**1. Alignment for each sample**

Reference: ARS-UI\_Ramb\_v2.0 (<https://www.ncbi.nlm.nih.gov/assembly/GCF_016772045.1>)

*bowtie2 -p 32 -x Reference\_Genome\_index/ram2\_genome -q ${sample}.fastq 2 >> mapping.log | samtools sort - -o ${sample}.RAM2.sorted.bam*

**2. Add sample information into the alignment Bam file**

*gatk AddOrReplaceReadGroups \*

*-I=${sample}.RAM2.sorted.bam \*

*-O=${sample}.sorted.Add.bam \*

*--RGID=$sample --RGLB=lib1 --RGPL=Ion\_Torrent --RGPU=unit1 --RGSM=$sample*

*samtools index ${sample}.sorted.Add.bam*

*samtools flagstat ${sample}.sorted.Add.bam*

**3. Variants calling by GATK**

3.1 Generate gvcf file of each sample by HaplotypeCaller

*refdir=/mnt/ceph/bmurdoch/Shang/data/refer2\_Ramb2*

*bamfile=${sample}.sorted.Add.bam*

*gatk --java-options "-Xmx320g" HaplotypeCaller \*

*-R $refdir/ram2\_all.fa \*

*-I $bamfile \*

*-O ${sample}\_RAM2.g.vcf.gz \*

*-ERC GVCF \*

*--native-pair-hmm-threads 32*

3.2 Generate vcf file of each sample by GenotypeGVCFs

*gatk --java-options "-Xmx320g" GenomicsDBImport \*

*--genomicsdb-workspace-path database/${sample}\_database \*

*-L GATK/interval.list \*

*--sample-name-map inputmap/input\_${sample}.map \*

*--tmp-dir=database/tmpdir*

*gatk --java-options "-Xmx320g" GenotypeGVCFs \*

*-R $refdir/ram2\_all.fa \*

*-V gendb://database/${sample}\_database \*

*-stand-call-conf 10 \*

*-O VCF/GATK\_${sample}\_raw.vcf*

**4. Variants calling by Freebayes**

*freebayes-parallel $refdir/ram2\_all.fa.100m.regions 32 -f $refdir/ram2\_all.fa ${sample}.sorted.Add.bam*

*>FB\_${sample}\_raw.vcf*

**5. Filter by Depth and Quality of each sample**

GATK:

*vcftools --vcf $GATK\_${sample}\_raw\_vcf --minQ 20 --min-meanDP 5 --out GATK\_${sample}\_Q20\_DP5 --recode --recode-INFO-all*

Freebayes:

*vcftools --vcf $FB\_${sample}\_raw\_vcf --minQ 20 --min-meanDP 5 --out FB\_${sample}\_Q20\_DP5 --recode --recode-INFO-all*

**6. Generate SNP and Indel vcf of each sample**

GATK SNP:

*vcftools --vcf GATK\_${sample}\_Q20\_DP5.recode.vcf --remove-indels --out ${sample}\_Q20\_DP5\_SNP\_GK --recode --recode-INFO-all*

GATK Indel:

*vcftools --vcf GATK\_${sample}\_Q20\_DP5.recode.vcf --keep-only-indels --out ${sample}\_Q20\_DP5\_INDEL\_GK --recode --recode-INFO-all*

Freebayes SNP:

*vcftools --vcf FB\_${sample}\_Q20\_DP5.recode.vcf --remove-indels --out ${sample}\_Q20\_DP5\_SNP\_FB --recode --recode-INFO-all*

Freebayes Indel:

*vcftools --vcf FB\_${sample}\_Q20\_DP5.recode.vcf --keep-only-indels --out ${sample}\_Q20\_DP5\_INDEL\_FB --recode --recode-INFO-all*

**7. Calculate the sequencing quality of SNP from each sample in GATK and Freebayes**

7.1 SNP position

*awk '/^NC/ {print $1"\t"$2"\t"$2+1}' ${sample}\_Q20\_DP5\_SNP\_FB |sort -k1,1 -k2,2n > ${sample}\_pos\_fb.bed*

