

Appendix

Section 1.1

1.1.1 R script for tested population and artificial infection

```
# Loading packages  
# R.Version()$version.string  
library("ggplot2")  
library("ggpubr")  
library("extrafont")  
# print(paste("ggplot2 version",packageVersion("ggplot2")))  
# print(paste("ggpubr version",packageVersion("ggpubr")))  
# print(paste("extrafont version",packageVersion("extrafont")))  
# remotes::install_version("Rttf2pt1", version = "1.3.8")  
# ttf_import()  
# fonts()  
# loadfonts(device = "pdf")  
# extrafont::loadfonts(device = "postscript")  
  
# Loading phenotypes  
pheno <- read.csv("./2017heteroPheno.csv", header=T)  
pheno$HD<-100*pheno$bcw/pheno$length^2  
head(pheno)  
str(pheno)  
attach(pheno)  
  
# Mean & S.D.  
meanSD <- data.frame(bcw,length,HD)  
as.data.frame( t(sapply(meanSD, function(cl)  
list(mean=round(mean(cl,na.rm=TRUE),2),sd=round(sd(cl,na.rm=TRUE),2))))  
  
# Normality test  
shapiro.test(bcw)  
shapiro.test(length)  
shapiro.test(HD)  
# shapiro.test(sqrt(bcw+1))  
  
# Pearson's r  
cor.test(bcw,length)  
# round(cor(bcw,length),2)
```

```

# Histogram
# HC

line_th_unit<-0.6 ##size unit, mm
scaleFUN <- function(x) sprintf("%.2f", x)
options(repr.plot.width=4, repr.plot.height=4)
p1<-ggplot(pheno, aes(x=bew))+

ggpubr::theme_pubr(base_family="Times New Roman",base_size = 12)+

geom_histogram(aes(y = ..density..),color="black",fill="grey",
,→binwidth=1,size=line_th_unit)+

geom_density(size=line_th_unit)+

labs(x="HC", y="Density")

p1$theme$line$size<-line_th_unit
p1$theme$line$size<-line_th_unit
p1$theme$axis.ticks$size<-line_th_unit
p1$theme$axis.line$size<-line_th_unit
p1$theme$panel.grid$colour<-"black"
p1$theme$axis.ticks$colour<-"black"

# sqrt(bew+1)

p3<-ggplot(pheno, aes(x=sqrt(bew+1)))+
ggpubr::theme_pubr(base_family="Times New Roman",base_size = 12)+

geom_histogram(aes(y = ..density..),color="black",fill="grey",
,→binwidth=0.1,size=line_th_unit)+

geom_density(size=line_th_unit)+

labs(x="Transformed HC", y="Density")

p3$theme$line$size<-line_th_unit
p3$theme$line$size<-line_th_unit
p3$theme$axis.ticks$size<-line_th_unit
p3$theme$axis.line$size<-line_th_unit
p3$theme$panel.grid $ colour<-"black"
p3$theme$axis.ticks$colour<-"black"

# SL

p2<-ggplot(pheno, aes(x=length))+

ggpubr::theme_pubr(base_family="Times New Roman",base_size = 12)+

geom_histogram(aes(y = ..density..),color="black",fill="grey",
,→binwidth=0.1,size=line_th_unit)+

labs(x="SL", y="Density")

```

```

geom_density(size=line_th_unit)
p2$theme$line$size<-line_th_unit
p2$theme$line$size<-line_th_unit
p2$theme$axis.ticks$size<-line_th_unit
p2$theme$axis.line$size<-line_th_unit
p2$theme$panel.grid $ colour<-"black"
p2$theme$axis.ticks$colour<-"black"
# p2

options(repr.plot.width=6, repr.plot.height=6)
figure <- ggpubr::ggarrange(p1, p3, p2,
labels = c("a", "b", "c"),
ncol = 1,
nrow = 3,
font.label = list(size = 14,
color = "black",
face = "plain",
family = "Times New Roman"))
figure

# Save
ggsave("Fig. 1.1.pdf",device="pdf",units="in",width =6, height =6 )
# ggsave("Fig1.eps",device="eps",units="in",width =7.2)
# ggsave("Fig. 1.1.tiff", width=7.2, units="in", res=300)

```

1.1.2 Bash script for genotyping

```
#!/bin/bash

# Trim
for i in `seq 1 252`
do
time=$(date)
echo ${time}
mkdir /mnt/c/gs2017ngsdata/TRIM/${i}
Samples=/mnt/c/gs2017ngsdata/lin-60808054/
,→FASTQ_Generation_2018-01-23_06_29_51Z-74544520/${i}
TRIM=/mnt/c/gs2017ngsdata/TRIM/${i}
java -jar /mnt/c/linux/bin/Trimmomatic-0.36/trimmomatic-0.36.jar PE
,→-threads 8 -trimlog ${TRIM}"/$S"${i}"trim.log" \
${Samples}/*_R1_001.fastq.gz ${Samples}/*_R2_001.fastq.gz \
${TRIM}"/$S"${i}"_paired_R1.fq.gz ${TRIM}"/$S"${i}"_unpaired_R1.fq.gz" \
${TRIM}"/$S"${i}"_paired_R2.fq.gz ${TRIM}"/$S"${i}"_unpaired_R2.fq.gz" \
ILLUMINACLIP:/mnt/c/linux/bin/Trimmomatic-0.36/adapters/TruSeq3-PE-2.fa:2:
,→30:10 CROP:146 HEADCROP:5 LEADING:19 TRAILING:19 SLIDINGWINDOW:30:20
,→AVGQUAL:20 MINLEN:60
done

# Index reference
reference=/mnt/c/linux/bin/reference/fugu5_SeedsDesign3342.fa
bwa index ${reference}
samtools faidx ${reference}

# Generate the sequence dictionary by running the following Picard command
java -jar /mnt/c/linux/bin/picard.jar CreateSequenceDictionary \
REFERENCE=${reference} \
OUTPUT=reference.dict

# Map on reference genome
reference=/mnt/c/linux/bin/reference/fugu5_SeedsDesign3342.fa
for i in `seq 1 252`
do
mkdir /mnt/c/gs2017ngsdata/BWA/${i}
BWA=/mnt/c/gs2017ngsdata/BWA/${i}
TRIM=/mnt/c/gs2017ngsdata/TRIM/${i}
```

```

bwa mem -t 16 -M $reference ${TRIM}/*$i*_paired_R1.fq.gz ${TRIM}/*
,→${i}_paired_R2.fq.gz -R "@RG\tID:$i\tSM:$i\tPL:Illumina"
,→> ${BWA}/*$i.bwa.sam
done

```

```

# Sort
for i in `seq 1 252`
do
BWA=/mnt/c/gs2017ngsdata/BWA/$i
TRIM=/mnt/c/gs2017ngsdata/TRIM/$i
samtools view -S -q 4 ${BWA}/*$i.bwa.sam -b > ${BWA}/*$i.bwa2.
,→bam"
samtools sort ${BWA}/*$i.bwa2.bam ${BWA}/*$i._sorted"
samtools index ${BWA}/*$i._sorted.bam"
samtools flagstat ${BWA}/*$i._sorted.bam > ${BWA}/*$i.flagstat"
echo ${BWA}/*$i._sorted >> /mnt/c/gs2017ngsdata/BWA/flagstat
cat ${BWA}/*$i.flagstat >> /mnt/c/gs2017ngsdata/BWA/flagstat
done

```

```

# Move files
cd /mnt/c/linux/bin/BWA
for i in `seq 1 252`
do
mkdir ${i}
cp /mnt/f/gs2017_lin/6_ngs_analysis/BWA/${i}/*$i._sorted.bam ./${i}/*
,→${i}.sorted.bam"
cp /mnt/f/gs2017_lin/6_ngs_analysis/BWA/${i}/*$i._sorted.bam.bai /
,→mnt/c/linux/bin/BWA/${i}/*$i._sorted.bam.bai"
done
# Consolidate GVCFs
DIR=/home/lin/GS2017_jointcalling_20200526
REF=/home/lin/GS2017_jointcalling_20200526/reference/
,→fugu5_SeedDesign3342.fa

```

```

gatk=/home/lin/gatk-4.1.6.0/gatk
# Use gnu-parallel
cat ${DIR}/sample_list.txt | parallel --verbose -j 6 "${gatk}"
,→HaplotypeCaller --output-mode EMIT_ALL_CONFIDENT_SITES -stand-call-conf

```

```

,→30 -R ${REF} -I ${DIR}/BWA_q10_bamOnly/{}_sorted.bam -O ${DIR}/VCF/
,→{}_pe_variants_gatk.g.vcf.gz -ERC GVCF"
# Joint-Call Cohort
echo "${gatk} CombineGVCFs" > ${DIR}/CombineGVCFs.sh
for i in `cat ${DIR}/sample_list.txt `
do
echo "-V ${DIR}/VCF/${i}_pe_variants_gatk.g.vcf.gz" >> ${DIR}/
,→CombineGVCFs.sh
done
echo "-R ${REF}" >> ${DIR}/CombineGVCFs.sh
echo "-O ${DIR}/VCF/cohort.g.vcf.gz" >> ${DIR}/CombineGVCFs.sh
cat ${DIR}/CombineGVCFs.sh | tr "\n" " " > ${DIR}/Spa_CombineGVCFs.sh
.${DIR}/Spa_CombineGVCFs.sh
tabix -p vcf ${DIR}/VCF/cohort.g.vcf.gz
${gatk} GenotypeGVCFs -R ${REF} -V ${DIR}/VCF/cohort.g.vcf.gz -O ${DIR}/
,→VCF/Output_jonit_call_cohort.vcf.gz --tmp-dir=${DIR}/VCF/temp
# SNP filtering
bgzip -d ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz
bgzip ${DIR}/VCF/Output_jonit_call_cohort.vcf
tabix -p vcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz
#vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --min-meanDP
,→6 --max-meanDP 500 --max-missing 0.3 --remove-indels --recode --stdout
,→> ${DIR}/VCF/Hetero_realigned_cov10_filtered.vcf
#10711 (18186)
#vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --min-meanDP
,→10 --max-meanDP 500 --max-missing 0.5 --remove-indels --recode
,→--stdout > ${DIR}/VCF/Hetero_realigned_cov10_filtered.vcf
#9437 (18186)
#vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --min-meanDP
,→15 --max-meanDP 500 --max-missing 0.5 --remove-indels
#8216 (18186)
#vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --min-meanDP
,→15 --max-meanDP 500 --max-missing 0.7 --remove-indels
#8184 (18186)

#vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --min-meanDP
,→15 --max-meanDP 500 --max-missing 0.7 --hwe 0.01 --minDP 10
,→--remove-indels

```

```

#7096 (18186)
#vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --min-meanDP
,→15 --max-meanDP 500 --max-missing 0.7 --hwe 0.05 --minDP 10
,→--remove-indels

#6718 (18186)
#vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --min-meanDP
,→15 --max-meanDP 500 --max-missing 0.7 --hwe 0.1 --minDP 10
,→--remove-indels

#6411 (18186)
vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --min-meanDP
,→15 --max-meanDP 500 --max-missing 0.7 --hwe 0.05 --minDP 10
,→--remove-indels --recode --stdout > ${DIR}/VCF/
,→Hetero_realigned_cov10_filtered.vcf

#6718 (18186)
zgrep -v "##" ${DIR}/VCF/Hetero_realigned_cov10_filtered.vcf | awk
,→
'{print $1"\t"$2}' > ${DIR}/VCF/position.list
cat -n ${DIR}/VCF/position.list | awk '{print $2"\t"$1}' > ${DIR}/VCF/
,→Chom.map
bgzip ${DIR}/VCF/Hetero_realigned_cov10_filtered.vcf
tabix -p vcf ${DIR}/VCF/Hetero_realigned_cov10_filtered.vcf.gz

# Imputation
java -jar LinkImputeR.jar -s accuracy.ini
java -jar LinkImputeR.jar impute2.xml "Case 4" output.vcf
# accuracy.ini

[Input]
filename = Hetero_realigned_cov10_filtered.vcf.gz
save = filtered.vcf

[InputFilters]
[Global]
depth = 3

[CaseFilters]
missing = 0.8,0.85,0.9,0.95

[Stats]
root = ./

level = table

[Output]
control = ./impute2.xml

```

```

[Log]
file = log.txt
level = debug

[Accuracy]
numbermasked = 500
mindepth = 5
# impute2.xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<linkimputepro>
<mode>impute</mode>
<input>
<filename>filtered.vcf</filename>
<maxdepth>100</maxdepth>
</input>
<case>
<name>Case 1</name>
<filter name="PositionMissing">
<threshold>0.8</threshold>
<mindepth>3</mindepth>
</filter>
<filter name="SampleMissing">
<threshold>0.8</threshold>
<mindepth>3</mindepth>
</filter>
<caller name="Binomial">
<error>0.01</error>
</caller>
<imputation name="KnniLD">
<k>12</k>
<l>36</l>
<knowndepth>3</knowndepth>
</imputation>
<combiner name="MaxDepth">
<w>0.01</w>
<maxdepth>3</maxdepth>
</combiner>
</case>
<case>

```

```

<name>Case 2</name>
<filter name="PositionMissing">
<threshold>0.85</threshold>
<mindepth>3</mindepth>
</filter>
<filter name="SampleMissing">
<threshold>0.85</threshold>
<mindepth>3</mindepth>
</filter>
<caller name="Binomial">
<error>0.01</error>
</caller>
<imputation name="KnniLD">
<k>25</k>
<l>36</l>
<knowndepth>3</knowndepth>
</imputation>
<combiner name="MaxDepth">
<w>0.5400000000000003</w>
<maxdepth>3</maxdepth>
</combiner>
</case>
<case>
<name>Case 3</name>
<filter name="PositionMissing">
<threshold>0.9</threshold>
<mindepth>3</mindepth>
</filter>
<filter name="SampleMissing">
<threshold>0.9</threshold>
<mindepth>3</mindepth>
</filter>
<caller name="Binomial">
<error>0.01</error>
</caller>
<imputation name="KnniLD">
<k>30</k>

```

```

<l>26</l>
<knowndepth>3</knowndepth>
</imputation>
<combiner name="MaxDepth">
<w>0.48000000000000026</w>
<maxdepth>3</maxdepth>
</combiner>

</case>
<case>
<name>Case 4</name>
<filter name="PositionMissing">
<threshold>0.95</threshold>
<mindepth>3</mindepth>
</filter>
<filter name="SampleMissing">
<threshold>0.95</threshold>
<mindepth>3</mindepth>
</filter>
<caller name="Binomial">
<error>0.01</error>
</caller>
<imputation name="KnniLD">
<k>9</k>
<l>28</l>
<knowndepth>3</knowndepth>
</imputation>
<combiner name="MaxDepth">
<w>0.01</w>
<maxdepth>3</maxdepth>
</combiner>
</case>
</linkimputepro>
# Imputed genotypes to SNP matrix
bgzip output.vcf
tabix -p vcf output.vcf.gz
vcftools --gzvcf output.vcf.gz --plink --chrom-map Chom.map --out
,→Hetero_realigned_cov10_filtered

```

```

plink1 --ped Hetero_realigned_cov10_filtered.ped --map
,→Hetero_realigned_cov10_filtered.map --recodeA --out
,→Hetero_realigned_cov10_filtered --noweb
echo "empty" > IDlist
zgrep "#CHROM" output.vcf.gz | cut -f10- | sed -e "s/\t/\n/g" >> IDlist
sed -e "s/empty//g" IDlist > IDs.txt
cat Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- >
,→Hetero_realigned_cov10_filtered2.raw
paste -d" " IDs.txt Hetero_realigned_cov10_filtered2.raw >
,→Hetero_realigned_cov10_filtered3.raw
sed -n "1p" Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- | sed
,→-e "s/ \n/g" > Hetero_realigned_cov10_filtered.pos

less Hetero_realigned_cov10_filtered.pos|sed -e "s/_/\t/g"|sed -e "s/:/\t/
,→g"|sed -e "s/-/\t/g"|awk '{print $2,$3+$5}'>lin.map
# Reads statistics
for i in `seq 252`
do
statsdir=/mnt/c/gs2017ngsdata
if [[ $i == 105 || $i == 114|| $i == 123 || $i == 132|| $i == 141 ||
,→$i == 150|| $i == 159|| $i == 168|| $i == 177|| $i == 186|| $i == 195 ||
,→$i == 204 ]]
then
continue
fi
/mnt/c/linux/bin/seqstats/seqstats /mnt/c/gs2017ngsdata/lin-60808054/
,→FASTQ_Generation_2018-01-23_06_29_51Z-74544520/${i}/
,→"${i}"_S"${i}"_L001_R1_001.fastq.gz|sed -n 1p|cut -f 2 >>${statsdir}/
,→raw_num_read_1.txt
/mnt/c/linux/bin/seqstats/seqstats /mnt/c/gs2017ngsdata/lin-60808054/
,→FASTQ_Generation_2018-01-23_06_29_51Z-74544520/${i}/
,→"${i}"_S"${i}"_L001_R2_001.fastq.gz|sed -n 1p|cut -f 2 >>${statsdir}/
,→raw_num_read_2.txt
/mnt/c/linux/bin/seqstats/seqstats /mnt/c/gs2017ngsdata/lin-60808054/
,→FASTQ_Generation_2018-01-23_06_29_51Z-74544520/${i}/
,→"${i}"_S"${i}"_L001_R1_001.fastq.gz|sed -n 2p|cut -f 2|cut -d" " -f1
,→>>${statsdir}/raw_bp_1.txt
/mnt/c/linux/bin/seqstats/seqstats /mnt/c/gs2017ngsdata/lin-60808054/

```

```
,→FASTQ_Generation_2018-01-23_06_29_51Z-74544520/${i}/  
,→"${i}"_S"${i}"_L001_R2_001.fastq.gz|sed -n 2p|cut -f 2|cut -d" " -f1  
,→>>${statsdir}/raw_bp_2.txt  
/mnt/c/linux/bin/seqstats/seqstats /mnt/c/gs2017ngsdata/TRIM/${i}/  
,→S"${i}"_paired_R1.fq.gz|sed -n 1p|cut -f 2 >>${statsdir}/  
,→trined_paired_num_read_1.txt  
/mnt/c/linux/bin/seqstats/seqstats /mnt/c/gs2017ngsdata/TRIM/${i}/  
,→S"${i}"_paired_R2.fq.gz|sed -n 1p|cut -f 2 >>${statsdir}/  
,→trined_paired_num_read_2.txt  
/mnt/c/linux/bin/seqstats/seqstats /mnt/c/gs2017ngsdata/TRIM/${i}/  
,→S"${i}"_paired_R1.fq.gz|sed -n 2p|cut -f 2|cut -d" " -f1 >>${statsdir}/  
,→trined_paired_bp_1.txt  
/mnt/c/linux/bin/seqstats/seqstats /mnt/c/gs2017ngsdata/TRIM/${i}/  
,→S"${i}"_paired_R2.fq.gz|sed -n 2p|cut -f 2|cut -d" " -f1 >>${statsdir}/  
,→trined_paired_bp_2.txt  
done
```

