | awk '{print $2}' | sort | uniq | wc -l)
echo -e "$coverageLevel\t$repID\t$num">> $singletonsDNums
done
done

##########################################################################
####
```

Information extraction from the VCFs - GT/DP

```
##########################################################################
####
mkdir ${CURRENT_DIR}/${CASE_NAME}/GT
mkdir ${CURRENT_DIR}/${CASE_NAME}/DP

for coverageLevel in ${coverages[]}; do for vcffile in $(find
${CURRENT_DIR}/${CASE_NAME}/mappings/$coverageLevel -name "*.vcf" | grep -v g.vcf |
grep decomposed ); do
base=$(basename $vcffile)
repID=$(echo $base |tr "_" " " | tr "." " " | awk '{print $4}' )
if [[ repID -eq "vcf" ]]; then
repID=1
fi
mkdir -p ${CURRENT_DIR}/${CASE_NAME}/GT/$coverageLevel/
mkdir -p ${CURRENT_DIR}/${CASE_NAME}/DP/$coverageLevel/
vcftools --vcf $vcffile --extract-FORMAT-info GT --out
${CURRENT_DIR}/${CASE_NAME}/GT/$coverageLevel/${base}
vcftools --vcf $vcffile --extract-FORMAT-info DP --out
```

```
${CURRENT_DIR}/${CASE_NAME}/DP/$coverageLevel/${base}
done
done

################################################################################
####
cat ${CURRENT_DIR}/${CASE_NAME}/files/true.singletons | grep Ind0 >
${CURRENT_DIR}/${CASE_NAME}/files/singletons.Ind0.txt
cat ${CURRENT_DIR}/${CASE_NAME}/files/true.singletons | grep Ind1 >
${CURRENT_DIR}/${CASE_NAME}/files/singletons.Ind1.txt
cat ${CURRENT_DIR}/${CASE_NAME}/files/true.singletons | grep Ind2 >
${CURRENT_DIR}/${CASE_NAME}/files/singletons.Ind2.txt
cat ${CURRENT_DIR}/${CASE_NAME}/files/true.singletons | grep Ind3 >
${CURRENT_DIR}/${CASE_NAME}/files/singletons.Ind3.txt
cat ${CURRENT_DIR}/${CASE_NAME}/files/true.singletons | grep Ind4 >
${CURRENT_DIR}/${CASE_NAME}/files/singletons.Ind4.txt
cat ${CURRENT_DIR}/${CASE_NAME}/files/true.singletons | grep Ind5 >
${CURRENT_DIR}/${CASE_NAME}/files/singletons.Ind5.txt
cat ${CURRENT_DIR}/${CASE_NAME}/files/true.singletons | grep Ind6 >
${CURRENT_DIR}/${CASE_NAME}/files/singletons.Ind6.txt
cat ${CURRENT_DIR}/${CASE_NAME}/files/true.singletons | grep Ind7 >
${CURRENT_DIR}/${CASE_NAME}/files/singletons.Ind7.txt
```