7.2 sequencing quality in the SNP position

*java -jar sam2tsv.jar -R $refdir/ram2\_all.fa ${sample}.sorted.Add.bam --regions ${sample}\_pos\_fb.bed > ${sample}\_pos\_quality\_fb.txt*

*awk '{if($10=="M") {print $4"\t"$8"\t"$8+1"\t"$9"\t"$6"\t"$7"\t"$10;tmp=$8;bq=$7}*

*if($10=="D") {print $4"\t"$8-1"\t"$8"\t"$9"\t"$6"\t"bq"\t"$10;tmp=$8}*

*if($10=="I") {print $4"\t"tmp"\t"tmp+1"\t"$9"\t"$6"\t"$7"\t"$10;bq=$7}} ' ${sample}\_pos\_quality\_fb.txt > ${allposfile}.bed*

*bedtools intersect -loj -a ${sample}\_pos\_fb.bed -b ${allposfile}.bed > ${sample}\_quality\_fb.info*

7.3 calculation poisson probability

*allpos=read.csv(${sample}\_quality\_fb.info, head=F, sep="\t", stringsAsFactors=FALSE, quote = "")*

*####pos\_name for each variant*

*posname=paste(allpos[,1],allpos[,2],sep=":")*

*nameallpos=cbind(allpos,posname)*

*markerpos=unique(posname) #####variants position*

*markerposp=matrix(,length(markerpos),7)*

*#### Chromosome, position, totalR, RefR, AltR, SequencingError, Probability*

*for (i in 1:length(markerpos)) ###Total variants number*

*{*

*tmp=nameallpos[nameallpos[,11]==markerpos[i],]*

*refbaseq=tmp[tmp[,7]==tmp[,8],9] ###ref quality*

*altbaseq=tmp[tmp[,7]!=tmp[,8],9] ###alt quality*

*yesvalue=numeric()*

*if(length(altbaseq)>0)*

*{for (k in 1:length(altbaseq))*

*{yesvalue[k]=10^(-(utf8ToInt(altbaseq[k])-33)/10)} ####sequencing error*

*markerposp[i,6]=mean(yesvalue)*

*lambda=(length(refbaseq)+length(altbaseq))\*mean(yesvalue)*

*poisP=ppois(length(altbaseq), lambda)*

*markerposp[i,7]=poisP*

*}*

*else*

*{markerposp[i,6]="\_"*

*markerposp[i,7]=0}*

*markerposp[i,1:2]=strsplit(markerpos[i],":")[[1]]*

*markerposp[i,3]=length(refbaseq)+length(altbaseq)*

*markerposp[i,4]=length(refbaseq)*

*markerposp[i,5]=length(altbaseq)*

*}*

*write.table(markerposp,paste0(samplename,".Pois.result"),quote=F,col.names=F,row.names=T)*

**8. Construct rHID database**

8.1 merge 5,061 samples’ vcf files

*bcftools merge --merge all Freebayes/FB\*Q20\_DP5.recode.vcf.gz -o Freebayes\_combined\_raw.vcf*

*bcftools merge --merge all GATK/GATK \*\_Q20\_DP5.recode.vcf.gz -o GATK\_combined\_raw.vcf*

8.2 VQSR

SNP:

*gatk --java-options "-Xmx320g" VariantRecalibrator \*

*-R $refdir/ram2\_all.fa \*

*-V $GVCF \*

*--resource:GFoverlap,known=false,training=true,truth=true,prior=10.0 05Gatk\_overlap.vcf \*

*-an DP -an QD -an FS -an SOR -an MQ -an MQRankSum -an ReadPosRankSum \*

*-mode SNP \*

*-tranche 100.0 -tranche 99.9 -tranche 99.0 -tranche 90.0 \*

*-O recalibrate\_SNP.recal \*

*--tranches-file recalibrate\_SNP.tranches \*

*--rscript-file recalibrate\_SNP\_plots.R*

*gatk ApplyVQSR \*

*-R $refdir/ram2\_all.fa \*

*-V $GVCF \*

*-ts-filter-level 99.0 \*

*-mode SNP \*

*--tranches-file recalibrate\_SNP.tranches \*

*--recal-file recalibrate\_SNP.recal \*

*-O GATK\_recalibrated\_snps\_raw\_indels.vcf*

Indel:

*gatk --java-options "-Xmx320g" VariantRecalibrator \*

*-R $refdir/ram2\_all.fa \*

*-V GATK\_recalibrated\_snps\_raw\_indels.vcf \*

*-an DP -an QD -an FS -an SOR -an MQ -an MQRankSum -an ReadPosRankSum \*

*-mode INDEL \*

*-tranche 100.0 -tranche 99.9 -tranche 99.0 -tranche 90.0 \*

*--max-gaussians 4 \*

*-O recalibrate\_INDEL.recal \*

*--tranches-file recalibrate\_INDEL.tranches \*

*--rscriptFile recalibrate\_INDEL\_plots.R*

*gatk ApplyVQSR \*

*-R $refdir/ram2\_all.fa \*

*-V GATK\_recalibrated\_snps\_raw\_indels.vcf \*

*-ts-filter-level 99.0 \*

*-mode INDEL \*

*--tranches-file recalibrate\_INDEL.tranches \*

*--recal-file recalibrate\_INDEL.recal \*

*-O GATK\_recalibrated\_variants.vcf*

8.3 generate rHID

*awk '/^NC/&&$6>=1000 {print $1"\t"$2"\t"$2+1}' FB\_recalibrated\_variants.vcf > 01fb\_HQ.bed*

*awk '/^NC/&&$6>=1000 {print $1"\t"$2"\t"$2+1}' GATK\_recalibrated\_variants.vcf > 01gk\_HQ.bed*

*awk '/^NC/ {n=0; for(i=10;i<=NF;i++) { if(index($i,"./.")==0&&index($i,"0/0")==0&&index($i,"0|0")==0) n++} ; if(n>1) {print $1"\t"$2"\t"$2+1"\t"n}}' FB\_recalibrated\_variants.vcf > 00fb\_2sample.bed*

*awk '/^NC/ {n=0; for(i=10;i<=NF;i++) { if(index($i,"./.")==0&&index($i,"0/0")==0&&index($i,"0|0")==0) n++} ; if(n>1) {print $1"\t"$2"\t"$2+1"\t"n}}' GATK\_recalibrated\_variants.vcf > 00gk\_2sample.bed*

*bedtools intersect -loj -a 00fb\_2sample.bed -b 00gk\_2sample.bed |awk '$5!="." {print $1"\t"$2"\t"$3}' > 01fb\_gk\_positive.bed*

*cat 01fb\_HQ.bed 01gk\_HQ.bed 01fb\_gk\_positive.bed |sort -k1,1 -k2,2n |uniq >rHID.bed*

**9. Identify the variants in rHID of each sample**

9.1 *Integrate quality*

*#SNP:*

*awk '{print $2"\t"$3"\t"$3+1"\t"$0}' ${sample}\_gk.Pois.result |sort -k1,1 -k2,2n > 01Pois\_bed/${sample}.bed*

*awk '/^NC/ {print $1"\t"$2"\t"$2+1"\t"$0}' GATK\_${sample}\_Q20\_DP5.recode.vcf |sort -k1,1 -k2,2n > 02GK/02Vcf\_bed/${sample}\_vcf.bed*

*bedtools intersect -loj -a 01Pois\_bed/${sample}.bed -b ${sample}\_vcf.bed |awk '$12!="." {print $1"\t"$2"\t"$3"\t"$7"\t"$8"\t"$9"\t"$11"\t"$20}' > ${sample}\_pois\_qual.bed*

#Indel:

awk '/^NC/ {print $1"\t"$2"\t"$2+1"\t"$6}' ${sample}\_Q20\_DP5\_INDEL\_GK.recode.vcf \

|sort -k1,1 -k2,2n > ${sample}\_vcf.bed

9.2 Integrate rHID information

SNP:

bedtools intersect -loj -a ${sample}\_pois\_qual.bed -b rHID.bed |awk '{ tmp=$1; for(i=2;i<=9;i++){tmp=tmp"\t"$i}; if($10==".") {print tmp"\tNo"}; if($10!=".") {print tmp"\tPos"}}' > ${sample}\_positive\_r\_gk\_SNP.txt