1.1.3 R script for genotyping

```
# Read statistics  
rb1<-read.table("raw_bp_1.txt")  
rb2<-read.table("raw_bp_2.txt")  
rn1<-read.table("raw_num_read_1.txt")  
rn2<-read.table("raw_num_read_1.txt")  
tb1<-read.table("trined_paired_bp_1.txt")  
tb2<-read.table("trined_paired_bp_2.txt")  
tn1<-read.table("trined_paired_num_read_1.txt")  
tn2<-read.table("trined_paired_num_read_2.txt")  
  
(sum(rb1)+sum(rb2))  
(sum(tb1)+sum(tb2))  
(sum(tb1)+sum(tb2))/(sum(rb1)+sum(rb2))  
  
mean(unlist(rn1+rn2))  
sd(unlist(rn1+rn2))  
  
mean(unlist(tn1+tn2))  
sd(unlist(tn1+tn2))  
mean(unlist(tn1+tn2))/mean(unlist(rn1+rn2))  
  
(sum(rn1)+sum(rn2))  
(sum(tn1)+sum(tn2))  
(sum(tn1)+sum(tn2))/(sum(rn1)+sum(rn2))  
  
(sum(rb1)+sum(rb2))/(sum(rn1)+sum(rn2))  
(sum(tb1)+sum(tb2))/(sum(tn1)+sum(tn2))
```

1.1.4 Python script for population structure

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm
import seaborn as sns; sns.set()
import matplotlib as mpl

# Loading data
pheno = pd.read_csv('./2017heteroPheno.csv')
#I = pd.read_table('./Hetero_realigned_cov10_filtered3.raw',sep="\s+")
I = pd.read_table('./reorderG.txt',sep="\s+")-1
bcw=pheno.bcw

# tSNE
from sklearn.manifold import TSNE
model = TSNE(random_state=0,perplexity=20)
tsne5 = model.fit_transform(I)

lin_wide=1.28
font_size=12
%matplotlib inline
%config InlineBackend.figure_format = 'svg'
plt.style.use('seaborn-white')
plt.style.use('seaborn-paper')
plt.rc('font',size=font_size)
plt.rc('lines',lw=lin_wide)
plt.rcParams['font.family'] = 'serif'
plt.rcParams['font.sans-serif'] = 'Times New Roman'
#plt.rcParams['axes.formatter.use_mathtext'] = True
#plt.rcParams['mathtext.bf'] = 'serif:normal'

#plt.rcParams['text.hinting'] = 'none'
#plt.rcParams['text.usetex'] = True
#plt.rcParams['axes.labelweight'] = 'normal'
plt.rcParams['axes.linewidth'] = lin_wide
plt.rcParams['xtick.major.width'] = lin_wide
plt.rcParams['xtick.minor.width'] = lin_wide
```

```

plt.rcParams['xtick.labelsize'] = font_size
plt.rcParams['ytick.major.width'] = lin_wide
plt.rcParams['ytick.minor.width'] = lin_wide
plt.rcParams['ytick.labelsize'] = font_size
plt.rcParams['axes.spines.bottom'] = True
# plt.style.use('bmh')
plt.scatter(tsne5[:, 0], tsne5[:, 1], edgecolor='k', c=bew, cmap=cm.
,→Reds, linewidth=lin_wide, s=50)
plt.xlabel("t-SNE coordinate 1", size=font_size)
plt.ylabel("t-SNE coordinate 2", size=font_size)
plt.subplots_adjust(bottom=0.1, right=0.8, top=0.9)
cax = plt.axes([0.85, 0.1, 0.075, 0.8])
plt.colorbar(cax=cax)
plt.gcf().set_size_inches(7, 6)
plt.savefig("tSNE_HC.pdf", format="pdf")
plt.show()
# plt.savefig("PCA_diets.png", dpi=300)

```

1.1.5 R script for heritability and genetic correlation

```
reorderG<-read.table("GS2017/reorderG.txt")
x<-as.matrix(reorderG)-1
pheno<-read.csv("GS2017/2017heteroPheno.csv")
n<-dim(x)[1]

library("sommer")
packageVersion("sommer")
A <- A.mat(x,n.core=8)
row.names(A)=1:n;colnames(A)=1:n
data <- data.frame(tbcw=bew,length=length,id=factor(1:n))
attach(data)
ans.m <- mmer(cbind(tbcw,length)~1,random=~ vs(id, Gu=A, Gtc=unsm(2)),
               rcov=~ vs(units, Gtc=unsm(2)),
               data=data)
(sp_cor<-cov2cor(ans.m$sigma$`u:id`))
pin(ans.m, h2~ V1/(V1+V4))
pin(ans.m, h2~ V3/(V3+V6))
```

1.1.6 R script for GWAS

```
# Loading packages
# R.Version()$version.string

library("ggplot2")
library("ggpubr")
library("extrafont")

# print(paste("ggplot2 version",packageVersion("ggplot2")))
# print(paste("ggpubr version",packageVersion("ggpubr")))
# print(paste("extrafont version",packageVersion("extrafont")))
# remotes::install_version("Rttf2pt1", version = "1.3.8")
# ttf_import()
# fonts()
# loadfonts(device = "pdf")

# Loading phenotypes
pheno <- read.csv("./2017heteroPheno.csv", header=T)
pheno$HD<-100*pheno$bcw/pheno$length^2
attach(pheno)
n<-length(pheno$bcw)

# Loading imputed genotypes
geno <-read.table("./Imputation/Hetero_realigned_cov10_filtered3.raw",
,header=T) # Imputed
dim(geno)

#cat IDlist|sed -e "s/S/t/g"|sort -n|sed -e "s/t/S/g"|awk "{print
,->$2}">ID_list.txt
id<-read.table("ID_list.txt")#fitSID
reorderIndex<-c()
for (i in id$V1){
reorderIndex<-c(reorderIndex,match(i,rownames(geno)))
}
reorderG<-geno[reorderIndex,]
rownames(reorderG)=1:n
x <- as.matrix(reorderedG)-1
write.table(reorderedG,"reorderedG.txt")

# GBLUP
```

```

library(rrBLUP)
print(paste("rrBLUP version",packageVersion("rrBLUP")))
# genomic relationship matrix
A <- A.mat(x)
row.names(A)=1:n;colnames(A)=1:n

# data
data <- data.frame(tbcw=pheno$bcbw,length=pheno$length,
bcbw_d_length2=pheno$HD, gid=1:n,x=x)
row.names(A)=1:n;colnames(A)=1:n

# GWAS for HC
lin_map<-read.table("./Imputation/lin.map")
g <- data.frame(rownames(lin_map),lin_map$V1, lin_map$V2, t(x))
rownames(g) <-1:nrow(g)
colnames(g) <-c("marker", "chrom", "pos", rownames(x))
# Bonferroni-corrected significance threshold
(thred <- round(-log10(0.05/nrow(g)),3))

bcwframe <-data.frame(1:n, data$bcbw)
colnames(bcwframe) <-c("gid", "bcbw")
# Performing GWAS
GWAS_bcw <-GWAS(bcwframe, g,K=A, n.PC = 10)
# SNP markers with lowest P-values
GWAS_bcw$chrom[head(order(GWAS_bcw$bcbw,decreasing = TRUE),3)]
# Corresponding position (base pair)
GWAS_bcw$pos[head(order(GWAS_bcw$bcbw,decreasing = TRUE),3)]

# top100
top100<-cbind(rep("GS2017",100), GWAS_bcw$chrom[head(order(GWAS_bcw$bcbw,decreasing =
,TRUE),100)],
GWAS_bcw$pos[head(order(GWAS_bcw$bcbw,decreasing =
,TRUE),100)])
colname<-c("Exp","chr","pos")
write.csv(top100,"GS2017_GAWS_HC100.csv")

# Manhattan plot
line_th_unit<-0.6

```

```

Chrom<-GWAS_bcw$chrom
GWAS_bcw$chrom<-factor(Chrom,levels=c(as.character(1:22)))
options(repr.plot.width=8, repr.plot.height=4)
my_y_title <- expression(paste("-", "log"[10], (" ", italic("p"), ") for
,→HC"))
p1<-ggplot(GWAS_bcw, aes(x=1:dim(x)[2], y=tbcw, color=chrom))+  

  ggpubr::theme_pubr(base_family="Times New Roman", base_size =
,→12, border = TRUE)+  

  geom_point()+
  scale_color_manual(values = rep(c("black", "skyblue"), 24 ))+  

  labs(x="Physical order of SNPs", y=my_y_title)+  

  theme(legend.position="none",
axis.text.y=element_text(size=12),
axis.text.x=element_text(size=12),
text=element_text(size=12,
family="Times New Roman"))+  

  geom_hline(yintercept=thred, linetype="dashed", color = "red")+
  geom_text(aes(0,thred,label = thred, vjust = 1.2),color =
,→"black",family="Times New Roman")
p1$theme$line$size<-line_th_unit
p1$theme$line$size<-line_th_unit
p1$theme$axis.ticks$size<-line_th_unit
p1$theme$axis.line$size<-line_th_unit
p1$theme$panel.grid$colour<-"black"
p1$theme$axis.ticks$colour<-"black"
p1

```

```

fmBC_HC=BGLR(y=data$tbcw,ETA=list(list(X=x,model='BayesC')),
  nIter=10000,burnIn=2000,verbose = FALSE)
mean(unlist(fmBC_HC$ETA[[[]]$b))
sd(unlist(fmBC_HC$ETA[[[]]$b)))
max(unlist(fmBC_HC$ETA[[[]]$b)))

```

```

# GWAS for SL
lengthframe <-data.frame(1:n, length)
colnames(lengthframe) <-c("gid", "length")
# Performing GWAS
GWAS_length <-GWAS(lengthframe, g,K=A, n.PC = 10)

```

```

# SNP markers with lowest P-values
GWAS_length$chrom[head(order(GWAS_length$length,decreasing = TRUE),3)]
# Corresponding position (base pair)
GWAS_length$pos[head(order(GWAS_length$length,decreasing = TRUE),3)]

# top100
top100<-cbind(rep("GS2017",100),
GWAS_length$chrom[head(order(GWAS_length$length,decreasing
,→= TRUE),100)],
GWAS_length$pos[head(order(GWAS_length$length,decreasing =
,→TRUE),100)])
colname<-c("Exp","chr","pos")
write.csv(top100,"GS2017_GAWS_SL100.csv")

Chrom<-GWAS_length$chrom
GWAS_length$chrom<-factor(Chrom,levels=c(as.character(1:22)))
options(repr.plot.width=8, repr.plot.height=4)
my_y_title <- expression(paste("-", "log"[10], "(", italic("p"), ") for
,→SL"))
p2<-ggplot(GWAS_length, aes(x=1:dim(x)[2], y=length, color=chrom)) +
  ggpubr::theme_pubr(base_family="Times New Roman", base_size =
,→12, border = TRUE) +
  geom_point() +
  scale_color_manual(values = rep(c("black", "skyblue"), 24 )) +
  labs(x="Physical order of SNPs", y=my_y_title) +
  theme(legend.position="none",
  axis.text.y=element_text(size=12),
  axis.text.x=element_text(size=12),
  text=element_text(size=12,
  family="Times New Roman")) +
  geom_hline(yintercept=thred, linetype="dashed", color = "red") +
  geom_text(aes(0,thred, label = thred, vjust = 1.2), color =
,→"black", family="Times New Roman")
p2$theme$line$size<-line_th_unit
p2$theme$line$size<-line_th_unit
p2$theme$axis.ticks$size<-line_th_unit
p2$theme$axis.line$size<-line_th_unit
p2$theme$panel.grid$colour<-"black"

```

```

p2$theme$axis.ticks$colour<-"black"
p2

options(repr.plot.width=5, repr.plot.height=5)
figure <- ggpubr::ggarrange(p1, p2,
labels = c("a", "b"),
ncol = 1,
nrow = 2,
font.label = list(size = 14,
color = "black",
face = "plain",
family = "Times New Roman"))
# figure

ggsave("Fig. 1.2.pdf",device="pdf",units="in",width =6, height = 6 )
# ggsave("Fig2.eps",device="eps",units="in",width =7.2)
# ggsave("Fig. 1.2.tiff", width=7.2, units="in", res=300)

fmBC=BGLR(y=data$length,ETA=list(list(X=x,model='BayesC')),
nIter=10000,burnIn=2000,verbose = FALSE)

mean(unlist(fmBC$ETA[[[]]]$b))

max(unlist(fmBC$ETA[[[]]]$b))

```

Section 1.2

1.2.1 R script for predictive ability of GP

```
# HC

# Platform information
library(benchmarkme)
get_platform_info()$OS.type
get_r_version()$version.string
get_cpu()$model_name;get_cpu()$no_of_cores
get_ram()

# Parallel computation
library(doParallel)
library(foreach)
cl<-makeCluster(16)

geno <-read.table("./Hetero_realigned_cov10_filtered3.raw", header=T)
pheno <- read.csv("./2017heteroPheno.csv", header=T)
attach(pheno)
n<-length(pheno$bew)
#cat IDlist|sed -e "s/S/t/g"|sort -n|sed -e "s/t/S/g"|awk "{print $2}">ID_list.txt
id<-read.table("ID_list.txt")#fitSID
reorderIndex<-c()
for (i in id$V1){
  reorderIndex<-c(reorderIndex,match(i,rownames(geno)))
}
reorderG<-geno[reorderIndex,]
rownames(reorderG)=1:n
x <- as.matrix(reorderG)-1
data <- data.frame(tbcw=sqrt(pheno$bew+1),bcw=pheno$bew,length=pheno$length,gid=1:240)

# Parameters for cross validation
repeats <- 10
n.fold <- 10
n.sample <- length(pheno$bew)
CM<-7

# Package
library(rrBLUP)
```

```

packageVersion("rrBLUP")

# Marker-based relationship matrix (Endelman et al. 2011)
A <- A.mat(x, n.core=8)
row.names(A)=1:240;colnames(A)=1:240

registerDoParallel(cl)
system.time({
  GBLUP<-foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {
      bcw_test <- data
      bcw_test$tbcw[id == i] <- NA
      res <- kin.blup(bcw_test, K=A, geno="gid", pheno="tbcw",covariate = "length")
      cor(data$tbcw[id==i],res$pred[id==i])
    }
  }
})
stopImplicitCluster()

mean(unlist(GBLUP))

data <- data.frame(tbcw=sqrt(pheno$bew+1),
                    bew=pheno$bew,
                    hc2=sqrt(pheno$bew+1)/pheno$length^2,
                    length=pheno$length,
                    length2=pheno$length^2,gid=1:240)

resHC1 <- kin.blup(data, K=A, geno="gid", pheno="tbcw",covariate = "length")
#resHC2 <- kin.blup(data, K=A, geno="gid", pheno="tbcw")
#resHC3 <- kin.blup(data, K=A, geno="gid", pheno="hc2")
#resHC4 <- kin.blup(data, K=A, geno="gid", pheno="tbcw",covariate = "length2")
resSL <- kin.blup(data, K=A, geno="gid", pheno="length")

GEBVHC1 <- resHC1$pred
GEBVHC2 <- resHC2$pred
GEBVHC3 <- resHC3$pred

```

```

GEBVHC4 <- resHC4$pred
GEBVSL <- resSL$pred

cor(GEBVHC1,GEBVSL)
cor(GEBVHC2,GEBVSL)
cor(GEBVHC3,GEBVSL)
cor(GEBVHC4,GEBVSL)
cor(GEBVHC3,GEBVHC1)
cor(GEBVHC3,GEBVHC2)
cor(GEBVHC4,GEBVHC1)

cor(data$tbcw,data$length,use="complete")
cor(data$hc2,data$length,use="complete")
#plot(GEBVHC1,GEBVSL)

cor.test(GEBVHC1,GEBVSL)

library("BGLR")
packageVersion("BGLR")

registerDoParallel(cl)
system.time({
BA <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$tbcw[id == i] <- NA
    fmBA=BGLR(y=bcw_test$tbcw,
    ETA=list(
      mrk = list(X=x,model='BayesA'),
      fixed=list(~length,data=bcw_test,model='FIXED')),
      nIter=10000,burnIn=2000,verbose = FALSE)
    cor(data$tbcw[id == i],fmBA$yHat[id == i])
  }
})
stopImplicitCluster()
})

```

```

registerDoParallel(cl)
system.time({
  BB <- foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="BGLR") %dopar% {
      bcw_test <- data
      bcw_test$tbcw[id == i] <- NA
      fmBB=BGLR(y=bew_test$tbcw,
                  ETA=list(
                    mrk = list(X=x,model='BayesB'),
                    fixed=list(~length,data=bew_test,model='FIXED')),
                    nIter=10000,burnIn=2000,verbose = FALSE)
      #fmBB=BGLR(y=bew_test$tbcw,ETA=list(list(X=x,model='BayesB')),nIter=10000,burnIn=2000,verbose
      = FALSE)
      cor(data$tbcw[id == i],fmBB$yHat[id == i])
    }
  }
})
stopImplicitCluster()

registerDoParallel(cl)
system.time({
  BC <- foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="BGLR") %dopar% {
      bcw_test <- data
      bcw_test$tbcw[id == i] <- NA
      fmBC=BGLR(y=bew_test$tbcw,
                  ETA=list(
                    mrk = list(X=x,model='BayesC'),
                    fixed=list(~length,data=bew_test,model='FIXED')),
                    nIter=10000,burnIn=2000,verbose = FALSE)
      #fmBC=BGLR(y=bew_test$tbcw,ETA=list(list(X=x,model='BayesC')),nIter=10000,burnIn=2000,verbose
      = FALSE)
      cor(data$tbcw[id == i],fmBC$yHat[id == i])
    }
  }
})

```

```

        }
    }
})

stopImplicitCluster()

registerDoParallel(cl)
system.time({
  BL <- foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="BGLR") %dopar% {
      bcw_test <- data
      bcw_test$tbcw[id == i] <- NA
      fmBL=BGLR(y=bcw_test$tbcw,
                  ETA=list(
                    mrk = list(X=x,model='BL'),
                    fixed=list(~length,data=bcw_test,model='FIXED')),
                  nIter=10000,burnIn=2000,verbose = FALSE)
      #fmBL=BGLR(y=bcw_test$tbcw,ETA=list(list(X=x,model='BL')),nIter=10000,burnIn=2000,verbose =
      FALSE)
      cor(data$tbcw[id == i],fmBL$yHat[id == i])
    }
  }
})
stopImplicitCluster()

registerDoParallel(cl)
system.time({
  BRR <- foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="BGLR") %dopar% {
      bcw_test <- data
      bcw_test$tbcw[id == i] <- NA
      fmBRR=BGLR(y=bcw_test$tbcw,
                  ETA=list(
                    mrk = list(X=x,model='BRR'),
                    fixed=list(~length,data=bcw_test,model='FIXED'))),
    }
  }
})

```

```

nIter=10000,burnIn=2000,verbose = FALSE)
#fmBRR=BGLR(y=bcw_test$bcw,ETA=list(list(X=x,model='BRR')),nIter=10000,burnIn=2000,verbose
= FALSE)
  cor(data$bcw[id == i],fmBRR$yHat[id == i])
}
}
})
stopImplicitCluster()

