Examples of code usage during allelic binding analysis

Introduction:

The purpose of this document is to provide guidance to individuals interested in conducting similar studies. The data processing described here is carried out in a Linux environment using simple command line tools, while the allelic binding analysis is performed in R Studio. For the DAP-seq analysis, single-end reads of 50-100 base pairs were utilized. Please be aware that different sequencing tools may require different commands and software.

List of packages required for the analyses:

Trimmomatic- http://www.usadellab.org/cms/?page=trimmomatic
Bowtie2- https://bowtie-bio.sourceforge.net/bowtie2/manual.shtml
Samtools- http://www.htslib.org/
Deeptools- https://deeptools.readthedocs.io/en/latest/
IDR- https://hbctraining.github.io/Intro-to-ChIPseq/lessons/07_handling-replicates-idr.html
Bedtools- https://bedtools.readthedocs.io/en/latest/
GEM- https://bedtools.readthedocs.io/en/latest/
minimap2- https://github.com/lh3/minimap2

Section 1: Obtaining syntenic peaks between the B73 and A632 reference genomes and retrieving read counts from DAP-seq TF and gDNA (the negative control) libraries.

# Filtering adapters and low quality reads using Trimmomatic tool

```
java -jar trimmomatic-0.38.jar SE -phred33 -threads 6 {input: DAP-seq fasta file} {output: clean DAP-seq fasta file} ILLUMINACLIP:TruSeq3-SE.fa:2:30:10 SLIDINGWINDOW:6:15 MINLEN:30
```

Note: The adapter sequences used for making DAP-seq libraries were manually included in the TruSeq3-SE.fa file.

# Indexing reference genome and mapping reads to both B73 and A632 reference genomes

```
bowtie2-build  {input: reference genome in fasta format} --threads 6  {output: indexed reference genome for alighment}
```

```
bowtie2 -x {Maize genome} -q {input: clean DAP-seq fasta file} --sensitive --threads 6 -S {output: sam file}
```

# Extracting uniquely mapped reads

```
samtools view -h -F 4 {Input}| grep -v 'XS:i' > {output}
samtools view -h -h -Sb {input} -o {output}
samtools sort -Sb {input -o {output}
samtools index {input}
```

# Extracting uniquely mapped reads with perfect match (MAPQ= 42)

```
samtools view -q 42 -bs {input} -o {output}
samtools sort -Sb {input} -o {output}
samtools index {input}
```

# Visualizing mapped reads in IGV browser by converting bam files to bigwig files

```
bamCoverage -b  {input: bam file}  -o {output: bigwig file}  -bs 1 --normalizeUsing CPM
```

Note: The bigwig files were used to visualize the accumulated reads of peaks in IGV browser, shown in main Figure 4a and 5a.

# Identifying peaks using GEM tool for each DAP-seq sample

```
java -jar gem/gem.jar --d gem/Read_Distribution_default.txt --g {Maize genome size file} --genome
{Maize genome file}  --exptCond1 {input: uniquely mapped bam file of TF DAP-seq}  --ctrlCond1  {input:
uniquely mapped bam file of Halo} --f BAM --k_min 6 --k_max 13 --k_seqs 5000 --smooth 5 --mrc 20 --
poisson_control --outBED --outMEME --outNP --out {output: peak files}
```

Note: The narrow peak file is used for the IDR (Irreproducible Discovery Rate) test to assess the reproducibility of peak calls between biological replicates

# Identifying reproducible peaks using the IDR test for each TF DAP-seq dataset

```
idr --samples {input: peak file replicate 1} {output: peak file replicate 2} --input-file-type narrowPeak --
output-file  {output: high confidence peak file} --plot
```