Indel:

bedtools intersect -loj -a ${sample}\_vcf.bed -b rHID.bed |awk '{ tmp=$1; for(i=2;i<=5;i++){tmp=tmp"\t"$i}; if($6==".") {print tmp"\tNo"}; if($6!=".") {print tmp"\tPos"}}' > ${sample}\_positive\_r\_gk\_INDEL.txt

**10. FDR of variants for each sample**

For SNP:

*allpos=read.csv(${sample}\_positive\_r\_gk.txt, head=F,sep="\t", stringsAsFactors=FALSE, quote = "")*

*all8=allpos[order(allpos[,8],decreasing = T),]*

*all7=all8[order(all8[,7],decreasing = T),]*

*Fdr=numeric()*

*for (i in 1:dim(all7)[1])*

*{*

*tmp=all7[1:i,]*

*posn=length(tmp[,8])*

*negn=length(tmp[tmp[,10]=="No",8])*

*Fdr[i]=negn/posn*

*if(Fdr[i]>0.01||(all7[i,7]-1<0&&all7[i,10]=="No")) {break}*

*}*

*Posit=all7[1:(i-1),]*

*Negat=all7[i:dim(all7)[1],]*

*if(type=="GK")*

*{Posit1=Posit[Posit[,8]>350|Posit[,9]>10,]*

*Negat1=Posit[Posit[,8]<=350&Posit[,9]<=10,]*

*Posit2=Negat[Negat[,7]>0.99&Negat[,8]>350&Negat[,9]>10&Negat[,10]=="Pos",]*

*Negat2=Negat[Negat[,7]<=0.99|Negat[,8]<=350|Negat[,9]<=10|Negat[,10]=="No",]*

*}*

*Positive=rbind(Posit1, Posit2)*

*Negative=rbind(Negat1, Negat2)*

*print(paste("Before: ", dim(all7)[1], "After: ",dim(Positive)[1]))*

*write.table(Positive,paste(samplename,"\_gk.Positive\_SNP",sep=""),quote=F,col.names=F)*

*write.table(Negative,paste(samplename,"\_gk.Negative\_SNP",sep=""),quote=F,col.names=F)*

For Indel:

*allpos=read.csv(${sample}\_positive\_r\_gk\_INDEL.txt,head=F,sep="\t",stringsAsFactors=FALSE,quote = "")*

*all7=allpos[order(allpos[,4],decreasing = T),]*

*for (i in 1:dim(all7)[1])*

*{*

*tmp=all7[1:i,]*

*posn=length(tmp[,4])*

*negn=length(tmp[tmp[,6]=="No",4])*

*Fdr=negn/posn*

*#print(i)*

*#print(Fdr)*

*if(Fdr>0.01||all7[i,4]<100) {break}*

*}*

*Posit=all7[1:(i-1),]*

*Negat=all7[i:dim(all7)[1],]*

*cutoff=150*

*Posit1=Posit[Posit[,4]>cutoff&Posit[,5]>10,]*

*Negat1=Posit[Posit[,4]<=cutoff|Posit[,5]<=10,]*

*Posit2=Negat[Negat[,4]>cutoff&Negat[,5]>10&Negat[,6]=="Pos",]*

*Negat2=Negat[Negat[,4]<=cutoff|Negat[,5]<=10|Negat[,6]=="No",]*

*Positive=rbind(Posit1, Posit2)*

*Negative=rbind(Negat1, Negat2)*

*print(paste("Before: ", dim(all7)[1], "After: ",dim(Positive)[1]))*

*write.table(Positive,paste(samplename,"\_gk.Positive\_Indel",sep=""),quote=F,col.names=F)*

*write.table(Negative,paste(samplename,"\_gk.Negative\_Indel",sep=""),quote=F,col.names=F)*

**11 Generate final variant list for all samples**

#SNP:

*awk '{print $2"\t"$3"\t"$3+1}' ${sample}\_gk.Positive\_SNP > ${sample}\_gk.venn*