# Reproducing kernel
p<-ncol(x)
D<-(as.matrix(dist(x,method='euclidean'))^2)/p
h<-0.5;K<-exp(-h*D)
ETA<-list(list(K=K,model='RKHS'))

registerDoParallel(cl)
system.time({
RKHS <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$bcw[id == i] <- NA
    fmRKHS=BGLR(y=bcw_test$bcw,ETA=ETA,nIter=10000,burnIn=2000,verbose = FALSE)
    cor(pheno$bcw[id == i],fmRKHS$yHat[id == i])
  }
}
})
stopImplicitCluster()

#Save result
Acc<-data.frame(unlist(GLBLUP),unlist(BA),unlist(BB),unlist(BC),unlist(BL),unlist(BRR),unlist(RKHS))
colnames(Acc)<-c("GLBLUP","BA","BB","BC","BL","BRR","RKHS")

(summary<-data.frame(Acc_mean=sapply(Acc,function(x) round(mean(x),digits = 3)),
Acc_SE=sapply(Acc,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3))))
```

h2 = 0.308

```

(summary<-data.frame(Acc_mean=sapply(Acc,function(x) round(mean(x/sqrt(h2)),digits = 3)),
Acc_SE=sapply(Acc,function(x) round(sd(x/sqrt(h2))/sqrt(repeats*n.fold),digits = 3)))))

#previous

(summary<-data.frame(Acc_mean=sapply(Acc,function(x) round(mean(x),digits = 3)),
Acc_SE=sapply(Acc,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3)))))

write.csv(Acc,"All_models_Acc.csv")
#Acc<-read.csv("All_models_Acc.xlsx")

#Load data(from this page and deep learning models)
library("readxl")
data<-read_excel("All_models_Acc.xlsx")
Acc_all<-subset(data,select = - c(...1))

(summary<-data.frame(Acc_mean=sapply(Acc_all,function(x) round(mean(x),digits = 3)),
Acc_SE=sapply(Acc_all,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3)))))

write.xlsx(summary, "1.7 summary.xlsx")

# SL

# Platform information
library(benchmarkme)
get_platform_info()$OS.type
get_r_version()$version.string
get_cpu()$model_name;get_cpu()$no_of_cores
get_ram()

# Parallel computation
library(doParallel)
library(foreach)
cl<-makeCluster(16)

geno <-read.table("./Hetero_realigned_cov10_filtered3.raw", header=T)
pheno <- read.csv("./2017heteroPheno.csv", header=T)
attach(pheno)
n<-length(pheno$bew)
#cat IDlist|sed -e "s/S/t/g"|sort -n|sed -e "s/t/S/g"|awk "{print $2}">ID_list.txt

```

```

id<-read.table("ID_list.txt")#fitSID
reorderIndex<-c()
for (i in id$V1){
  reorderIndex<-c(reorderIndex,match(i,rownames(geno)))
}
reorderG<-geno[reorderIndex,]
rownames(reorderG)=1:n
x <- as.matrix(reorderG)-1
data <- data.frame(tbcw=sqrt(pheno$bcb+1),bcw=pheno$bcb,length=pheno$length,gid=1:240)

# Parameters for cross validation
repeats <- 10
n.fold <- 10
n.sample <- length(pheno$bcb)
CM<-7

# Package
library(rrBLUP)
packageVersion("rrBLUP")

# Marker-based relationship matrix (Endelman et al. 2011)
A <- A.mat(x, n.core=8)
row.names(A)=1:240;colnames(A)=1:240

registerDoParallel(cl)
system.time({
  GBLUP<-foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {
      bcb_test <- data
      bcb_test$length[id == i] <- NA
      res <- kin.blup(bcb_test, K=A, geno="gid", pheno="length")
      cor(data$length[id==i],res$pred[id==i])
    }
  }
})
stopImplicitCluster()

```

```

mean(unlist(GLBLUP))

library("BGLR")
packageVersion("BGLR")

registerDoParallel(cl)
system.time({
  BA <- foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="BGLR") %dopar% {
      bcw_test <- data
      bcw_test$length[id == i] <- NA
      fmBA=BGLR(y=bcw_test$length,ETA=list(list(X=x,model='BayesA')),nIter=10000,burnIn=2000,verbose
      = FALSE)
      cor(data$length[id == i],fmBA$yHat[id == i])
    }
  }
})
stopImplicitCluster()

registerDoParallel(cl)
system.time({
  BB <- foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="BGLR") %dopar% {
      bcw_test <- data
      bcw_test$length[id == i] <- NA
      fmBB=BGLR(y=bcw_test$length,ETA=list(list(X=x,model='BayesB')),nIter=10000,burnIn=2000,verbose
      = FALSE)
      cor(data$length[id == i],fmBB$yHat[id == i])
    }
  }
})
stopImplicitCluster()

```

```

registerDoParallel(cl)
system.time({
BC <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$length[id == i] <- NA
    fmBC=BGLR(y=bew_test$length,ETA=list(list(X=x,model='BayesC')),nIter=10000,burnIn=2000,verbose
    = FALSE)
    cor(data$length[id == i],fmBC$yHat[id == i])
  }
}
))
stopImplicitCluster()

registerDoParallel(cl)
system.time({
BL <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$length[id == i] <- NA
    fmBL=BGLR(y=bew_test$length,ETA=list(list(X=x,model='BL')),nIter=10000,burnIn=2000,verbose
    = FALSE)
    cor(data$length[id == i],fmBL$yHat[id == i])
  }
}
))
stopImplicitCluster()

registerDoParallel(cl)
system.time({
BRR <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {

```

```

bcw_test <- data
bcw_test$length[id == i] <- NA
fmBRR=BGLR(y=bcw_test$length,ETA=list(list(X=x,model='BRR')),nIter=10000,burnIn=2000,verbose
= FALSE)
  cor(data$length[id == i],fmBRR$yHat[id == i])
}
}
})
stopImplicitCluster()

# Reproducing kernel
p<-ncol(x)
D<-(as.matrix(dist(x,method='euclidean'))^2)/p
h<-0.5;K<-exp(-h*D)
ETA<-list(list(K=K,model='RKHS'))

registerDoParallel(cl)
system.time({
RKHS <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$length[id == i] <- NA
    fmRKHS=BGLR(y=bcw_test$length,ETA=ETA,nIter=10000,burnIn=2000,verbose = FALSE)
    cor(pheno$length[id == i],fmRKHS$yHat[id == i])
  }
}
})
stopImplicitCluster()

#Save result
Acc<-data.frame(unlist(GLUP),unlist(BA),unlist(BB),unlist(BC),unlist(BL),unlist(BRR),unlist(RKHS))
colnames(Acc)<-c("GLUP","BA","BB","BC","BL","BRR","RKHS")

(summary<-data.frame(Acc_mean=sapply(Acc,function(x) round(mean(x),digits = 3)),
Acc_SE=sapply(Acc,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3))))

```

```

h2 = 0.405

(summary<-data.frame(Acc_mean=sapply(Acc,function(x) round(mean(x/sqrt(h2)),digits = 3)),
Acc_SE=sapply(Acc,function(x) round(sd(x/sqrt(h2))/sqrt(repeats*n.fold),digits = 3)))))

#previous

(summary<-data.frame(Acc_mean=sapply(Acc,function(x) round(mean(x),digits = 3)),
Acc_SE=sapply(Acc,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3)))))

write.csv(Acc,"All_models_AccSL.csv")
#Acc<-read.csv("All_models_AccSL.xlsx")

#Load data(from this page and deep learning models)
library("readxl")
data<-read_excel("All_models_Acc.xlsx")
Acc_all<-subset(data,select = - c(...1))

(summary<-data.frame(Acc_mean=sapply(Acc_all,function(x) round(mean(x),digits = 3)),
Acc_SE=sapply(Acc_all,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3)))))

write.xlsx(summary, "1.7 summary.xlsx")

```

1.2.2 Python script for predictive ability of GP

```
# HC
#!/usr/bin/env python
# # To unify the sampler, we borrow function from R.
repeats = int(10)
nfold = int(10)
nsample = int(240)
cf=int(nsample/nfold)
lf=nsample-cf
import rpy2.robj as robjects
import numpy as np
setseed = robjects.r['set.seed']
sample = robjects.r['sample']
sd = robjects.r["sd"]
import rpy2.robj.numpy2ri
rpy2.robj.numpy2ri.activate()

# # Platform information
import platform
platform.uname()
from tensorflow.python.client import device_lib
device_lib.list_local_devices()
import sys
print (sys.version)

# # Loading data
import pandas as pd
pheno = pd.read_csv('./2017heteroPheno.csv')
#I = pd.read_table('./Hetero_realigned_cov10_filtered3.raw',sep="\s+").as_matrix()-1
I = pd.read_table('./reorderG.txt',sep="\s+").as_matrix()-1
bcw=pheno.bcw.as_matrix()
length=pheno.length.as_matrix()
AccSum1=np.reshape(np.zeros(repeats*nfold),(nfold,repeats))
AccSum2=np.copy(AccSum1);AccSum3=np.copy(AccSum1);AccSum4=np.copy(AccSum1);AccSum5=np.copy(AccSum1);AccSum6=np.copy(AccSum1)
```

```

## Import packages for deep learning models
from scipy.stats.stats import pearsonr
import keras
import tensorflow as tf
from keras.models import Sequential
from keras.layers import Dense, Activation
import time
print("version of tensorflow_gpu:",tf.__version__)
print("version of tensorflow_bakend_keras:",keras.__version__)

import sklearn
from sklearn.svm import SVR
sklearn.__version__

#SVR-linear
start = time.time()
#10-repeat-5-fold Cross validation as same as R
for j in range(repeats):
    setseed(100+3*(j+1)+1)
    id =np.array(sample(robjects.IntVector(np.arange(1,(nsample+1)) %nfold)))
    for i in range(nfold):
        test = np.array(np.where(id==i)).reshape(cf)
        train = np.array(np.where(id!=i)).reshape(lf)
        #Model
        svr_rbf = SVR(kernel='linear',gamma='auto')
        svr_rbf.fit(I[train],bcw[train])
        #Prediction
        pred = svr_rbf.predict(I[test]).reshape(cf)
        pred1 = svr_rbf.predict(I[train]).reshape(lf)
        #Accuracy
        AccSum1[i,j] = pearsonr(pred,bcw[test])[0]
        AccSum2[i,j] = pearsonr(pred1,bcw[train])[0]
    end = time.time()
    print("eplased time",end - start)

print(np.mean(AccSum1),np.mean(AccSum2))
sd(AccSum1)/np.sqrt(repeats*nfold)

```

```

start = time.time()

#10-repeat-5-fold Cross validation as same as R

for j in range(repeats):
    setseed(100+3*(j+1)+1)
    id = np.array(sample(robjects.IntVector(np.arange(1,(nsample+1)) %nfold)))
    for i in range(nfold):
        test = np.array(np.where(id==i)).reshape(cf)
        train = np.array(np.where(id!=i)).reshape(lf)
        #Model
        svr_rbf = SVR(kernel='poly',gamma='auto')
        svr_rbf.fit(I[train],bcw[train])
        #Prediction
        pred = svr_rbf.predict(I[test]).reshape(cf)
        pred1 = svr_rbf.predict(I[train]).reshape(lf)
        #Accuracy
        AccSum1[i,j] = pearsonr(pred,bcw[test])[0]
        AccSum2[i,j] = pearsonr(pred1,bcw[train])[0]
    end = time.time()

print("eplased time",end - start)
print(np.mean(AccSum1),np.mean(AccSum2))
sd(AccSum1)/np.sqrt(repeats*nfold)

start = time.time()

#10-repeat-5-fold Cross validation as same as R

for j in range(repeats):
    setseed(100+3*(j+1)+1)
    id = np.array(sample(robjects.IntVector(np.arange(1,(nsample+1)) %nfold)))
    for i in range(nfold):
        test = np.array(np.where(id==i)).reshape(cf)
        train = np.array(np.where(id!=i)).reshape(lf)
        #Model
        svr_rbf = SVR(kernel='rbf',gamma='auto')
        svr_rbf.fit(I[train],bcw[train])
        #Prediction
        pred = svr_rbf.predict(I[test]).reshape(cf)

```

```

pred1 = svr_rbf.predict(I[train]).reshape(lf)
#Accuracy
AccSum1[i,j] = pearsonr(pred,bcw[test])[0]
AccSum2[i,j] = pearsonr(pred1,bcw[train])[0]
end = time.time()

print("eplased time",end - start)
print(np.mean(AccSum1),np.mean(AccSum2))
sd(AccSum1)/np.sqrt(repeats*nfold)

from sklearn.ensemble import RandomForestRegressor

start = time.time()
#10-repeat-5-fold Cross validation as same as R
for j in range(repeats):
    setseed(100+3*(j+1)+1)
    id = np.array(sample(robjcts.IntVector(np.arange(1,(nsample+1)) %nfold)))
    for i in range(nfold):
        test = np.array(np.where(id==i)).reshape(cf)
        train = np.array(np.where(id!=i)).reshape(lf)
        #Model
        regrf = RandomForestRegressor(n_estimators=2000,max_depth=5,n_jobs=-1,random_state=0)
        regrf.fit(I[train],bcw[train])
        #Prediction
        pred = regrf.predict(I[test]).reshape(cf)
        pred1 = regrf.predict(I[train]).reshape(lf)
        #Accuracy
        AccSum1[i,j] = pearsonr(pred,bcw[test])[0]
        AccSum2[i,j] = pearsonr(pred1,bcw[train])[0]
    end = time.time()
    print(np.mean(AccSum1),np.mean(AccSum2))
    sd(AccSum1)/np.sqrt(repeats*nfold)

# # Neural network
start = time.time()

```

```

#10-repeat-5-fold Cross validation as same as R

for j in range(repeats):
    setseed(100+3*(j+1)+1)
    id =np.array(sample(robjects.IntVector(np.arange(1,(nsample+1)) %nfold)))
    for i in range(nfold):
        test = np.array(np.where(id==i)).reshape(cf)
        train = np.array(np.where(id!=i)).reshape(lf)
        #Model
        model = Sequential()
        model.add(Dense(200, activation='relu', input_dim=6707))
        model.add(Dense(20, activation='relu'))
        model.add(Dense(1))
        #Optimizer
        model.compile(optimizer='Adam', loss='mse')
        #Feed the data
        model.fit(I[train,:],bcw[train],epochs=30, batch_size=128,verbose=0)
        #Prediction
        pred = model.predict(I[test,:],batch_size=None, verbose=0).reshape(cf)
        pred1 = model.predict(I[train,:],batch_size=None, verbose=0).reshape(lf)
        #Accuracy
        AccSum1[i,j] = pearsonr(pred,bcw[test])[0]
        AccSum2[i,j] = pearsonr(pred1,bcw[train])[0]
    end = time.time()

    print("eplased time",end - start)
    #validate on test and train group
    print(np.mean(AccSum1),np.mean(AccSum2))
    sd(AccSum1)/np.sqrt(repeats*nfold)

    # # Multi task deep learning model
    import keras
    from keras.layers import Input, Dense
    from keras.models import Model
    from keras import optimizers

    #Optimizer
    rmsprop=optimizers.RMSprop()

```

```

start = time.time()

#10-repeat-5-fold Cross validation as same as R

for j in range(repeats):
    setseed(100+3*(j+1)+1)
    id = np.array(sample(robjcts.IntVector(np.arange(1,(nsample+1)) %nfold)))
    for i in range(nfold):
        test = np.array(np.where(id==i)).reshape(cf)
        train = np.array(np.where(id!=i)).reshape(lf)
        #Model
        main_input = Input(shape=(6707,), name='main_input')
        x = Dense(200, activation='relu')(main_input)

        x1 = Dense(20, activation='relu')(x)
        y1 = Dense(1,name='y1')(x1)

        x2 = Dense(100, activation='relu')(x)
        y2= Dense(1,name='y2')(x2)

        model = Model(inputs=[main_input], outputs=[y1,y2])
        #Compile
        model.compile(optimizer=rmsprop,loss={'y1': 'mse','y2':'mse'})
        #Train
        model.fit({'main_input': I[train,:]},{'y1': bcw[train],'y2': length[train]},callbacks=None,verbose=0,
                  epochs=30, batch_size=128)
        #Accuracy
        AccSum3[i,j] = pearsonr(model.predict(I[test,:], batch_size=None, verbose=0)[0].reshape(cf),bcw[test])[0]
        AccSum4[i,j] = pearsonr(model.predict(I[train,:], batch_size=None, verbose=0)[0].reshape(lf),bcw[train])[0]
        AccSum5[i,j] = pearsonr(model.predict(I[test,:], batch_size=None, verbose=0)[1].reshape(cf),length[test])[0]
        AccSum6[i,j] = pearsonr(model.predict(I[train,:], batch_size=None,
                                              verbose=0)[1].reshape(lf),length[train])[0]
    end = time.time()

    print("eplased time",end - start)
    #validate on test and train
    print(np.mean(AccSum3),np.mean(AccSum4))
    sd(AccSum3)/np.sqrt(repeats*nfold)

```

```

print(np.mean(AccSum5),np.mean(AccSum6))
sd(AccSum5)/np.sqrt(repeats*nfold)

#save result
import pandas as pd
seven_models_r = pd.read_excel("All_models_Acc.xlsx")
v1=pd.DataFrame(pd.DataFrame(AccSum1).values.flatten(),columns=["NN"],index=range(1,51))
v2=pd.DataFrame(pd.DataFrame(AccSum3).values.flatten(),columns=["MNN"],index=range(1,51))
pd.concat([seven_models_r,v1,v2],axis=1).to_excel('All_models_Acc.xlsx')

# SL
#SL
#!/usr/bin/env python
## To unify the sampler, we borrow function from R.
repeats = int(10)
nfold = int(10)
nsample = int(240)
cf=int(nsample/nfold)
lf=int(nsample-cf)
import rpy2.robj as robj
import numpy as np
setseed = robj.r['set.seed']
sample = robj.r['sample']
sd = robj.r["sd"]
import rpy2.robj.numpy2ri
rpy2.robj.numpy2ri.activate()

## Platform information
import platform
platform.uname()
from tensorflow.python.client import device_lib
device_lib.list_local_devices()
import sys
print (sys.version)

```

```

# # Loading data
import pandas as pd
pheno = pd.read_csv('./2017heteroPheno.csv')
I = pd.read_table('./reorderG.txt',sep="\s+").as_matrix()-1
bcw=pheno.bcw.as_matrix()
length=pheno.length.as_matrix()
AccSum1=np.reshape(np.zeros(repeats*nfold),(nfold,repeats))
AccSum2=np.copy(AccSum1);AccSum3=np.copy(AccSum1);AccSum4=np.copy(AccSum1)

```

```

# # Import packages for deep learning models
from scipy.stats.stats import pearsonr
import keras
import tensorflow as tf
from keras.models import Sequential
from keras.layers import Dense, Activation
import time
print("version of tensorflow_gpu:",tf.__version__)
print("version of tensorflow_bakend_keras:",keras.__version__)

```