# Extracting 5kb upstream regions and 5kb downstream regions of genes in B73 and A632 reference genomes

```
cat {input: Maize genome gff3 file}|awk -v OFS='\t' '{if($3=="gene") print $1,$4,$5,$6,$7, $9}' | sed
$'s/;/\t/g'|cut -f1-6 >{output: bed file of gene regions}

cat {input: bed file of gene regions} |awk -v OFS='\t' '{print $1,$2-5000,$3+5000,$4,$5,$6}'> {output: bed
file of 5kb regions of each genes}
```

# Identifying genes associated with DAP-seq peaks within 5kb upstream and 5kb downstream regions of the annotated transcribe regions

```
bedtools intersect -a {input: high confident peak file from GEM tool} -b {input: bed file of 5kb regions of
each genes} -wa -wb > {output:  TF DAP-seq targets}
```

# Obtaining the fasta sequences from DAP-seq peaks identified in GEM tool

```
bedtools getfasta -fi {input: Maize genome fasta file} -bed {input: DAPseq high confidence peak file} -fo {output: fasta file of peaks}
```

# Obtaining syntenic peaks in both the B73 and A632 reference genomes, the sequence of peaks was aligned to their reciprocal reference genomes. For example, the peaks identified from the B73 reference genome were aligned to the A632 reference genome, establishing peak coordinates in both genomes

```
minimap2 –secondary=no -a {Maize genome fasta file} {input: fasta file of peaks} > {output: sam file}
```

```
samtools view -S -b {input: sam file}  > {output: bam file}
```

#Converting alignment sam file to genomic position bed file

```
bedtools bamtobed -i {input: sam file}  > {output: bed file}
```

Note: During the reciprocal alignment, the syntenic peak size may exhibit dramatic variations. To enable a better comparison, we only considered peaks with mapping quality (MAPQ)> 40, and the peak size were slightly adjusted to ensure comparability between both peaks.

Table 1. Example of the synthetic peak files before filtering steps

B73 position				A632 position					
chr	Str	end	peak size	chr	Str	end	MAPQ	Strand	peak size
1	32937	33139	202	1	47070	47272	60	+	202
1	34741	35156	415	1	48874	49289	60	+	415
1	35842	36043	201	1	49974	50176	60	+	202
1	39970	40172	202	1	54103	54305	60	+	202
1	40744	41112	368	1	54878	55246	60	+	368
1	46484	46690	206	6	49299850	49300059	1	+	209
1	65504	65706	202	1	79713	79915	60	+	202
1	68679	68881	202	1	82890	83092	60	+	202
1	76505	76736	231	1	231516489	231516694	1	+	205
1	105802	106004	202	1	121244	121446	60	+	202
1	106073	106275	202	1	121515	121717	60	+	202

# Adjusting peak size of the peak in the aligned reference genome

```
cat {input: synthetic peak file (Table 1)} |awk -v OFS='\t' '{print $1,$2,$3,$4,$5,$6,$7,$8,$9,$10,int(($10-$4)/2)}'| awk -v OFS='\t' '{if($4<200) print $1,$2-$11,$3+$11,$4,$5,$6,$7,$8,$9,$10,$11;else print $1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11}'|awk -v OFS='\t' '{print $1,$2,$3,$3-$2,$5,$6,$7,$8,$9,$10,$11*2}'|sort -u > {output: reciprocal peak position file}
```

# Counting the number of perfect matching reads from the DAP-seq library and gDNA library in both the B73 and A632 reference genomes

```
bedtools intersect -b {Input: bam file from gDNA library of B73 reference} -a {B73 reciprocal peak position file} -c -bed > {output: gDNA read counts in B73}
```

```
bedtools intersect -b {Input: bam file from TF DAP-seq library of B73 reference} -a {B73 reciprocal peak position file} -c -bed > {output: TF DAP-seq read counts in B73}
```

```
bedtools intersect -b {Input: bam file from gDNA library of A632 reference} -a {A632 reciprocal peak position file} -c -bed > {output: gDNA read counts in A632}
```

```
bedtools intersect -b {Input: bam file from TF DAP-seq library of A632 reference} -a {A632 reciprocal peak position file} -c -bed > {output: TF DAP-seq read counts in A632}
```