*cat \*gk.venn |sort -k1,1 -k2,2n |uniq >all\_SNP\_GK.bed*

*cat all\_SNP\_GK.bed all\_SNP\_FB.bed |sort -k1,1 -k2,2n |uniq > SNP\_GK\_FB.bed*

*bedtools intersect -loj -a SNP\_GK\_FB.bed -b all\_SNP\_GK.bed |bedtools intersect -loj -a - -b all\_SNP\_FB.bed >SNP\_GK\_FB\_raw.bed*

*awk '{if($4!="."&&$7!=".") {print $1"\t"$2"\t"$3"\tGK-FB"}*

*if($4!="."&&$7==".") {print $1"\t"$2"\t"$3"\tGK"}*

*if($4=="."&&$7!=".") {print $1"\t"$2"\t"$3"\tFB"}}' SNP\_GK\_FB\_raw.bed > SNP\_GK\_FB\_final.bed*

#Indel:

awk '{print $2"\t"$3"\t"$3+1}' ${sample}\_gk.Positive\_Indel > ${sample}\_gk.venn

*cat \*gk.venn |sort -k1,1 -k2,2n |uniq >INDEL\_GK.bed*

cat INDEL\_GK.bed INDEL\_FB.bed |sort -k1,1 -k2,2n |uniq > INDEL\_GK\_FB.bed

bedtools intersect -loj -a INDEL\_GK\_FB.bed -b INDEL\_GK.bed |bedtools intersect -loj -a - -b INDEL\_FB.bed |awk '{if($4!="."&&$7!=".") {print $1"\t"$2"\t"$3"\tGK-FB"}

if($4!="."&&$7==".") {print $1"\t"$2"\t"$3"\tGK"}

if($4=="."&&$7!=".") {print $1"\t"$2"\t"$3"\tFB"}}' > INDEL\_GK\_FB\_final.bed

#merge SNP and Indel

cat SNP\_GK\_FB.bed INDEL\_GK\_FB.bed |sort -k1,1 -k2,2n |uniq > SNP\_INDEL.bed

bedtools intersect -loj -a SNP\_INDEL.bed -b SNP\_GK\_FB\_final.bed |bedtools intersect -loj -a - -b INDEL\_GK\_FB\_final.bed |awk '{if($4!="."&&$8!=".") {print $1"\t"$2"\t"$3"\tSNP:INDEL|"$7":"$11}

if($4!="."&&$8==".") {print $1"\t"$2"\t"$3"\tSNP|"$7}

if($4=="."&&$8!=".") {print $1"\t"$2"\t"$3"\tINDEL|"$11}}' > SNP\_INDEL\_GK\_FB\_final.bed

#Final vcf file after VQSR in GATK

awk '{if($4!="SNP|FB"&&$4!="INDEL|FB"&&$4!="SNP:INDEL|FB:FB"&&$4!="SNP:INDEL|FB:GK"&&$4!="SNP:INDEL|FB:GK-FB") {print $0}}' SNP\_INDEL\_GK\_FB\_final.bed > SNP\_INDEL\_GK.bed

awk '/^NC/ {print $1"\t"$2"\t"$2+1"\t"$0}' *GATK\_recalibrated\_variants.vcf* > GK\_v.bed

bedtools intersect -loj -a SNP\_INDEL\_GK.bed -b GK\_v.bed > SNP\_INDEL\_GK\_final.vcf

#list genotype and reads number

awk '{nn=0; tmp1=$1"\t"$2"\t"$4; tmp2=$11"\t"$12"\t"$13

for(i=17;i<=NF;i++)

{ split($i,a,":");

if(a[1]=="./."){tmp2=tmp2"\t."}

if(a[1]!="./."){tmp2=tmp2"\t"a[1]":"a[3];nn=nn+1}

}

{print tmp1"\t"nn"\t"tmp2}}' SNP\_INDEL\_GK\_final.vcf > SNP\_INDEL\_GK.final

#merge GATK and Freebayes

cat SNP\_INDEL\_GK.final SNP\_INDEL\_FB.final |sort -k1,1 -k2,2n > SNP\_INDEL\_SN.final