```

import sklearn
from sklearn.svm import SVR
sklearn.__version__

#SVR-linear
start = time.time()
#10-repeat-5-fold Cross validation as same as R
for j in range(repeats):
    setseed(100+3*(j+1)+1)
    id =np.array(sample(robjects.IntVector(np.arange(1,(nsample+1)) %nfold)))
    for i in range(nfold):
        test = np.array(np.where(id==i)).reshape(cf)
        train = np.array(np.where(id!=i)).reshape(lf)
        #Model
        svr_rbf = SVR(kernel='linear',gamma='auto')
        svr_rbf.fit(I[train],length[train])
        #Prediction

```

```

pred = svr_rbf.predict(I[test]).reshape(cf)
pred1 = svr_rbf.predict(I[train]).reshape(lf)
#Accuracy
AccSum1[i,j] = pearsonr(pred,length[test])[0]
AccSum2[i,j] = pearsonr(pred1,length[train])[0]
end = time.time()

print("eplased time",end - start)
print(np.mean(AccSum1),np.mean(AccSum2))
sd(AccSum1)/np.sqrt(repeats*nfold)

start = time.time()
#10-repeat-5-fold Cross validation as same as R
for j in range(repeats):
    setseed(100+3*(j+1)+1)
    id = np.array(sample(robjcts.IntVector(np.arange(1,(nsample+1)) %nfold)))
    for i in range(nfold):
        test = np.array(np.where(id==i)).reshape(cf)
        train = np.array(np.where(id!=i)).reshape(lf)
        #Model
        svr_rbf = SVR(kernel='poly',gamma='auto')
        svr_rbf.fit(I[train],length[train])
        #Prediction
        pred = svr_rbf.predict(I[test]).reshape(cf)
        pred1 = svr_rbf.predict(I[train]).reshape(lf)
        #Accuracy
        AccSum1[i,j] = pearsonr(pred,length[test])[0]
        AccSum2[i,j] = pearsonr(pred1,length[train])[0]
    end = time.time()

    print("eplased time",end - start)
    print(np.mean(AccSum1),np.mean(AccSum2))
    sd(AccSum1)/np.sqrt(repeats*nfold)

start = time.time()
#10-repeat-5-fold Cross validation as same as R
for j in range(repeats):
    setseed(100+3*(j+1)+1)

```

```

id =np.array(sample(robjcts.IntVector(np.arange(1,(nsample+1)) %nfold)))
for i in range(nfold):
    test = np.array(np.where(id==i)).reshape(cf)
    train = np.array(np.where(id!=i)).reshape(lf)
    #Model
    svr_rbf = SVR(kernel='rbf',gamma='auto')
    svr_rbf.fit(I[train],length[train])
    #Prediction
    pred = svr_rbf.predict(I[test]).reshape(cf)
    pred1 = svr_rbf.predict(I[train]).reshape(lf)
    #Accuracy
    AccSum1[i,j] = pearsonr(pred,length[test])[0]
    AccSum2[i,j] = pearsonr(pred1,length[train])[0]
end = time.time()

print("eplased time",end - start)
print(np.mean(AccSum1),np.mean(AccSum2))
sd(AccSum1)/np.sqrt(repeats*nfold)

```

```

from sklearn.ensemble import RandomForestRegressor
start = time.time()
#10-repeat-5-fold Cross validation as same as R
for j in range(repeats):
    setseed(100+3*(j+1)+1)
    id =np.array(sample(robjcts.IntVector(np.arange(1,(nsample+1)) %nfold)))
    for i in range(nfold):
        test = np.array(np.where(id==i)).reshape(cf)
        train = np.array(np.where(id!=i)).reshape(lf)
        #Model
        regrf = RandomForestRegressor(n_estimators=2000,max_depth=5,n_jobs=-1,random_state=0)
        regrf.fit(I[train],length[train])
        #Prediction
        pred = regrf.predict(I[test]).reshape(cf)
        pred1 = regrf.predict(I[train]).reshape(lf)
        #Accuracy
        AccSum1[i,j] = pearsonr(pred,length[test])[0]
        AccSum2[i,j] = pearsonr(pred1,length[train])[0]
end = time.time()

```

```

print(np.mean(AccSum1),np.mean(AccSum2))
sd(AccSum1)/np.sqrt(repeats*nfold)

# # Neural network
start = time.time()
#10-repeat-5-fold Cross validation as same as R
for j in range(repeats):
    setseed(100+3*(j+1)+1)
    id = np.array(sample(robjects.IntVector(np.arange(1,(nsample+1)) %nfold)))
    for i in range(nfold):
        test = np.array(np.where(id==i)).reshape(cf)
        train = np.array(np.where(id!=i)).reshape(lf)
        #Model
        model = Sequential()
        model.add(Dense(200, activation='relu', input_dim=6707))
        model.add(Dense(20, activation='relu'))
        model.add(Dense(1))
        #Optimizer
        model.compile(optimizer='Adam', loss='mse')
        #Feed the data
        model.fit(I[train,:],length[train],epochs=30, batch_size=128,verbose=0)
        #Prediction
        pred = model.predict(I[test,:],batch_size=None, verbose=0).reshape(cf)
        pred1 = model.predict(I[train,:],batch_size=None, verbose=0).reshape(lf)
        #Accuracy
        AccSum1[i,j] = pearsonr(pred,length[test])[0]
        AccSum2[i,j] = pearsonr(pred1,length[train])[0]
    end = time.time()

    print("eplased time",end - start)
    #validate on test and train group
    print(np.mean(AccSum1),np.mean(AccSum2))
    sd(AccSum1)/np.sqrt(repeats*nfold)

# # Multi task deep learning model

```

```

import keras
from keras.layers import Input, Dense
from keras.models import Model
from keras import optimizers

#Optimizer
rmsprop=optimizers.RMSprop()

start = time.time()
#10-repeat-5-fold Cross validation as same as R
for j in range(repeats):
    setseed(100+3*(j+1)+1)
    id =np.array(sample(robjcts.IntVector(np.arange(1,(nsample+1)) %nfold)))
    for i in range(nfold):
        test = np.array(np.where(id==i)).reshape(cf)
        train = np.array(np.where(id!=i)).reshape(lf)
        #Model
        main_input = Input(shape=(6707,), name='main_input')
        x = Dense(200, activation='relu')(main_input)

        x1 = Dense(20, activation='relu')(x)
        y1 = Dense(1,name='y1')(x1)

        x2 = Dense(100, activation='relu')(x)
        y2= Dense(1,name='y2')(x2)

        model = Model(inputs=[main_input], outputs=[y1,y2])
        #Compile
        model.compile(optimizer=rmsprop,loss={'y1': 'mse','y2':'mse'})
        #Train
        model.fit({'main_input': I[train,:]},{'y1': length[train],'y2': bcw[train]},
                  epochs=30, batch_size=128,callbacks=None,verbose=0)
        #Accuracy
        AccSum3[i,j] =pearsonr(model.predict(I[test,:], batch_size=None, verbose=0)[0].reshape(cf),length[test])[0]
        AccSum4[i,j] =pearsonr(model.predict(I[train,:],
                                              batch_size=None,
                                              verbose=0)[0].reshape(lf),length[train])[0]
end = time.time()

```

```
print("eplased time",end - start)
#validate on test and train
print(np.mean(AccSum3),np.mean(AccSum4))
sd(AccSum3)/np.sqrt(repeats*nfold)

#save result
import pandas as pd
seven_models_r = pd.read_excel("All_models_Acc.xlsx")
v1=pd.DataFrame(pd.DataFrame(AccSum1).values.flatten(),columns=["NN"],index=range(1,51))
v2=pd.DataFrame(pd.DataFrame(AccSum3).values.flatten(),columns=["MNN"],index=range(1,51))
pd.concat([seven_models_r,v1,v2],axis=1).to_excel('All_models_Acc.xlsx')
```

Section 1.3

1.3.1 R script for simulation

```
library("AlphaSimR")
library("taRifx")
library("readxl")

packageVersion("AlphaSimR")

data<-read_xlsx("FuguMap.xlsx") # FUGU5 cm&bp
lin_map<-read.table("lin.map") # SNP chips. Numbers of SNPs per Chr
pheno <- read.csv("./2017heteroPheno.csv", header=T) # 2017gp data trait
geno <-read.table("./Hetero_realigned_cov10_filtered3.raw", row.names=1, header=T) # 2017gp data genome
attach(pheno)
n<-length(pheno$bcw)
id<-read.table("ID_list.txt")#fitSID
reorderIndex<-c()
for (i in id$V1){
  reorderIndex<-c(reorderIndex,match(i,rownames(geno)))
}
reorderG<-geno[reorderIndex,]
rownames(reorderG)=1:n
x <- as.matrix(reorderG)-1

var(bcw);var(length)

Malecm<-destring(data$Male)
Femalecm<-destring(data$Female)
Bp<-destring(data$`After merging`)
ave_cm<-(Malecm+Femalecm)/2/100
ratio<-sum(Femalecm)/sum(Malecm)

founderPop=runMacs2(10000,nChr = 22, segSites=NULL, Ne=1000, bp=Bp, mutRate = 2.5e-08,
                     inbred = FALSE, ploidy = 2L , nThreads = NULL)

m=1:22
for(i in 1:22){m[i]=dim(lin_map[lin_map$V1==i,])[1]}
sum(m)
```

```

library(rrBLUP)
packageVersion("rrBLUP")
A<- A.mat(x,n.core=8)
row.names(A)=1:240;colnames(A)=1:240

library("sommer")
packageVersion("sommer")
data <- data.frame(tbcw=sqrt(bcw+1),length=length,id=factor(1:240))
attach(data)
ans.m <- mmmer(cbind(tbcw,length)~1,random=~ vs(id, Gu=A, Gtc=unsm(2)),
                 rcov=~ vs(units, Gtc=unsm(2)),
                 data=data)
(sp_cor<-cov2cor(ans.m$sigma$`u:id`))
pin(ans.m, h2~ V1/(V1+V4))
pin(ans.m, h2~ V3/(V3+V6))

#major revision
library("sommer")
packageVersion("sommer")
data <- data.frame(bcw_d_length2=bew/length, length=length,
                   id=factor(1:240))
attach(data)
ans.m <- mmmer(cbind(bcw_d_length2,length)~1,random=~ vs(id, Gu=A, Gtc=unsm(2)),
                 rcov=~ vs(units, Gtc=unsm(2)),
                 data=data)
(sp_cor<-cov2cor(ans.m$sigma$`u:id`))
pin(ans.m, h2~ V1/(V1+V4))
pin(ans.m, h2~ V3/(V3+V6))

h2_bew<-summary(ans.m)$varcomp[1,1]/(summary(ans.m)$varcomp[1,1]+summary(ans.m)$varcomp[4,1])
h2_length<-summary(ans.m)$varcomp[3,1]/(summary(ans.m)$varcomp[3,1]+summary(ans.m)$varcomp[6,1])

round_sp<-2
(sp_h2<-c(round(h2_bew,round_sp),round(h2_length,round_sp)))
(sp_mean<-c(100,round(mean(length),round_sp)))
#(sp_gvar<-
c(round(summary(ans.m)$varcomp[1,1],round_sp),round(summary(ans.m)$varcomp[3,1],round_sp)))      data
transformed

```

```

(sp_gvar<-c(round(var(bcw),round_sp),round(var(length),round_sp))*sp_h2)

SP= SimParam$new(founderPop)
SP$addSnpChip(m)
SP$setGender("yes_rand")
SP$addTraitA(nQtlPerChr =500, mean = sp_mean ,var = sp_gvar,corA = sp_cor )
SP$setVarE(h2=sp_h2)
SP$setRecRatio(round(ratio,round_sp))

round(ratio,round_sp)

pop =newPop(founderPop)

mean(pop@pheno[,1])
mean(pop@pheno[,2])
mean(bcw)
mean(length)
var(pop@pheno[,1])
var(pop@pheno[,2])
var(bcw)
var(length)

# Broodfish pupulation for breeding cycle 0th
set.seed(1)
popr = selectCross(pop, nFemale = 20, nMale = 20, nCrosses = 400, nProgeny = 20 , simParam = SP, use = "rand")
mean(popr@pheno[,1])
mean(popr@pheno[,2])

genMeanc1<-c()
genCorc1<-c()
genVarcc1<-c()
Pvarc1<-c()
for (n in 1:50){
  set.seed(5*n+100)
  pop1=popr[sample(8000,2000)]
  mean(pop1@pheno[,1])
  mean(pop1@pheno[,2])
  popn1<-pop1
}

```

```

genMean1 = meanG(popn1)
genCor1 = cov2cor(genParam(popn1)$varA)[1,2]
genVar1 = c(genParam(popn1)$varA[1,1],genParam(popn1)$varA[2,2])
Pvar1<-c(varP(popn1)[1,1],varP(popn1)[2,2])
for(generation in 1:10){
  popn1 = selectCross(pop =popn1, nFemale=20, nMale=20, use="rand", nCrosses=400,nProgeny = 20)
  set.seed(3*generation+100)
  popn1<- popn1[sample(8000,2000)]
  genMean1 = c(genMean1, meanG(popn1))
  genCor1 = c(genCor1, cov2cor(genParam(popn1)$varA)[1,2])
  genVar1 = c(genVar1,genParam(popn1)$varA[1,1],genParam(popn1)$varA[2,2])
  Pvar1<-c(Pvar1,varP(popn1)[1,1],varP(popn1)[2,2])
}
genMeanc1<-cbind(genMeanc1, genMean1)
genCorc1<-cbind(genCorc1, genCor1)
genVarc1<-cbind(genVarc1, genVar1)
Pvarc1<-cbind(Pvarc1,Pvar1)
}

write.table(genMeanc1,"genMeanc1.csv",sep=";")
write.table(genCorc1,"genCorc1.csv",sep=";")
write.table(genVarc1,"genVarc1.csv",sep=";")
write.table(Pvarc1,"Pvarc1.csv",sep=";")
genMeancM<-read.csv2("genMeanc1.csv")

head(genMeanc1)

genMeancM

Lineplot<-function(x){
line_th_unit<-0.6
library(reshape2)
colnames(x)<-c(1:50)
genMeanc1plot1<-c()
genMeanc1plot2<-c()
for (i in 1:22){
  if((i %% 2) == 0) {
    genMeanc1plot1<-rbind(genMeanc1plot1,x[i,])
  }
}

```

```

} else {
  genMeanc1plot2<-rbind(genMeanc1plot2,x[i,])
}
}

row.names(genMeanc1plot1)<-1:11
row.names(genMeanc1plot2)<-1:11
SLplot<-melt(genMeanc1plot1)
colnames(SLplot)<-c("BCs","Scenarios","values")
PCplot<-melt(genMeanc1plot2)
colnames(PCplot)<-c("BCs","Scenarios","values")
plott<-rbind(SLplot,PCplot)
plot1<-cbind(plott, rep(c("SL","PC"),each=550))
colnames(plot1)[4]<-"trait"
plot<-data.frame("BCs" = as.factor(plot1$BCs),
  "Scenarios"= as.factor(plot1$Scenarios),
  "values" = plot1$values,
  "trait" = as.factor(plot1$trait))

library(ggplot2)
options(repr.plot.width=6, repr.plot.height=4)
y <- ggplot(plot) +
  geom_line(data=plot[1:550,],aes(x=BCs,           y=values,           group=Scenarios),alpha      =
  1/10,color="blue",size=line_th_unit)+ 
  geom_line(data=plot[551:1000,],aes(x=BCs,           y=values/6,           group=Scenarios),alpha      =
  1/4,color="red",size=line_th_unit)+ 
  labs(x = "Broodstock population")+
  scale_x_discrete(labels=c(
    expression("F"[ "0"]),
    expression("F"[ "1"]),
    expression("F"[ "2"]),
    expression("F"[ "3"]),
    expression("F"[ "4"]),
    expression("F"[ "5"]),
    expression("F"[ "6"]),
    expression("F"[ "7"]),
    expression("F"[ "8"]),
    expression("F"[ "9"]),
    expression("F"[ "10"])))+
  scale_y_continuous(

```

```

limits=c(0,25),
name = expression("Average TBV for SL"),
sec.axis = sec_axis(~.*6,name = "Average TBV for HC")+
theme_bw()+
theme(axis.text.y=element_text(size=12,family="",color="black"),
      axis.text.x=element_text(size=12,family="",color="black"),
      text=element_text(size=12, family=""))
y$theme$line$size<-line_th_unit
y$theme$axis.ticks$size<-line_th_unit
y$theme$axis.line$size<-line_th_unit
y$theme$axis.ticks$colour<-"black"
y$theme$panel.grid $ colour<-"white"

return(y)
}

```

(y1<-Lineplot(genMeanc1))

```

genMeanc2<-c()
genCorc2<-c()
genVarc2<-c()
Pvarc2<-c()
gsacccslc<-c()
for (n in 1:50){
  set.seed(5*n+100)
  pop1=popr[sample(8000,2000)]
  mean(pop1@pheno[,1])
  mean(pop1@pheno[,2])
  popn2<-pop1
  genMean2 = meanG(popn2)
  genCor2 = cov2cor(genParam(popn2)$varA)[1,2]
  genVar2 = c(genParam(popn2)$varA[1,1],genParam(popn2)$varA[2,2])
  Pvar2<-c(varP(popn2)[1,1],varP(popn2)[2,2])
  for(generation in 1:10){
    ans2 <- RRBLUP(popn2, traits=c(1, 2), simParam=SP)
    popn2 <- setEBV(popn2, solution=ans2, simParam=SP)
    gsacccslc<-cor(gv(popn2)[,2],ebv(popn2)[,2])
    popn2 = selectCross(pop =popn2, nFemale=20, nMale=20, use="ebv",trait= 2,

```

```

nCrosses=400,nProgeny = 20,selectTop=TRUE)
set.seed(3*generation+100)
popn2<- popn2[sample(8000,2000)]
genMean2 = c(genMean2, meanG(popn2))
genCor2 = c(genCor2, cov2cor(genParam(popn2)$varA)[1,2])
genVar2 = c(genVar2,genParam(popn2)$varA[1,1],genParam(popn2)$varA[2,2])
Pvar2<-c(Pvar2,varP(popn2)[1,1],varP(popn2)[2,2])
}
genMeanc2<-cbind(genMeanc2, genMean2)
genCorc2<-cbind(genCorc2, genCor2)
genVarc2<-cbind(genVarc2, genVar2)
Pvarc2<-cbind(Pvarc2,Pvar2)
gsaccslc<-cbind(gsaccslc,gsaccsl)
}

write.table(genMeanc2,"genMeanc2.csv",sep=";")
#genMeanc2<-read.csv2("genMeanc2.csv")

(y2<-Lineplot(genMeanc2))

genMeanc3<-c()
genCorc3<-c()
genVarc3<-c()
Pvarc3<-c()
gsacchctestc<-c()
gsacchctrainc<-c()

for (n in 1:50){
  set.seed(5*n+100)
  pop1=popr[sample(8000,2000)]
  popn3<-pop1
  genMean3 = meanG(popn3)
  genCor3 = cov2cor(genParam(popn3)$varA)[1,2]
  genVar3 = c(genParam(popn3)$varA[1,1],genParam(popn3)$varA[2,2])
  Pvar3<-c(varP(popn3)[1,1],varP(popn3)[2,2])
  set.seed(5*n+100)
  trainId<-sample(2000,1000)
  popbro60=popn3[-trainId]
}

```