# Merging all 4 counts files from the DAP-seq library and gDNA library of B73 and A632 reference genome

```
paste  {input: TF DAP-seq read counts in B73}  {input: TF DAP-seq read counts in A632} {input: gDNA read counts in B73} {input:  gDNA read counts in A632}  |awk '{printf("minimap2_TF_peak_%d %s\n", NR, $0)}'| awk -v OFS='\t' '{ print $1, $2":"$3"-"$4, $6":"$7"-"$8, $5, $9, $13, $17 }' > {output: peak counts table for statistical analysis}
```

Note: The output peak count table is organized as shown below for input in R.

Table 2. Example of the final peak count table used as input in R

Peak_ID	ReadCounts B73_position	ReadCounts A632_position	ReadCounts gDNA_library B73_position	ReadCounts gDNA_library A632_position	TotalCount gDNA	B73-ratio gDNA	A632-ratio gDNA
minimap2_P1_peak_4144	15	5	14	6	20	0.700	0.300
minimap2_P1_peak_60713	10	10	13	13	26	0.500	0.500
minimap2_P1_peak_35816	14	6	38	16	54	0.704	0.296
minimap2_P1_peak_60904	13	7	44	14	58	0.759	0.241
minimap2_P1_peak_75218	11	9	36	27	63	0.571	0.429
minimap2_P1_peak_74791	18	2	58	6	64	0.906	0.094
minimap2_P1_peak_4439	19	1	65	1	66	0.985	0.015

Section 2: Identifying significant allelic specific binding (ASB) peaks in R Studio

```
rm(list=ls())

library(dplyr)

Peak_counts<- read.table("input: peak count table.csv", header = T)

Peak_counts$B73_P1_DAP_freq <- Peak_counts[,2]/(Peak_counts[,2]+Peak_counts[,3])

Peak_counts$A632_P1_DAP_freq <- Peak_counts[,3]/(Peak_counts[,2]+Peak_counts[,3])

count_B73_A632 <- Peak_counts[,2:3]

dt<-as.table(as.matrix(count_B73_A632))

expected.frequencies<- as.table(as.matrix(Peak_counts[,7:8]))

PeakID <- as.matrix(Peak_ID)

result_resid <- data.frame(matrix(nrow = {input: number of peaks in the file}, ncol = 6))

for(i in 1:nrow(dt)){

  a <- chisq.test(dt[i,1:2],correct=FALSE,p=expected.frequencies[i,1:2])

  chisq_resid <- a$residuals

  chisq_p.value <- a$p.value

  result_resid[i,1] <- PeakID[i,1]

  result_resid[i,2:3] <- chisq_resid

  result_resid[i,4] <- chisq_p.value

  print(result_resid)

}

result_resid[,5] <-p.adjust(result_resid[,4], "fdr", n={input: number of peaks in the file})

result_resid[,6] = ifelse(result_resid[,5]<0.05,"sig","Not.sig")

colnames(result_resid) <- c("Peak_ID","residual_1","residual_2","chisq_p.value","FDR","result")

merge_results <- merge(Peak_counts,result_resid, by.x = "Peak_ID")

write.csv(merge_results, file = "output.csv")

ggplot(merge_results, aes(x=merge_results[,10], y=merge_results[,7]))+ geom_point(shape=pch,color=col)+ theme_classic()+ ggtitle("ASB significant peaks") + geom_hline(yintercept = 0.5,col="grey60") + geom_vline(xintercept = 0.5,col="grey60") + geom_abline(slope=1,intercept=0,col="grey60")+ xlab("B73 proportion in DAP-seq library") + ylab("B73 proportion in gDNA library") + theme(plot.title = element_text(face = "bold",hjust = 0.5))
```

Note: ggplot function is used to generate the plot in Extended Data Figure 8.