#Annotation

awk '/^NC/ {print $1"\t"$2"\t"$2+1"\t"$0}' SNP\_INDEL\_SN.final |sort -k1,1 -k2,2n |uniq > AllVarVCF.bed

bedtools intersect -loj -a AllVarVCF.bed -b Pos\_gene.bed> AllVar.final

######WGS

GATK joint calling

gatk --java-options "-Xmx256g" HaplotypeCaller \

-R $refdir/ram2\_all.fa \

-I $bamfile1 \

-O ${sample}\_new\_RAM2.g.vcf.gz \

-ERC GVCF \

--native-pair-hmm-threads 48

gatk --java-options "-Xmx320g" GenomicsDBImport \

--genomicsdb-workspace-path 02GenomicDB\_chr1/WGS5\_database\_chr1 \

-L NC\_056054.1 \

--sample-name-map WGS\_input\_5.map \

--tmp-dir 02GenomicDB\_chr1/tmp \

--batch-size 50 \

--reader-threads 40 \

--genomicsdb-shared-posixfs-optimizations true

gatk --java-options "-Xmx320g" GenotypeGVCFs \

-R $refdir/ram2\_all.fa \

-V gendb://02GenomicsDB/02GenomicDB\_chr1/WGS5\_database\_chr1 \

-stand-call-conf 10 \

-L NC\_056054.1 \

--genomicsdb-shared-posixfs-optimizations true \

-O 03Joint\_callVCF\_chr1/GATK\_joint\_raw10\_chr1.vcf

VQSR

gatk --java-options "-Xmx320g" VariantRecalibrator \

-R $refdir/ram2\_all.fa \

-V $outdir/03Joint\_callVCF\_chr1/GATK\_joint\_raw10\_chr1.vcf \

--resource:GFoverlap,known=false,training=true,truth=true,prior=10.0 05Gatk\_overlap.vcf \

-an DP -an QD -an FS -an SOR -an MQ -an MQRankSum -an ReadPosRankSum \

-mode SNP \

-tranche 100.0 -tranche 99.9 -tranche 99.0 -tranche 90.0 \

-O recalibrate\_SNP.recal \

--tranches-file recalibrate\_SNP.tranches \

--rscript-file recalibrate\_SNP\_plots.R

gatk ApplyVQSR \

-R $refdir/ram2\_all.fa \

-V $GVCF \

-ts-filter-level 99.0 \

-mode SNP \

--tranches-file recalibrate\_SNP.tranches \

--recal-file recalibrate\_SNP.recal \

-O GATK\_recalibrated\_snps\_raw\_indels.vcf

gatk --java-options "-Xmx320g -XX:ParallelGCThreads=40" VariantRecalibrator \

-R $refdir/ram2\_all.fa \

-V GATK\_recalibrated\_snps\_raw\_indels.vcf \

--resource:GFoverlap,known=false,training=true,truth=true,prior=10.0 05Gatk\_overlap.vcf \

-an DP -an QD -an FS -an SOR -an MQ -an MQRankSum -an ReadPosRankSum \

-mode INDEL \

-tranche 100.0 -tranche 99.9 -tranche 99.0 -tranche 90.0 \

--max-gaussians 4 \

-O recalibrate\_INDEL.recal \

--tranches-file recalibrate\_INDEL.tranches \

--rscript-file recalibrate\_INDEL\_plots.R

gatk ApplyVQSR \

-R $refdir/ram2\_all.fa \

-V GATK\_recalibrated\_snps\_raw\_indels.vcf \

-ts-filter-level 99.0 \

-mode INDEL \

--tranches-file recalibrate\_INDEL.tranches \

--recal-file recalibrate\_INDEL.recal \

-O 07GATK\_recalibrated\_variants.vcf

###Freebayes

ls $bamdir/\*NC056054.1.bam.RenameS\_FB.bam > Bam.fofn

freebayes-parallel ram2\_chr1.fa.100m.regions 3 -f $refdir/ram2\_all.fa --bam-list Bam.fofn \

> Freebayes\_WGS\_5sample\_jointcall.vcf