```

popn3<-popn3[trainId]
for(generation in 1:10){
  ans3 <- RRBLUP(popn3,traits=c(1,2), simParam=SP)
  popbro60 <- setEBV(popbro60, solution=ans3, simParam=SP)
  popn3 <- setEBV(popn3, solution=ans3, simParam=SP)
  gsacchctrain<-cor(gv(popn3)[,1],ebv(popn3)[,1])
  gsacchctest<-cor(gv(popbro60)[,1],ebv(popbro60)[,1])
  popn3 = selectCross(pop =popbro60, nFemale=20, nMale=20, use="ebv",trait=1,
                      nCrosses=400,nProgeny = 20,selectTop=FALSE)
  set.seed(3*generation+100)
  popn3r<- popn3[sample(8000,2000)]
  genMean3 = c(genMean3, meanG(popn3r))
  genCor3 = c(genCor3, cov2cor(genParam(popn3r)$varA)[1,2])
  genVar3 = c(genVar3,genParam(popn3r)$varA[1,1],genParam(popn3r)$varA[2,2])
  Pvar3<-c(Pvar3,varP(popn3r)[1,1],varP(popn3r)[2,2])

  trainId<-sample(2000,1000)
  popn3<- popn3r[trainId]
  popbro60<-popn3r[-trainId]
}

genMeanc3<-cbind(genMeanc3, genMean3)
genCorc3<-cbind(genCorc3, genCor3)
genVarc3<-cbind(genVarc3, genVar3)
Pvarc3<-cbind(Pvarc3,Pvar3)
gsacchcteste<-cbind(gsacchcteste,gsacchctest)
gsacchctrainc<-cbind(gsacchctrainc,gsacchctrain)
}

write.table(genMeanc3,"genMeanc3.csv",sep=";")
#genMeanc3<-read.csv3("genMeanc2.csv")

(y3<-Lineplot(genMeanc3))

pcv<-function(bcw,length){
  pcov<-cov(bcw,length)
  pcorm<-matrix(c(var(bcw),pcov,pcov,var(length)),nrow=2)
  return(pcorm)
}

```

```

w<-c(-1,1)

genMeanc4<-c()
genCorc4<-c()
genVarc4<-c()
Pvarc4<-c()
gsaccsmithtestc<-c()
gsaccsmithtrainc<-c()
smith_GSIc<-c()
for (n in 1:50){
  set.seed(5*n+100)
  pop1=popr[sample(8000,2000)]
  popn4<-pop1
  genMean4 = meanG(popn4)
  genCor4 = cov2cor(genParam(popn4)$varA)[1,2]
  genVar4 = c(genParam(popn4)$varA[1,1],genParam(popn4)$varA[2,2])
  Pvar4<-c(varP(popn4)[1,1],varP(popn4)[2,2])
  smith_GSI<-c()

  set.seed(5*n+100)
  trainId<-sample(2000,1000)
  popbro60=popn4[-trainId]
  popn4<-popn4[trainId]
  for(generation in 1:10){
    ans4 <- RRBLUP(popn4,traits=c(1,2), simParam=SP)
    popn4 <- setEBV(popn4, solution=ans4, simParam=SP)
    popbro60 <- setEBV(popbro60, solution=ans4, simParam=SP)

    gsaccsmithtrain<-c(cor(gv(popn4)[,1],ebv(popn4)[,1]),cor(gv(popn4)[,2],ebv(popn4)[,2]))
    gsaccsmithtest<-c(cor(gv(popbro60)[,1],ebv(popbro60)[,1]),cor(gv(popbro60)[,2],ebv(popbro60)[,2]))
    #gcov,pcov,index calculation
    n_popbro60<-length(popbro60@id)
    geno<-pullSnpGeno(popbro60)
    row.names(geno)=1:n_popbro60
    x <- as.matrix(geno)-1
    A <- A.mat(x,n.core=8)
    row.names(A)=1:n_popbro60;colnames(A)=1:n_popbro60
    #data4
    <-
  }
}

```

```

data.frame(bcw=sqrt(popbro60@pheno[,1]+1),length=popbro60@pheno[,2],id=factor(1:n_popbro60))

data4 <- data.frame(bcw=popbro60@pheno[,1],length=popbro60@pheno[,2],id=factor(1:n_popbro60))
ans.m4 <- mmer(cbind(bcw,length)~1,random=~ vs(id, Gu=A, Gtc=unsm(2)),
               rcov=~ vs(units, Gtc=unsm(2)),
               data=data4,verbose=FALSE)
p4<-pcv(popbro60@pheno[,1],popbro60@pheno[,2])
index<-smithHazel(w,ans.m4$sigma$`u:id`,p4)

smith_GSI<-c(smith_GSI,index)
popbro60@ebv<-cbind(popbro60@ebv,
                      matrix((index[1]*popbro60@ebv[,1]+index[2]*popbro60@ebv[,2]),nrow=1000, ncol=1))
popn4 = selectCross(pop =popbro60, nFemale=20, nMale=20, use="ebv",trait=3,
                     nCrosses=400,nProgeny = 20,selectTop=TRUE)
set.seed(3*generation+100)
popn4r<- popn4[sample(8000,2000)]
genMean4 = c(genMean4, meanG(popn4r))

genCor4 = c(genCor4, cov2cor(genParam(popn4r)$varA)[1,2])
genVar4 = c(genVar4,genParam(popn4r)$varA[1,1],genParam(popn4r)$varA[2,2])
Pvar4<-c(Pvar4,varP(popn4r)[1,1],varP(popn4r)[2,2])
trainId<-sample(2000,1000)
popn4<- popn4r[trainId]
popbro60<-popn4r[-trainId]
}

genMeanc4<-cbind(genMeanc4, genMean4)
genCorc4<-cbind(genCorc4, genCor4)
genVarc4<-cbind(genVarc4, genVar4)
Pvarc4<-cbind(Pvarc4,Pvar4)
gsaccsmithtestc<-cbind(gsaccsmithtestc,gsaccsmithtest)
gsaccsmithtrainc<-cbind(gsaccsmithtrainc,gsaccsmithtrain)
smith_GSIc<-cbind(smith_GSIc,smith_GSI)
}

write.table(genMeanc4,"genMeanc4.csv",sep=";")
#genMeanc3<-read.csv3("genMeanc2.csv")

(y4<-Lineplot(genMeanc4))

```

```

DGLGSI<-function(P,G,d){
  i<-solve(P)%%G%*%solve(G%*%solve(P)%*%G)%*%d
  return(i)
}

(d=matrix(c(-3,0.3),2,1))

genMeanc5<-c()
genCorc5<-c()
genVarc5<-c()
Pvarc5<-c()
gsaccDGtestc<-c()
gsaccDGtrainc<-c()
DG_GSIc<-c()
for (n in 1:50){
  set.seed(5*n+100)
  pop1=popr[sample(8000,2000)]
  popn5<-pop1
  genMean5 = meanG(popn5)
  genCor5 = cov2cor(genParam(popn5)$varA)[1,2]
  genVar5 = c(genParam(popn5)$varA[1,1],genParam(popn5)$varA[2,2])
  Pvar5<-c(varP(popn5)[1,1],varP(popn5)[2,2])
  DG_GSI<-c()

  set.seed(5*n+100)
  trainId<-sample(2000,1000)
  popbro60=popn5[-trainId]
  popn5<-popn5[trainId]
  for(generation in 1:10){
    ans5 <- RRBLUP(popn5,traits=c(1,2), simParam=SP)
    popn5 <- setEBV(popn5, solution=ans5, simParam=SP)
    popbro60 <- setEBV(popbro60, solution=ans5, simParam=SP)

    gsaccDGtrain<-c(cor(gv(popn5)[,1],ebv(popn5)[,1]),cor(gv(popn5)[,2],ebv(popn5)[,2]))
    gsaccDGtest<-c(cor(gv(popbro60)[,1],ebv(popbro60)[,1]),cor(gv(popbro60)[,2],ebv(popbro60)[,2]))
    #gcov,pcov,index calculation
  }
}

```

```

n_popbro60<-length(popbro60@id)
geno<-pullSnpGeno(popbro60)
row.names(geno)=1:n_popbro60
x <- as.matrix(geno)-1
A <- A.mat(x,n.core=8)
row.names(A)=1:n_popbro60;colnames(A)=1:n_popbro60
#If transformed data was use, the genetic va(co)riance matrix will not be estimated correctly.
#                                     data5                                     <-
data.frame(bcw=sqrt(popbro60@pheno[,1]+1),length=popbro60@pheno[,2],id=factor(1:n_popbro60))
data5 <- data.frame(bcw=popbro60@pheno[,1],length=popbro60@pheno[,2],id=factor(1:n_popbro60))
ans.m5 <- mmmer(cbind(bcw,length)~1,random=~ vs(id, Gu=A, Gtc=unsm(2)),
                 rcov=~ vs(units, Gtc=unsm(2)),
                 data=data5,verbose=FALSE)
p5<-pcv(popbro60@pheno[,1],popbro60@pheno[,2])
index<-DGLGSI(p5,ans.m5$sigma$`u:id`,d)

DG_GSI<-c(DG_GSI,index)
popbro60@ebv<-cbind(popbro60@ebv,
                      matrix((index[1]*popbro60@ebv[,1]+index[2]*popbro60@ebv[,2]),nrow=1000, ncol=1))
popn5 = selectCross(pop =popbro60, nFemale=20, nMale=20, use="ebv",trait=3,
                     nCrosses=400,nProgeny = 20,selectTop=TRUE)
set.seed(3*generation+100)
popn5r<- popn5[sample(8000,2000)]
genMean5 = c(genMean5, meanG(popn5r))

genCor5 = c(genCor5, cov2cor(genParam(popn5r)$varA)[1,2])
genVar5 = c(genVar5,genParam(popn5r)$varA[1,1],genParam(popn5r)$varA[2,2])
Pvar5<-c(Pvar5,varP(popn5r)[1,1],varP(popn5r)[2,2])
trainId<-sample(2000,1000)
popn5<- popn5r[trainId]
popbro60<-popn5r[-trainId]
}

genMeanc5<-cbind(genMeanc5, genMean5)
genCorc5<-cbind(genCorc5, genCor5)
genVarc5<-cbind(genVarc5, genVar5)
Pvarc5<-cbind(Pvarc5,Pvar5)
gsaccDGtestc<-cbind(gsaccDGtestc,gsaccDGtest)
gsaccDGtrainc<-cbind(gsaccDGtrainc,gsaccDGtrain)

```

```

DG_GSIc<-cbind(DG_GSIc,DG_GSI)
}

write.table(genMeanc5,"genMeanc5.csv",sep=";")
#genMeanc3<-read.csv3("genMeanc2.csv")

(y5<-Lineplot(genMeanc5))

(d=matrix(c(-1,1),2,1))

genMeanc6<-c()
genCorc6<-c()
genVarc6<-c()
Pvarc6<-c()
gsaccDGtestc<-c()
gsaccDGtrainc<-c()
DG_GSIc<-c()
for (n in 1:50){
  set.seed(5*n+100)
  pop1=popr[sample(8000,2000)]
  popn6<-pop1
  genMean6 = meanG(popn6)
  genCor6 = cov2cor(genParam(popn6)$varA)[1,2]
  genVar6 = c(genParam(popn6)$varA[1,1],genParam(popn6)$varA[2,2])
  Pvar6<-c(varP(popn6)[1,1],varP(popn6)[2,2])
  DG_GSI<-c()

  set.seed(5*n+100)
  trainId<-sample(2000,1000)
  popbro60=popn6[-trainId]
  popn6<-popn6[trainId]
  for(generation in 1:10){
    ans6 <- RRBLUP(popn6,traits=c(1,2), simParam=SP)
    popn6 <- setEBV(popn6, solution=ans6, simParam=SP)
    popbro60 <- setEBV(popbro60, solution=ans6, simParam=SP)

    gsaccDGtrain<-c(cor(gv(popn6)[,1],ebv(popn6)[,1]),cor(gv(popn6)[,2],ebv(popn6)[,2]))
    gsaccDGtest<-c(cor(gv(popbro60)[,1],ebv(popbro60)[,1]),cor(gv(popbro60)[,2],ebv(popbro60)[,2]))
  }
}

```

```

#gcov,pcov,index calculation
n_popbro60<-length(popbro60@id)
geno<-pullSnpGeno(popbro60)
row.names(geno)=1:n_popbro60
x <- as.matrix(geno)-1
A <- A.mat(x,n.core=8)
row.names(A)=1:n_popbro60;colnames(A)=1:n_popbro60
#If transformed data was use, the genetic va(co)riance matrix will not be estimated correctly.
#                                     data6                                     <-
data.frame(bcw=sqrt(popbro60@pheno[,1]+1),length=popbro60@pheno[,2],id=factor(1:n_popbro60))
data6 <- data.frame(bcw=popbro60@pheno[,1],length=popbro60@pheno[,2],id=factor(1:n_popbro60))
ans.m6 <- mmmer(cbind(bcw,length)~1,random=~ vs(id, Gu=A, Gtc=unsm(2)),
                 rcov=~ vs(units, Gtc=unsm(2)),
                 data=data6,verbose=FALSE)
p6<-pcv(popbro60@pheno[,1],popbro60@pheno[,2])
index<-DGLGSI(p6,ans.m6$sigma$`u:id`,d)

DG_GSI<-c(DG_GSI,index)
popbro60@ebv<-cbind(popbro60@ebv,
                      matrix((index[1]*popbro60@ebv[,1]+index[2]*popbro60@ebv[,2]),nrow=1000, ncol=1))
popn6 = selectCross(pop =popbro60, nFemale=20, nMale=20, use="ebv",trait=3,
                     nCrosses=400,nProgeny = 20,selectTop=TRUE)
set.seed(3*generation+100)
popn6r<- popn6[sample(8000,2000)]
genMean6 = c(genMean6, meanG(popn6r))

genCor6 = c(genCor6, cov2cor(genParam(popn6r)$varA)[1,2])
genVar6 = c(genVar6,genParam(popn6r)$varA[1,1],genParam(popn6r)$varA[2,2])
Pvar6<-c(Pvar6,varP(popn6r)[1,1],varP(popn6r)[2,2])
trainId<-sample(2000,1000)
popn6<- popn6r[trainId]
popbro60<-popn6r[-trainId]
}
genMeanc6<-cbind(genMeanc6, genMean6)
genCorc6<-cbind(genCorc6, genCor6)
genVarcc6<-cbind(genVarcc6, genVar6)
Pvarcc6<-cbind(Pvarcc6,Pvar6)
gsaccDGtestc<-cbind(gsaccDGtestc,gsaccDGtest)

```

```

gsaccDGtrainc<-cbind(gsaccDGtrainc,gsaccDGtrain)
DG_GSIc<-cbind(DG_GSIc,DG_GSI)
}

write.table(genMeanc6,"genMeanc6.csv",sep=";")

(y6<-Lineplot(genMeanc6))

w<-c(-3,0.3)

genMeanc7<-c()
genCorc7<-c()
genVarc7<-c()
Pvarc7<-c()
gsaccsmithtestc<-c()
gsaccsmithtrainc<-c()
smith_GSIc<-c()
for (n in 1:50){
set.seed(5*n+100)
pop1=popr[sample(8000,2000)]
popn7<-pop1
genMean7 = meanG(popn7)
genCor7 = cov2cor(genParam(popn7)$varA)[1,2]
genVar7 = c(genParam(popn7)$varA[1,1],genParam(popn7)$varA[2,2])
Pvar7<-c(varP(popn7)[1,1],varP(popn7)[2,2])
smith_GSI<-c()

set.seed(5*n+100)
trainId<-sample(2000,1000)
popbro60=popn7[-trainId]
popn7<-popn7[trainId]
for(generation in 1:10){
  ans7 <- RRBLUP(popn7,traits=c(1,2), simParam=SP)
  popn7 <- setEBV(popn7, solution=ans7, simParam=SP)
  popbro60 <- setEBV(popbro60, solution=ans7, simParam=SP)

  gsaccsmithtrain<-c(cor(gv(popn7)[,1],ebv(popn7)[,1]),cor(gv(popn7)[,2],ebv(popn7)[,2]))
}
}

```

```

gsaccsmithtest<-c(cor(gv(popbro60)[,1],ebv(popbro60)[,1]),cor(gv(popbro60)[,2],ebv(popbro60)[,2]))
#gcov,pcov,index calculation
n_popbro60<-length(popbro60@id)
geno<-pullSnpGeno(popbro60)
row.names(geno)=1:n_popbro60
x <- as.matrix(geno)-1
A <- A.mat(x,n.core=8)
row.names(A)=1:n_popbro60;colnames(A)=1:n_popbro60
#data7
data.frame(bcw=sqrt(popbro60@pheno[,1]+1),length=popbro60@pheno[,2],id=factor(1:n_popbro60)) <-

```

```

data7 <- data.frame(bcw=popbro60@pheno[,1],length=popbro60@pheno[,2],id=factor(1:n_popbro60))
ans.m7 <- mmmer(cbind(bcw,length)~1,random=~ vs(id, Gu=A, Gtc=unsm(2)),
rcov=~ vs(units, Gtc=unsm(2)),
data=data7,verbose=FALSE)
p7<-pcv(popbro60@pheno[,1],popbro60@pheno[,2])
index<-smithHazel(w,ans.m7$sigma$`u:id`,p7)

```

```

smith_GSI<-c(smith_GSI,index)
popbro60@ebv<-cbind(popbro60@ebv,
matrix((index[1]*popbro60@ebv[,1]+index[2]*popbro60@ebv[,2]),nrow=1000, ncol=1))
popn7 = selectCross(pop =popbro60, nFemale=20, nMale=20, use="ebv",trait=3,
nCrosses=700,nProgeny = 20,selectTop=TRUE)
set.seed(3*generation+100)
popn7r<- popn7[sample(8000,2000)]
genMean7 = c(genMean7, meanG(popn7r))

```

```

genCor7 = c(genCor7, cov2cor(genParam(popn7r)$varA)[1,2])
genVar7 = c(genVar7,genParam(popn7r)$varA[1,1],genParam(popn7r)$varA[2,2])
Pvar7<-c(Pvar7,varP(popn7r)[1,1],varP(popn7r)[2,2])
trainId<-sample(2000,1000)
popn7<- popn7r[trainId]
popbro60<-popn7r[-trainId]
}
genMeanc7<-cbind(genMeanc7, genMean7)
genCorc7<-cbind(genCorc7, genCor7)
genVarc7<-cbind(genVarc7, genVar7)
Pvarc7<-cbind(Pvarc7,Pvar7)

```

```

gsaccsmithtestc<-cbind(gsaccsmithtestc,gsaccsmithtest)
gsaccsmithtrainc<-cbind(gsaccsmithtrainc,gsaccsmithtrain)
smith_GSIc<-cbind(smith_GSIc,smith_GSI)
}

(y7<-Lineplot(genMeanc7))
write.table(genMeanc7,"genMeanc7.csv",sep=";")

library("cowplot")
options(repr.plot.width=16, repr.plot.height=10)
all<- ggdraw() +
  draw_plot(y1, x = 0, y = 0.64, width = 0.32, height = 0.32) +
  draw_plot(y2, x = 0.32, y = 0.64, width = 0.32, height = 0.32) +
  draw_plot(y3, x = 0, y = 0.32, width = 0.32, height = 0.32) +
  draw_plot(y4, x = 0.32, y = 0.32, width = 0.32, height = 0.32) +
  draw_plot(y7, x = 0, y = 0, width = 0.32, height = 0.32) +
  draw_plot(y5, x = 0.32, y = 0, width = 0.32, height = 0.32) +
  draw_plot_label(label    =   c("RND",    "GS[SL]",    "GS[HC]", "S1[SHI]",    "S2[SHI]", "S[DGI]"),    size
=12,family="serif",
  x = c(0.04, 00.36, 0.04, 0.36,0.04,0.36), y = c(0.94, 0.94, 0.62, 0.62,0.30,0.30),parse =TRUE)
all
ggsave("Fig4.pdf",device="pdf",units="in",font="Helvetica",width=16, height=10)

library("ggpubr")
library(extrafont)

figure <- ggarrange(y1,y2,y3,y4,y5,
  labels = c("a","b","c","d","e"),
  ncol = 2, nrow = 3,font.label = list(size = 12, color = "black", face = "bold", family = ""))
  
figure

ggsave("Fig4.pdf",device="pdf",units="in",font="Helvetica",width =7.2)

```

Section 2.1

2.1.1 R script for artificial infection

```
R.Version()$version.string  
library("ggplot2")  
library("ggpubr")  
library("extrafont")  
  
print(paste("ggplot2 version",packageVersion("ggplot2")))  
print(paste("ggpubr version",packageVersion("ggpubr")))  
print(paste("extrafont version",packageVersion("extrafont")))  
# remotes::install_version("Rttf2pt1", version = "1.3.8")  
# ttf_import()  
# fonts()  
# loadfonts(device = "pdf")  
# extrafont::loadfonts(device = "postscript")  
  
# Loading phenotypes  
pheno <- read.table("./Lin2018_phenotype.txt", header=T)  
colnames(pheno)<-c("ID","SID","length","BW","bcw","PostTank","PreTank","TankName")  
pheno$HD<-100*pheno$bcw/pheno$length^2  
head(pheno)  
str(pheno)  
attach(pheno)  
  
# Mean & S.D.  
meanSD <- data.frame(bcw,length,HD, BW)  
as.data.frame( t(sapply(meanSD, function(cl)  
list(mean=round(mean(cl,na.rm=TRUE),2),sd=round(sd(cl,na.rm=TRUE),2))))  
  
# Normality test  
shapiro.test(bcw)  
shapiro.test(length)  
shapiro.test(HD)  
shapiro.test(BW)  
shapiro.test(sqrt(bcw+1))  
shapiro.test(sign(bcw) * abs(bcw)^(1/3))  
shapiro.test(log(bcw+1))
```

```

cor(bcw, length, method = "kendall")

cor.test(bcw, length, method = "kendall")

library(caret)

bctbcw<-BoxCoxTrans(bcw+1)
bctbcw
predictbcw<-predict(bctbcw,bcw+1) # log(bcw+1)
# normal transformation
pheno["tbcw"]<-predictbcw
p3<-ggplot(pheno, aes(x=tbcw))+
  ggpibr::theme_pubr(base_family="Times New Roman",base_size = 12)+
  geom_histogram(aes(y = ..density..),color="black",fill="grey", binwidth=0.4,size=line_th_unit)+
  labs(x="Transformed HC", y="Density")+
  geom_density(size=line_th_unit)
p3$theme$line$size<-line_th_unit
p3$theme$line$size<-line_th_unit
p3$theme$axis.ticks$size<-line_th_unit
p3$theme$axis.line$size<-line_th_unit
p3$theme$panel.grid $ colour<-"black"
p3$theme$axis.ticks$colour<-"black"
#p3

# Pearson's r
cor.test(bcw,length)
cor.test(predictbcw,length)

cor.test(length,BW)
# round(cor(bcw,length),2)

# H. okamotoi count
line_th_unit<-0.6 ##size unit, mm
scaleFUN <- function(x) sprintf("%.2f", x)
options(repr.plot.width=4, repr.plot.height=4)
p1<-ggplot(pheno, aes(x=bcw))+
  ggpibr::theme_pubr(base_family="Times New Roman",base_size = 12)+
  geom_histogram(aes(y = ..density..),color="black",fill="grey" ,binwidth=1,size=line_th_unit)+
  geom_density(size=line_th_unit)+
```

```

  labs(x="HC", y="Density")
  p1$theme$line$size<-line_th_unit
  p1$theme$line$size<-line_th_unit
  p1$theme$axis.ticks$size<-line_th_unit
  p1$theme$axis.line$size<-line_th_unit
  p1$theme$panel.grid$colour<-"black"
  p1$theme$axis.ticks$colour<-"black"
# p1

# Standard length
p2<-ggplot(pheno, aes(x=length))+  

  ggpibr::theme_pubr(base_family="Times New Roman",base_size = 12)+  

  geom_histogram(aes(y = ..density..),color="black",fill="grey", binwidth=0.1,size=line_th_unit)+  

  labs(x="SL", y="Density")+
  geom_density(size=line_th_unit)+  

  labs(x="SL", y="Density")
p2$theme$line$size<-line_th_unit
p2$theme$line$size<-line_th_unit
p2$theme$axis.ticks$size<-line_th_unit
p2$theme$axis.line$size<-line_th_unit
p2$theme$panel.grid $ colour<- "black"
p2$theme$axis.ticks$colour<-"black"
# p2

options(repr.plot.width=6, repr.plot.height=6)
figure <- ggpibr::ggarrange(p1, p2,
  labels = c("a", "b"),
  ncol = 1,
  nrow = 3,
  font.label = list(size = 14,
    color = "black",
    face = "plain",
    family = "Times New Roman"))
figure

ggsave("Fig. 2.1.pdf",device="pdf",units="in",width =6, height =6 )
# ggsave("Fig1.eps",device="eps",units="in",width =7.2)
# ggsave("Fig. 1.1.tiff", width=7.2, units="in", res=300)

```

2.1.2 Bash script for genotyping

```
#!/bin/bash  
#sh TRIM.sh 1>TRIM.log 2>TRIM.err
```

```
TRIMMO=/home/lin/Lin2018_hetero/soft/Trimmomatic-0.39/trimmomatic-0.39.jar
```

```
DIR=/home/lin/Lin2018_hetero/  
FASTQ=${DIR}/201912_60_01_1  
ls ${FASTQ} | sed -e "s/_H/t/g" | cut -f1 | uniq -d | cut -f2 > ${DIR}/CODES/sample.list
```

```
for i in `cat ${DIR}/CODES/sample.list`
```

```
do
```

```
mkdir ${DIR}/TRIM/${i}
```

```
TRIM=${DIR}/TRIM/${i}
```

```
java -jar ${TRIMMO} \
```

```
PE -threads 32 \
```

```
 ${FASTQ}/${i}_*_1.fq.gz ${FASTQ}/${i}_*_2.fq.gz \
```

```
 ${TRIM}/${i}"_paired_R1.fq.gz" ${TRIM}/${i}"_unpaired_R1.fq.gz" \
```

```
 ${TRIM}/${i}"_paired_R2.fq.gz" ${TRIM}/${i}"_unpaired_R2.fq.gz" \
```

```
 ILLUMINACLIP:/home/lin/Lin2018_hetero/soft/Trimmomatic-0.39/adapters/TruSeq2-PE.fa:2:30:10
```

```
 SLIDINGWINDOW:30:20 AVGQUAL:20
```

```
Done
```

```
#!/bin/bash
```

```
#sh TRIM.sh 1>TRIM.log 2>TRIM.err
```

```
TRIMMO=/home/lin/Lin2018_hetero/soft/Trimmomatic-0.39/trimmomatic-0.39.jar
```

```
DIR=/home/lin/Lin2018_hetero/
```

```
FASTQ=${DIR}/201912_61_01_1
```

```
ls ${FASTQ} | sed -e "s/_H/t/g" | cut -f1 | uniq -d | cut -f2 > ${DIR}/CODES/sample2.list
```

```

for i in `cat ${DIR}/CODES/sample2.list`
do
mkdir ${DIR}/TRIM/${i}
TRIM=${DIR}/TRIM/${i}

java -jar ${TRIMMO} \
PE -threads 32 \
${FASTQ}/${i}_*_1.fq.gz ${FASTQ}/${i}_*_2.fq.gz \
${TRIM}/${i}"_paired_R1.fq.gz" ${TRIM}/${i}"_unpaired_R1.fq.gz" \
${TRIM}/${i}"_paired_R2.fq.gz" ${TRIM}/${i}"_unpaired_R2.fq.gz" \
ILLUMINACLIP:/home/lin/Lin2018_hetero/soft/Trimmomatic-0.39/adapters/TruSeq2-PE.fa:2:30:10
SLIDINGWINDOW:30:20 AVGQUAL:20
Done

```

```

#!/bin/bash
#sh TRIM.sh 1>>TRIM.log 2>>TRIM.err

```

```
TRIMMO=/home/lin/Lin2018_hetero/soft/Trimmomatic-0.39/trimmomatic-0.39.jar
```

```

DIR=/home/lin/Lin2018_hetero/
FASTQ=${DIR}/201911_11_01_1
ls ${FASTQ} | sed -e "s/_H/t/g" | cut -f1 | uniq -d | cut -f2 > ${DIR}/CODES/sample3.list

```

```

for i in `cat ${DIR}/CODES/sample3.list`
do
mkdir ${DIR}/TRIM/${i}
TRIM=${DIR}/TRIM/${i}

java -jar ${TRIMMO} \
PE -threads 32 \
${FASTQ}/${i}_*_1.fq.gz ${FASTQ}/${i}_*_2.fq.gz \
${TRIM}/${i}"_paired_R1.fq.gz" ${TRIM}/${i}"_unpaired_R1.fq.gz" \
${TRIM}/${i}"_paired_R2.fq.gz" ${TRIM}/${i}"_unpaired_R2.fq.gz" \
ILLUMINACLIP:/home/lin/Lin2018_hetero/soft/Trimmomatic-0.39/adapters/TruSeq2-PE.fa:2:30:10

```

```
SLIDINGWINDOW:30:20 AVGQUAL:20
```

```
Done
```

```
#!/bin/bash  
#sh BWA.sh 1>BWA.log 2>BWA.err  
  
DIR=/home/lin/Lin2018_hetero/  
REF=/home/lin/Lin2018_hetero/reference/fr3.fa
```

```
bwa index ${REF}  
samtools faidx ${REF}
```

```
mkdir ${DIR}/BWA/
```

```
for i in `cat ${DIR}/CODES/sample4.list`  
do  
mkdir ${DIR}/BWA/${i}  
BWA=${DIR}/BWA/${i}  
TRIM=${DIR}/TRIM/${i}
```

```
bwa mem -t 32 -M ${REF} ${TRIM}/${i}_paired_R1.fq.gz ${TRIM}/${i}_paired_R2.fq.gz -R  
"@RG\tID:\"$i\"\tSM:\"$i\"\tPL:Illumina" | samtools view -b -F 2308 -q 10 -@ 32 -o ${BWA}/${i}.bam"  
samtools sort -n -@ 32 -o ${BWA}/${i}_namesort.bam ${BWA}/${i}.bam  
samtools fixmate -@ 32 -m ${BWA}/${i}_namesort.bam ${BWA}/${i}_fixmate.bam  
samtools sort -@ 32 -o ${BWA}/${i}_sorted.bam ${BWA}/${i}_fixmate.bam  
samtools index ${BWA}/${i}_sorted.bam
```

```
rm ${BWA}/${i}.sam  
rm ${BWA}/${i}.bam  
rm ${BWA}/${i}_namesort.bam  
rm ${BWA}/${i}_fixmate.bam
```

```
done
```

```
#!/bin/bash  
#sh GATK_GRAS_parallel.sh 1>GATK_GRAS_parallel.log 2>GATK_GRAS_parallel.err
```

```

#PBS -N gatk
#PBS -l select=1:ncpus=8:mem=15gb,walltime=2:00:00
#PBS -j oe

echo "START -----"

#module add java/12.0.2
#module load parallel
#module load gatk

DIR=/home/lin/Lin2018_hetero
REF=/home/lin/Lin2018_hetero/reference/fr3.fa
gatk=/home/lin/gatk-4.1.6.0/gatk

### Use gnu-parallel to use multiple cores
### within one script
cat ${DIR}/CODES/sample4.list | parallel --verbose -j 6 "${gatk}" HaplotypeCaller --output-mode EMIT_ALL_CONFIDENT_SITES -stand-call-conf 30 -R ${REF} -I ${DIR}/BWA/{}_{}/{}_sorted.bam -O ${DIR}/BWA/{}_{}/{}_pe_variants_gatk.g.vcf.gz -ERC GVCF"

echo "FINISH -----"

## Bash

## Call variants per-sample (GVCF mode)

DIR=/home/lin/GS2018_SNPcalling/
REF=/home/lin/GS2018_SNPcalling/reference/fr3.fa
gatk=/home/lin/gatk-4.1.6.0/gatk

#rm ${DIR}/BWA/*/*_pe_variants_gatk.g.*
cp ${DIR}/BWA/*/*_pe_variants_gatk.g.* ${DIR}/VCF
##sudo chmod +x *

echo "${gatk} CombineGVCFs" > ${DIR}/myscript/CombineGVCFs.sh

```

```

for i in `cat ${DIR}/CODES/sample4.list`
do
echo "-V ${DIR}/VCF/"${i}"_pe_variants_gatk.g.vcf.gz" >> ${DIR}/myscript/CombineGVCFs.sh
done
echo "-R ${REF}" >> ${DIR}/myscript/CombineGVCFs.sh
echo "-O ${DIR}/VCF/cohort.g.vcf.gz" >> ${DIR}/myscript/CombineGVCFs.sh

cat ${DIR}/myscript/CombineGVCFs.sh | tr "\n" " " > ${DIR}/myscript/Spa_CombineGVCFs.sh

.${DIR}/myscript/Spa_CombineGVCFs.sh

tabix -p vcf ${DIR}/VCF/cohort.g.vcf.gz

${gatk}      GenotypeGVCFs      -R      ${REF}      -V      ${DIR}/VCF/cohort.g.vcf.gz      -O
${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --tmp-dir=${DIR}/VCF/temp

#bgzip -d ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz
#grep "##contig" ${DIR}/VCF/Output_jonit_call_cohort.vcf | sed -e "s/=/. /g" -e "s/,/. /g" | awk '{print $3}' >
${DIR}/VCF/pos.list
#cat -n ${DIR}/VCF/pos.list | awk '{print $2"\t"$1}' > ${DIR}/VCF/Chom.map
#bgzip ${DIR}/VCF/Output_jonit_call_cohort.vcf
tabix -p vcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz

```

DIR=/home/lin/GS2018_SNPCalling/

```

# SNP filtering test 1
vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --chr chr1 --chr chr2 --chr chr3 --chr chr4 --chr
chr5 --chr chr6 --chr chr7 --chr chr8 --chr chr9 --chr chr10 --chr chr11 --chr chr12 --chr chr13 --chr chr14 --chr
chr15 --chr chr16 --chr chr17 --chr chr18 --chr chr19 --chr chr20 --chr chr21 --chr chr22 --remove-indels --min-
alleles 2 --max-alleles 2 --minQ 20 --min-meanDP 20 --minDP 10 --max-meanDP 300 --maf 0.01 --hwe 0.05 --
max-missing 0.7
# kept 12548 out of a possible 2196091 Sites

```

```

#
mkdir ${DIR}/Post_VCF_210922
filter=${DIR}/Post_VCF_210922
task(){
mkdir ${filter}/${i};

```

```

vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --chr chr1 --chr chr2 --chr chr3 --chr chr4 --chr
chr5 --chr chr6 --chr chr7 --chr chr8 --chr chr9 --chr chr10 --chr chr11 --chr chr12 --chr chr13 --chr chr14 --chr
chr15 --chr chr16 --chr chr17 --chr chr18 --chr chr19 --chr chr20 --chr chr21 --chr chr22 --remove-indels --min-
alleles 2 --max-alleles 2 --minQ 20 --min-meanDP 20 --minDP 10 --max-meanDP 300 --maf 0.01 --hwe 0.05 --
max-missing ${i} --recode --stdout > ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf;
zgrep -v "##" ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf | awk '{print $1"\t"$2}' >
${filter}/${i}/position.list;
cat -n ${filter}/${i}/position.list | awk '{print $2"\t"$1}' > ${filter}/${i}/Chom.map;
bgzip ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf;
tabix -p vcf ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf.gz;
vcftools --gzvcf ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf.gz --plink --chrom-map
${filter}/${i}/Chom.map --out ${filter}/${i}/Hetero_realigned_cov10_filtered;
plink1 --ped ${filter}/${i}/Hetero_realigned_cov10_filtered.ped --map
${filter}/${i}/Hetero_realigned_cov10_filtered.map --recodeA --out
${filter}/${i}/Hetero_realigned_cov10_filtered --noweb;
echo "empty" > ${filter}/${i}/IDlist;
zgrep "#CHROM" ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf.gz | cut -f10- | sed -e "s/\t\n/g" >>
${filter}/${i}/IDlist;
sed -e "s/empty//g" ${filter}/${i}/IDlist > ${filter}/${i}/IDs.txt
cat ${filter}/${i}/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- >
${filter}/${i}/Hetero_realigned_cov10_filtered2.raw;
paste -d " " ${filter}/${i}/IDs.txt ${filter}/${i}/Hetero_realigned_cov10_filtered2.raw >
${filter}/${i}/Hetero_realigned_cov10_filtered3.raw;
sed -n "1p" ${filter}/${i}/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- | sed -e "s/ \n/g" >
${filter}/${i}/Hetero_realigned_cov10_filtered.pos;
less ${filter}/${i}/Hetero_realigned_cov10_filtered.pos|sed -e "s/_/\t/g"|sed -e "s:/\t/g"|sed -e "s/-/\t/g"|sed
"s://"chr"/g"| awk '{print $1, $2}'>${filter}/${i}/lin.map
}

# for i in $(seq 0 0.1 1);do task "$i" &done
i=0.7
task "$i"

# Ne = 35
# vcf2genepop https://github.com/z0on/2bRAD_denovo/blob/master/vcf2genepop.pl

gunzip ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf.gz
/home/lin/GS2018_SNPcalling/Haplo_20210102/vcf2genepop.pl
vcf=${filter}/${i}/Hetero_realigned_cov10_filtered.vcf> ${filter}/filtered_Ne.gen

```

```

bgzip ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf

awk '/chr/ { print; print "pop"; next }1' ${filter}/filtered_Ne.gen >> ${filter}/filtered_Ne_pop.gen


# imputaion
Im_soft=/home/lin/GS2018_SNPcalling/VCF_imputation/20200114_linkimpute
mkdir ${DIR}/VCF_imputation_210922
imputation=${DIR}/VCF_imputation_210922
taskIM(){
mkdir ${imputation}/${i}
java -jar ${Im_soft}/LinkImpute.jar ${filter}/${i}/Hetero_realigned_cov10_filtered.raw
${imputation}/${i}/Imputed.raw;
cat ${imputation}/${i}/Imputed.raw | cut -d " " -f7-> ${imputation}/${i}/Hetero_realigned_cov10_filtered2.raw;
cp ${filter}/${i}/IDs.txt ${imputation}/${i}/IDs.txt
cp ${filter}/${i}/lin.map ${imputation}/${i}/lin.map
paste -d" " ${imputation}/${i}/IDs.txt ${imputation}/${i}/Hetero_realigned_cov10_filtered2.raw >
${imputation}/${i}/Hetero_realigned_cov10_filtered3.raw;
}

taskIM "$i"

#!/bin/bash

echo "START -----"

#module add java/12.0.2
#module load parallel
#module load gatk
statsdir=/home/lin/GS2018_SNPcalling/Readstat

### Use gnu-parallel to use multiple cores
### within one script
cat /home/lin/GS2018_SNPcalling/CODES/sample4.list | parallel --verbose -j 6 "sudo chmod +xr
/home/lin/GS2018_SNPcalling/TRIM/{}{}_paired_R1.fq.gz|
sudo chmod +xr /home/lin/GS2018_SNPcalling/TRIM/{}{}_paired_R2.fq.gz|
/home/lin/seqstats/seqstats /home/lin/GS2018_SNPcalling/TRIM/{}{}_paired_R1.fq.gz |sed -n 1p|cut -f 2

```

```

>>${statsdir}/trined_paired_num_read_1.txt|
/home/lin/seqstats/seqstats /home/lin/GS2018_SNPcalling/TRIM/{}_{}/paired_R2.fq.gz |sed -n 1p|cut -f 2
>>${statsdir}/trined_paired_num_read_2.txt|
/home/lin/seqstats/seqstats /home/lin/GS2018_SNPcalling/TRIM/{}_{}/paired_R1.fq.gz |sed -n 2p|cut -f 2|cut -d' -f1 >>${statsdir}/trined_paired_bp_1.txt|
/home/lin/seqstats/seqstats /home/lin/GS2018_SNPcalling/TRIM/{}_{}/paired_R2.fq.gz |sed -n 2p|cut -f 2|cut -d' -f1 >>${statsdir}/trined_paired_bp_2.txt"

echo "FINISH -----"

echo "START -----"

#module add java/12.0.2
#module load parallel
#module load gatk
statsdir=/home/lin/GS2018_SNPcalling/Readstat
### Use gnu-parallel to use multiple cores
### within one script
# /home/lin/seqstats/seqstats /mnt/c/gs2017ngsdata/lin-60808054/FASTQ_Generation_2018-01-23_06_29_51Z-74544520/${i}"_"${i}"_"${i}"_L001_R1_001.fastq.gz|sed -n 1p|cut -f 2 >>${statsdir}/raw_num_read_1.txt
#/home/lin/seqstats/seqstats /mnt/c/gs2017ngsdata/lin-60808054/FASTQ_Generation_2018-01-23_06_29_51Z-74544520/${i}"_"${i}"_"${i}"_L001_R2_001.fastq.gz|sed -n 1p|cut -f 2 >>${statsdir}/raw_num_read_2.txt
# /home/lin/seqstats/seqstats /mnt/c/gs2017ngsdata/lin-60808054/FASTQ_Generation_2018-01-23_06_29_51Z-74544520/${i}"_"${i}"_"${i}"_L001_R1_001.fastq.gz|sed -n 2p|cut -f 2|cut -d" " -f1 >>${statsdir}/raw_bp_1.txt
#/home/lin/seqstats/seqstats /mnt/c/gs2017ngsdata/lin-60808054/FASTQ_Generation_2018-01-23_06_29_51Z-74544520/${i}"_"${i}"_"${i}"_L001_R2_001.fastq.gz|sed -n 2p|cut -f 2|cut -d" " -f1 >>${statsdir}/raw_bp_2.txt
sudo chmod +xr ${fastqfile}/*

fastqfile=/media/lin/Elements/GS2018_rawread/201912_60_01_1
ls ${fastqfile} | cat |grep _1.fq.gz | parallel --verbose -j 6 "/home/lin/seqstats/seqstats ${fastqfile}{} |sed -n 1p|cut -f 2 >> ${statsdir}/raw_num_read_1a.txt"
/home/lin/seqstats/seqstats ${fastqfile}{} |sed -n 2p|cut -f 2|cut -d' -f1 >> ${statsdir}/raw_bp_1a.txt"
ls ${fastqfile} | cat |grep _2.fq.gz | parallel --verbose -j 6 "/home/lin/seqstats/seqstats ${fastqfile}{} |sed -n 1p|cut -f 2 >> ${statsdir}/raw_num_read_2a.txt"
/home/lin/seqstats/seqstats ${fastqfile}{} |sed -n 2p|cut -f 2|cut -d' -f1 >> ${statsdir}/raw_bp_2a.txt"

fastqfile2=/media/lin/Elements/GS2018_rawread/201912_61_01_1

```

```

ls ${fastqfile2} | cat |grep _1.fq.gz | parallel --verbose -j 6 "/home/lin/seqstats/seqstats ${fastqfile2}/{}/|sed -n
1p|cut -f 2 >> ${statsdir}/raw_num_read_1b.txt"
/home/lin/seqstats/seqstats ${fastqfile2}/{}/|sed -n 2p|cut -f 2|cut -d' ' -f1 >> ${statsdir}/raw_bp_1b.txt"
ls ${fastqfile2} | cat |grep _2.fq.gz | parallel --verbose -j 6 "/home/lin/seqstats/seqstats ${fastqfile2}/{}/|sed -n
1p|cut -f 2 >> ${statsdir}/raw_num_read_2b.txt"
/home/lin/seqstats/seqstats ${fastqfile2}/{}/|sed -n 2p|cut -f 2|cut -d' ' -f1 >> ${statsdir}/raw_bp_2b.txt"

fastqfile3=/media/lin/Elements/GS2018_rawread/201911_11_01_1
ls ${fastqfile3} | cat |grep _1.fq.gz | parallel --verbose -j 6 "/home/lin/seqstats/seqstats ${fastqfile3}/{}/|sed -n
1p|cut -f 2 >> ${statsdir}/raw_num_read_1c.txt"
/home/lin/seqstats/seqstats ${fastqfile3}/{}/|sed -n 2p|cut -f 2|cut -d' ' -f1 >> ${statsdir}/raw_bp_1c.txt"
ls ${fastqfile3} | cat |grep _2.fq.gz | parallel --verbose -j 6 "/home/lin/seqstats/seqstats ${fastqfile3}/{}/|sed -n
1p|cut -f 2 >> ${statsdir}/raw_num_read_2c.txt"
/home/lin/seqstats/seqstats ${fastqfile3}/{}/|sed -n 2p|cut -f 2|cut -d' ' -f1 >> ${statsdir}/raw_bp_2c.txt"

ls ${statsdir}| cat | grep raw_num_read_1 | parallel --verbose "cat ${statsdir}/{}/>>
${statsdir}/raw_num_read_1.txt"
ls ${statsdir}| cat | grep raw_num_read_2 | parallel --verbose "cat ${statsdir}/{}/>>
${statsdir}/raw_num_read_2.txt"
ls ${statsdir}| cat | grep raw_bp_1 | parallel --verbose "cat ${statsdir}/{}/>> ${statsdir}/raw_bp_1.txt"
ls ${statsdir}| cat | grep raw_bp_2 | parallel --verbose "cat ${statsdir}/{}/>> ${statsdir}/raw_bp_2.txt"

```

2.1.3 R script for genotyping

```
rb1<-read.table("raw_bp_1.txt")
rb2<-read.table("raw_bp_2.txt")
rn1<-read.table("raw_num_read_1.txt")
rn2<-read.table("raw_num_read_1.txt")
tb1<-read.table("trined_paired_bp_1.txt")
tb2<-read.table("trined_paired_bp_2.txt")
tn1<-read.table("trined_paired_num_read_1.txt")
tn2<-read.table("trined_paired_num_read_2.txt")

(sum(rb1)+sum(rb2))
(sum(tb1)+sum(tb2))
(sum(tb1)+sum(tb2))/(sum(rb1)+sum(rb2))

mean(unlist(rn1+rn2))
sd(unlist(rn1+rn2))

mean(unlist(tn1+tn2))
sd(unlist(tn1+tn2))

mean(unlist(tn1+tn2))/mean(unlist(rn1+rn2))

(sum(rn1)+sum(rn2))
(sum(tn1)+sum(tn2))
(sum(tn1)+sum(tn2))/(sum(rn1)+sum(rn2))

(sum(rb1)+sum(rb2))/(sum(rn1)+sum(rn2))
(sum(tb1)+sum(tb2))/(sum(tn1)+sum(tn2))
```

2.1.4 Python script for population structure

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm
import seaborn as sns; sns.set()
import matplotlib as mpl

# Loading data
pheno = pd.read_table("./Lin2018_phenotype.txt")
#I = pd.read_table("./Hetero_realigned_cov10_filtered3.raw",sep="\s+")
I = pd.read_table("./reorderG.txt",sep="\s+")-1
bcw=pheno["count"]

from sklearn.manifold import TSNE
model = TSNE(random_state=0,perplexity=20)
tsne5 = model.fit_transform(I)

lin_wide=1.28
font_size=12
%matplotlib inline
%config InlineBackend.figure_format = 'svg'
plt.style.use('seaborn-white')
plt.style.use('seaborn-paper')
plt.rc('font',size=font_size)
plt.rc('lines',lw=lin_wide)
plt.rcParams['font.family'] = 'serif'
plt.rcParams['font.sans-serif'] = 'Times New Roman'
#plt.rcParams['axes.formatter.use_mathtext'] = True
#plt.rcParams['mathtext.bf'] = 'serif:normal'
#plt.rcParams['text.hinting'] = 'none'
#plt.rcParams['text.usetex'] = True
#plt.rcParams['axes.labelweight'] = 'normal'

plt.rcParams['axes.linewidth'] = lin_wide
plt.rcParams['xtick.major.width'] = lin_wide
plt.rcParams['xtick.minor.width'] = lin_wide
plt.rcParams['xtick.labelsize'] = font_size
```

```

plt.rcParams['ytick.major.width'] = lin_wide
plt.rcParams['ytick.minor.width'] = lin_wide
plt.rcParams['ytick.labelsize'] = font_size
plt.rcParams['axes.spines.bottom'] = True
# plt.style.use('bmh')

plt.scatter(tsne5[:, 0], tsne5[:, 1], edgecolor='k', c=bcw, cmap=cm.Reds, linewidth=lin_wide, s=s_value)
plt.xlabel("t-SNE coordinate 1", size=font_size)
plt.ylabel("t-SNE coordinate 2", size=font_size)
plt.subplots_adjust(bottom=0.1, right=0.8, top=0.9)
cax = plt.axes([0.85, 0.1, 0.075, 0.8])
plt.colorbar(cax=cax)
plt.gcf().set_size_inches(7, 6)
plt.savefig("C2.3_tSNE_HC.pdf", format="pdf")
plt.show()
# plt.savefig("PCA_diets.png", dpi=300)

np.corrcoef(tsne5[:, 0], bcw)
np.corrcoef(tsne5[:, 1], bcw)

```

2.1.5 R script for heritability and genetic correlation

```
reorderG<-read.table("GS2018/reorderG.txt")
x<-as.matrix(reorderG)-1
pheno<-read.table("GS2018/Lin2018_phenotype.txt",header = TRUE)
n<-dim(x)[1]

library("sommer")
packageVersion("sommer")
A <- A.mat(x,n.core=8)
row.names(A)=1:n;colnames(A)=1:n
data <- data.frame(tbcw=count,length=SL,id=factor(1:n))
attach(data)
ans.m <- mmer(cbind(tbcw,length)~1,random=~ vs(id, Gu=A, Gtc=unsm(2)),
               rcov=~ vs(units, Gtc=unsm(2)),
               date.warning=FALSE,
               data=data)
(sp_cor<-cov2cor(ans.m$sigma$`u:id`))
pin(ans.m, h2~ V1/(V1+V4))
pin(ans.m, h2~ V3/(V3+V6))

#SL
# Platform information
library(benchmarkme)
get_platform_info()$OS.type
get_r_version()$version.string
get_cpu()$model_name;get_cpu()$no_of_cores
get_ram()

# Parallel computation
library(doParallel)
library(foreach)
cl<-makeCluster(16)

geno <-read.table("./Hetero_realigned_cov10_filtered3.raw", header=T)
pheno <- read.csv("./2017heteroPheno.csv", header=T)
attach(pheno)
n<-length(pheno$bew)
#cat IDlist|sed -e "s/S/t/g"|sort -n|sed -e "s/t/S/g"|awk "{print $2}">ID_list.txt
```

```

id<-read.table("ID_list.txt")#fitSID
reorderIndex<-c()
for (i in id$V1){
  reorderIndex<-c(reorderIndex,match(i,rownames(geno)))
}
reorderG<-geno[reorderIndex,]
rownames(reorderG)=1:n
x <- as.matrix(reorderG)-1
data <- data.frame(tbcw=sqrt(pheno$bew+1),bew=pheno$bew,length=pheno$length,gid=1:240)

# Parameters for cross validation
repeats <- 10
n.fold <- 10
n.sample <- length(pheno$bew)
CM<-7

# Package
library(rrBLUP)
packageVersion("rrBLUP")

# Marker-based relationship matrix (Endelman et al. 2011)
A <- A.mat(x, n.core=8)
row.names(A)=1:240;colnames(A)=1:240

registerDoParallel(cl)
system.time({
  GBLUP<-foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {
      bcw_test <- data
      bcw_test$length[id == i] <- NA
      res <- kin.blup(bcw_test, K=A, geno="gid", pheno="length")
      cor(data$length[id==i],res$pred[id==i])
    }
  }
})
stopImplicitCluster()

```

```

mean(unlist(GLBLUP))

library("BGLR")
packageVersion("BGLR")

registerDoParallel(cl)
system.time({
  BA <- foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="BGLR") %dopar% {
      bcw_test <- data
      bcw_test$length[id == i] <- NA
      fmBA=BGLR(y=bcw_test$length,ETA=list(list(X=x,model='BayesA')),nIter=10000,burnIn=2000,verbose
      = FALSE)
      cor(data$length[id == i],fmBA$yHat[id == i])
    }
  }
})
stopImplicitCluster()

registerDoParallel(cl)
system.time({
  BB <- foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="BGLR") %dopar% {
      bcw_test <- data
      bcw_test$length[id == i] <- NA
      fmBB=BGLR(y=bcw_test$length,ETA=list(list(X=x,model='BayesB')),nIter=10000,burnIn=2000,verbose
      = FALSE)
      cor(data$length[id == i],fmBB$yHat[id == i])
    }
  }
})
stopImplicitCluster()

```

```

registerDoParallel(cl)
system.time({
BC <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$length[id == i] <- NA
    fmBC=BGLR(y=bew_test$length,ETA=list(list(X=x,model='BayesC')),nIter=10000,burnIn=2000,verbose
= FALSE)
    cor(data$length[id == i],fmBC$yHat[id == i])
  }
}
))
stopImplicitCluster()

registerDoParallel(cl)
system.time({
BL <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$length[id == i] <- NA
    fmBL=BGLR(y=bew_test$length,ETA=list(list(X=x,model='BL')),nIter=10000,burnIn=2000,verbose
= FALSE)
    cor(data$length[id == i],fmBL$yHat[id == i])
  }
}
))
stopImplicitCluster()

registerDoParallel(cl)
system.time({
BRR <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {

```

```

bcw_test <- data
bcw_test$length[id == i] <- NA
fmBRR=BGLR(y=bcw_test$length,ETA=list(list(X=x,model='BRR')),nIter=10000,burnIn=2000,verbose
= FALSE)
  cor(data$length[id == i],fmBRR$yHat[id == i])
}
}
})
stopImplicitCluster()

# Reproducing kernel
p<-ncol(x)
D<-(as.matrix(dist(x,method='euclidean'))^2)/p
h<-0.5;K<-exp(-h*D)
ETA<-list(list(K=K,model='RKHS'))

registerDoParallel(cl)
system.time({
RKHS <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$length[id == i] <- NA
    fmRKHS=BGLR(y=bcw_test$length,ETA=ETA,nIter=10000,burnIn=2000,verbose = FALSE)
    cor(pheno$length[id == i],fmRKHS$yHat[id == i])
  }
}
})
stopImplicitCluster()

#Save result
Acc<-data.frame(unlist(GLUP),unlist(BA),unlist(BB),unlist(BC),unlist(BL),unlist(BRR),unlist(RKHS))
colnames(Acc)<-c("GLUP","BA","BB","BC","BL","BRR","RKHS")

(summary<-data.frame(Acc_mean=sapply(Acc,function(x) round(mean(x),digits = 3)),
Acc_SE=sapply(Acc,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3))))
```

```

h2 = 0.405

(summary<-data.frame(Acc_mean=sapply(Acc,function(x) round(mean(x/sqrt(h2)),digits = 3)),
Acc_SE=sapply(Acc,function(x) round(sd(x/sqrt(h2))/sqrt(repeats*n.fold),digits = 3)))))

#previous

(summary<-data.frame(Acc_mean=sapply(Acc,function(x) round(mean(x),digits = 3)),
Acc_SE=sapply(Acc,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3)))))

write.csv(Acc,"All_models_AccSL.csv")
#Acc<-read.csv("All_models_AccSL.xlsx")

#Load data(from this page and deep learning models)
library("readxl")
data<-read_excel("All_models_Acc.xlsx")
Acc_all<-subset(data,select = - c(...1))

(summary<-data.frame(Acc_mean=sapply(Acc_all,function(x) round(mean(x),digits = 3)),
Acc_SE=sapply(Acc_all,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3)))))

write.xlsx(summary, "1.7 summary.xlsx")

```

2.1.6 R script for GWAS

```
R.Version()$version.string  
library("ggplot2")  
library("ggpubr")  
library("extrafont")  
print(paste("ggplot2 version",packageVersion("ggplot2")))  
print(paste("ggpubr version",packageVersion("ggpubr")))  
print(paste("extrafont version",packageVersion("extrafont")))  
# remotes::install_version("Rttf2pt1", version = "1.3.8")  
# ttf_import()  
# fonts()  
# loadfonts(device = "pdf")  
  
# Loading phenotypes  
pheno <- read.table("./Lin2018_phenotype.txt", header=T)  
colnames(pheno)<-c("ID","SID","length","BW","bcw","PostTank","PreTank","TankName")  
pheno$HD<-100*pheno$bcw/pheno$length^2  
attach(pheno)  
n<-length(pheno$bcw)  
  
# Loading imputed genotypes  
geno <-read.table("./Imputation/Hetero_realigned_cov10_filtered3.raw", header=T) # Imputed  
dim(geno)  
  
# Reordering  
id<-read.table("IDlist")#fitSID  
idreorder<-read.table("reorder.txt")  
reorderIndex<-c()  
for (i in idreorder$V1){  
  reorderIndex<-c(reorderIndex,match(i,id$V1))  
}  
reorderG<-geno[reorderIndex,]  
row.names(reorderG)=1:dim(geno)[1]  
write.table(reorderG,"reorderG.txt")  
x <- as.matrix(reorderG)-1  
  
# GBLUP  
library(rrBLUP)
```

```

print(paste("rrBLUP version",packageVersion("rrBLUP")))

# genomic relationship matrix
A <- A.mat(x)
row.names(A)=1:n;colnames(A)=1:n

# data
data <- data.frame(tbcw=pheno$bew,length=pheno$length,
                     bew_d_length2=pheno$HD,BW=pheno$BW, gid=1:n,x=x)
row.names(A)=1:n;colnames(A)=1:n

lin_map<-read.table("./Imputation/lin.map")
g <- data.frame(rownames(lin_map),lin_map$V1, lin_map$V2, t(x))
rownames(g) <-1:nrow(g)
colnames(g) <-c("marker", "chrom", "pos", rownames(x))
# Bonferroni-corrected significance threshold
(thred <- round(-log10(0.05/nrow(g)),3))

bewframe <-data.frame(1:n, data$bew)
colnames(bewframe)<-c("gid", "bew")
# Performing GWAS
GWAS_bew <-GWAS(bewframe, g,K=A, n.PC = 10)
# SNP markers with lowest P-values
GWAS_bew$chrom[head(order(GWAS_bew$bew,decreasing = TRUE),3)]
# Corresponding position (base pair)
GWAS_bew$pos[head(order(GWAS_bew$bew,decreasing = TRUE),3)]

g[(g$chrom==3)&(g$pos<(5531819+1000))&(g$pos>(5531819-1000)),]

table(unlist(g[(g$chrom==3)&(g$pos==5531819),][c(-1,-2,-3)]))

mean(data$bew)
mean(data[unlist(g[(g$chrom==3)&(g$pos==5531819),][c(-1,-2,-3)])==-1,$bew])
mean(data[unlist(g[(g$chrom==3)&(g$pos==5531819),][c(-1,-2,-3)])==0,$bew])
mean(data[unlist(g[(g$chrom==3)&(g$pos==5531819),][c(-1,-2,-3)])==1,$bew])

fmBC_HC=BGLR(y=data$bew,ETA=list(list(X=x,model='BayesC')),
               nIter=10000,burnIn=2000,verbose = FALSE)

```

```

#      fmBC_HC$ETA[[[]]$b[as.numeric(GWAS_bcw$marker[head(order(GWAS_bcw$tbcw,decreasing
TRUE),3))]

# https://github.com/gglinzjie/Useful-code-for-GS/blob/master/sort%26gene%20effect%20plot.ipynb
max(abs(fmBC_HC$ETA[[[]]$b))

lin_map["id"]=1:dim(lin_map)[1]
bayesC_effect_map<-lin_map[order(lin_map$V2),]
bayesC_effect_map_all<-bayesC_effect_map[order(bayesC_effect_map$V1),]
# sortgeno<-geno[,sortM$id]

lin_map["id"]=1:dim(lin_map)[1]
bayesC_effect_map<-lin_map[order(lin_map$V2),]
bayesC_effect_map_all<-bayesC_effect_map[order(bayesC_effect_map$V1),]
# sortgeno<-geno[,sortM$id]
data_gg_HC<-data.frame(
  abs_effect= as.numeric(abs(fmBC_HC$ETA[[[]]$b)[bayesC_effect_map_all$id]),
  chr= factor(bayesC_effect_map_all$V1,levels=c(as.character(1:22)))))

max(abs(fmBC_HC$ETA[[[]]$b))

# Bayes C plot
line_th_unit<-0.6
# Chrom<-data_gg_HC$chr
# data_gg_HC$chrom<-factor(Chrom,levels=c(as.character(1:22)))
options(repr.plot.width=8, repr.plot.height=4)
# my_y_title <- expression(paste("-", "log"[10], "(", italic("p"), ") for HC"))
pbayesc_hc<-ggplot(data_gg_HC, aes(x=1:dim(x)[2], y=abs_effect, color=chr))+

  ggpubr::theme_pubr(base_family="Times New Roman",base_size = 12,border = TRUE)+

  geom_point()+
  scale_color_manual(values = rep(c("black", "skyblue"), 24 ))+
  labs(x="Physical order of SNPs", y="Effect absolute value for HC")+
  theme(legend.position="none",
        axis.text.y=element_text(size=12),
        axis.text.x=element_text(size=12),
        text=element_text(size=12,
                          family="Times New Roman"))

# geom_hline(yintercept=thred, linetype="dashed", color = "red")+

```

```

# geom_text(aes(0,thred,label = thred, vjust = 1.2),color = "black",family="Times New Roman")
pbayesc_hc$theme$line$size<-line_th_unit
pbayesc_hc$theme$line$size<-line_th_unit
pbayesc_hc$theme$axis.ticks$size<-line_th_unit
pbayesc_hc$theme$axis.line$size<-line_th_unit
pbayesc_hc$theme$panel.grid$colour<-"black"
pbayesc_hc$theme$axis.ticks$colour<-"black"
pbayesc_hc

# top100
top100<-cbind(rep("GS2018",100),GWAS_bcw$chrom[head(order(GWAS_bcw$tbcw,decreasing = TRUE),100)],GWAS_bcw$pos[head(order(GWAS_bcw$tbcw,decreasing = TRUE),100)])
colname<-c("Exp","chr","pos")
write.csv(top100,"GS2018_GAWS_HC100.csv")

# Manhattan plot
line_th_unit<-0.6
Chrom<-GWAS_bcw$chrom
GWAS_bcw$chrom<-factor(Chrom,levels=c(as.character(1:22)))
options(repr.plot.width=8, repr.plot.height=4)
my_y_title <- expression(paste("-", "log"[10], "(", italic("p"), ") for HC"))
p1<-ggplot(GWAS_bcw, aes(x=1:dim(x)[2], y=tbcw, color=chrom))+
  ggpubr::theme_pubr(base_family="Times New Roman",base_size = 12,border = TRUE)+ 
  geom_point()+
  scale_color_manual(values = rep(c("black", "skyblue"), 24 ))+
  labs(x="Physical order of SNPs", y=my_y_title)+ 
  theme(legend.position="none",
        axis.text.y=element_text(size=12),
        axis.text.x=element_text(size=12),
        text=element_text(size=12,
                          family="Times New Roman"))+
  geom_hline(yintercept=thred, linetype="dashed", color = "red")+
  geom_text(aes(0,thred,label = thred, vjust = 1.2),color = "black",family="Times New Roman")
p1$theme$line$size<-line_th_unit
p1$theme$line$size<-line_th_unit
p1$theme$axis.ticks$size<-line_th_unit
p1$theme$axis.line$size<-line_th_unit
p1$theme$panel.grid$colour<-"black"

```

```

p1$theme$axis.ticks$colour<-"black"
p1

lengthframe <-data.frame(1:n, length)
colnames(lengthframe) <-c("gid", "length")
# Performing GWAS
GWAS_length <-GWAS(lengthframe, g,K=A, n.PC = 10)
# SNP markers with lowest P-values
GWAS_length$chrom[head(order(GWAS_length$length,decreasing = TRUE),3)]
# Corresponding position (base pair)
GWAS_length$pos[head(order(GWAS_length$length,decreasing = TRUE),3)]]

fmBC_SL=BGLR(y=data$length,ETA=list(list(X=x,model='BayesC')),
nIter=10000,burnIn=2000,verbose = FALSE)
# fmBC$ETA[[[]]$b[as.numeric(GWAS_length$marker[head(order(GWAS_length$length,decreasing = TRUE),3)])]] = max(abs(fmBC_SL$ETA[[[]]$b)))

max(abs(fmBC_SL$ETA[[[]]$b)))

data_gg_SL<-data.frame(
  abs_effect= as.numeric(abs(fmBC_SL$ETA[[[]]$b)[bayesC_effect_map_all$id])),
  chr= factor(bayesC_effect_map_all$V1,levels=c(as.character(1:22)))))

# Bayes C plot
line_th_unit<-0.6
# Chrom<-data_gg_HC$chr
# data_gg_HC$chrom<-factor(Chrom,levels=c(as.character(1:22)))
options(repr.plot.width=8, repr.plot.height=4)
# my_y_title <- expression(paste("-", "log"[10], "(", italic("p"), ") for HC"))
pbayesc_sl<-ggplot(data_gg_SL, aes(x=1:dim(x)[2], y=abs_effect, color=chr))+
  ggpubr::theme_pubr(base_family="Times New Roman",base_size = 12,border = TRUE)+geom_point()+
  scale_color_manual(values = rep(c("black", "skyblue"), 24 ))+
  labs(x="Physical order of SNPs", y="Effect absolute value for SL")+
  theme(legend.position="none",
        axis.text.y=element_text(size=12),
        axis.text.x=element_text(size=12),
        text=element_text(size=12,

```

```

family="Times New Roman"))

# geom_hline(yintercept=thred, linetype="dashed", color = "red")+
# geom_text(aes(0,thred,label = thred, vjust = 1.2),color = "black",family="Times New Roman")

pbayesc_sl$theme$line$size<-line_th_unit
pbayesc_sl$theme$line$size<-line_th_unit
pbayesc_sl$theme$axis.ticks$size<-line_th_unit
pbayesc_sl$theme$axis.line$size<-line_th_unit
pbayesc_sl$theme$panel.grid$colour<-"black"
pbayesc_sl$theme$axis.ticks$colour<-"black"
pbayesc_sl

options(repr.plot.width=5, repr.plot.height=5)
figure <- ggpubr::ggarrange(pbayesc_hc, pbayesc_sl,
                           labels = c("a", "b"),
                           ncol = 1,
                           nrow = 2,
                           font.label = list(size = 14,
                                             color = "black",
                                             face = "plain",
                                             family = "Times New Roman"))

# figure

ggsave("Fig. Bayes C.pdf",device="pdf",units="in",width =6, height = 6 )
# ggsave("Fig2.eps",device="eps",units="in",width =7.2)
# ggsave("Fig. 1.2.tiff", width=7.2, units="in", res=300)

# top100
top100<-cbind(rep("GS2018",100),GWAS_length$chrom[head(order(GWAS_length$length,decreasing = TRUE),100)],
                GWAS_length$pos[head(order(GWAS_length$length,decreasing = TRUE),100)])
colname<-c("Exp","chr","pos")
write.csv(top100,"GS2018_GAWS_SL100.csv")

Chrom<-GWAS_length$chrom
GWAS_length$chrom<-factor(Chrom,levels=c(as.character(1:22)))
options(repr.plot.width=8, repr.plot.height=4)
my_y_title <- expression(paste("-", "log"[10], "(", italic("p"), ") for SL"))
p2<-ggplot(GWAS_length, aes(x=1:dim(x)[2], y=length, color=chrom)) +

```

```

ggpubr::theme_pubr(base_family="Times New Roman",base_size = 12,border = TRUE)+  

  geom_point() +  

  scale_color_manual(values = rep(c("black", "skyblue"), 24 )) +  

  labs(x="Physical order of SNPs", y=my_y_title) +  

  theme(legend.position="none",  

        axis.text.y=element_text(size=12),  

        axis.text.x=element_text(size=12),  

        text=element_text(size=12,  

                          family="Times New Roman")) +  

  geom_hline(yintercept=thred, linetype="dashed", color = "red") +  

  geom_text(aes(0,thred,label = thred, vjust = 1.2),color = "black",family="Times New Roman")  

p2$theme$line$size<-line_th_unit  

p2$theme$line$size<-line_th_unit  

p2$theme$axis.ticks$size<-line_th_unit  

p2$theme$axis.line$size<-line_th_unit  

p2$theme$panel.grid$colour<-"black"  

p2$theme$axis.ticks$colour<-"black"  

p2  

options(repr.plot.width=5, repr.plot.height=5)  

figure <- ggpubr::ggarrange(p1, p2,  

                           labels = c("a", "b"),  

                           ncol = 1,  

                           nrow = 2,  

                           font.label = list(size = 14,  

                                             color = "black",  

                                             face = "plain",  

                                             family = "Times New Roman"))  

figure  

ggsave("Fig. 2.2.pdf",device="pdf",units="in",width =6, height = 6 )  

# ggsave("Fig2.eps",device="eps",units="in",width =7.2)  

# ggsave("Fig. 1.2.tiff", width=7.2, units="in", res=300)

```

Section 2.2

2.2.1 R script for predictive abilities of genomic prediction

```
# The same codes were used for loading genotypes, phenotypes and genomic relationship matrix in 2.1.1

#BLUP

# Parallel computation

library(doParallel)
library(foreach)

cl<-makeCluster(16)

# Parameters for cross validation

repeats <- 10
n.fold <- 5
n.sample <- length(pheno$bcw)

registerDoParallel(cl)
system.time({
  GBLUP<-foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {
      bcw_test <- data
      bcw_test$tbcw[id == i] <- NA
      res <- kin.blup(bcw_test, K=A, geno="gid", pheno="tbcw",covariate = "length")
      cor(data$tbcw[id==i],res$pred[id==i])
    }
  }
})
stopImplicitCluster()

registerDoParallel(cl)
system.time({
  GBLUP2<-foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {
      bcw_test <- data
      bcw_test$length[id == i] <- NA
      res <- kin.blup(bcw_test, K=A, geno="gid", pheno="length")
    }
  }
})
```

```

cor(data$length[id==i],res$pred[id==i])
}
}
})
stopImplicitCluster()

registerDoParallel(cl)
system.time({
GBLUP3<-foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {
    bcw_test <- data
    bcw_test$tbcw[id == i] <- NA
    res <- kin.blup(bcw_test, K=A, geno="gid", pheno="tbcw")
    cor(data$tbcw[id==i],res$pred[id==i])
  }
}
})
stopImplicitCluster()

library("BGLR")
packageVersion("BGLR")

registerDoParallel(cl)
system.time({
BC1 <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$tbcw[id == i] <- NA
    fmBC=BGLR(y=bcw_test$tbcw,
    ETA=list(
      mrk = list(X=x,model='BayesC'),
      fixed=list(~length,data=bcw_test,model='FIXED')),
      nIter=10000,burnIn=2000,verbose = FALSE)
    #fmBC=BGLR(y=bcw_test$tbcw,ETA=list(list(X=x,model='BayesC')),nIter=10000,burnIn=2000,verbose
}
}
})

```

```

= FALSE)
  cor(data$tbcw[id == i],fmBC$yHat[id == i])
}
}
})
stopImplicitCluster()

registerDoParallel(cl)
system.time({
  BC2 <- foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="BGLR") %dopar% {
      bcw_test <- data
      bcw_test$length[id == i] <- NA
      fmBC=BGLR(y=bcw_test$length,ETA=list(list(X=x,model='BayesC')),nIter=10000,burnIn=2000,verbose
= FALSE)
      cor(data$length[id == i],fmBC$yHat[id == i])
    }
  }
})
stopImplicitCluster()

registerDoParallel(cl)
system.time({
  BC3 <- foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="BGLR") %dopar% {
      bcw_test <- data
      bcw_test$tbcw[id == i] <- NA
      fmBC=BGLR(y=bcw_test$tbcw,ETA=list(list(X=x,model='BayesC')),nIter=10000,burnIn=2000,verbose =
FALSE)
      cor(data$tbcw[id == i],fmBC$yHat[id == i])
    }
  }
})
stopImplicitCluster()

```

```

# Reproducing kernel
p<-ncol(x)
D<-(as.matrix(dist(x,method='euclidean'))^2)/p
h<-0.5;K<-exp(-h*D)
ETA<-list(list(K=K,model='RKHS'))

registerDoParallel(cl)
system.time({
  RKHS1 <- foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="BGLR") %dopar% {
      bcw_test <- data
      bcw_test$tbcw[id == i] <- NA
      fmRKHS=BGLR(y=bcw_test$tbcw,ETA=ETA,nIter=10000,burnIn=2000,verbose = FALSE)
      cor(data$tbcw[id == i],fmRKHS$yHat[id == i])
    }
  }
})
stopImplicitCluster()

registerDoParallel(cl)
system.time({
  RKHS2 <- foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="BGLR") %dopar% {
      bcw_test <- data
      bcw_test$length[id == i] <- NA
      fmRKHS=BGLR(y=bcw_test$length,ETA=ETA,nIter=10000,burnIn=2000,verbose = FALSE)
      cor(data$length[id == i],fmRKHS$yHat[id == i])
    }
  }
})
stopImplicitCluster()

# Reproducing kernel

```

```

p<-ncol(x)
D<-(as.matrix(dist(x,method='euclidean'))^2)/p
h<-0.5;K<-exp(-h*D)

registerDoParallel(cl)
system.time({
  RKHS3 <- foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="BGLR") %dopar% {
      bcw_test <- data
      bcw_test$tbcw[id == i] <- NA
      fmRKHS=BGLR(y=bcw_test$tbcw,
                    ETA=list(mrk=list(K=K,model='RKHS'),
                               fixed=list(~length,data=bcw_test,model='FIXED')),
                    nIter=10000,burnIn=2000,verbose = FALSE)
      cor(data$tbcw[id == i],fmRKHS$yHat[id == i])
    }
  }
})
stopImplicitCluster()

# data
Acc<-data.frame(unlist(GLBLUP),unlist(BC1),unlist(RKHS1),unlist(GLBLUP2),unlist(BC2),unlist(RKHS2))
colnames(Acc)<-c("GLBLUP_HC","BayesC_HC","RKHS_HC", "GLBLUP_SL","BayesC_SL","RKHS_SL")
# Predictive ability
(summary<-data.frame(PA_mean=sapply(Acc,function(x) round(mean(x),digits = 3)),
PA_SE=sapply(Acc,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3)))))

# data
Acc<-data.frame(unlist(GLBLUP3),unlist(BC3))
colnames(Acc)<-c("GLBLUP_HC","BayesC_HC")
# Predictive ability
(summary<-data.frame(PA_mean=sapply(Acc,function(x) round(mean(x),digits = 3)),
PA_SE=sapply(Acc,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3)))))

# data

```

```

Acc<-data.frame(unlist(RKHS3))
colnames(Acc)<-c("RKHS3_HC")
# Predictive ability
(summary<-data.frame(PA_mean=sapply(Acc,function(x) round(mean(x),digits = 3)),
PA_SE=sapply(Acc,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3)))))

# Accuracy
h2_HC = 0.162
Acc1<-data.frame(GBLUP_HC=Acc[,c(1,2,3)])
(summary<-data.frame(Acc_mean=sapply(Acc1,function(x) round(mean(x/sqrt(h2_HC)),digits = 3)),
Acc_SE=sapply(Acc1,function(x) round(sd(x/sqrt(h2_HC))/sqrt(repeats*n.fold),digits = 3)))))

h2_SL = 0.537
Acc2<-data.frame(GBLUP_SL=Acc[,c(4,5,6)])
(summary<-data.frame(Acc_mean=sapply(Acc2,function(x) round(mean(x/sqrt(h2_SL)),digits = 3)),
Acc_SE=sapply(Acc2,function(x) round(sd(x/sqrt(h2_SL))/sqrt(repeats*n.fold),digits = 3))))

```

2.2.2 R script for the effect of the number of SNPs on genomic prediction under GBLUP model

```
# The same codes were used for loading genotypes, phenotypes and genomic relationship matrix in 2.1.1
x1<-x[,sample(dim(x)[2],500)]
A<- A.mat(x1)
row.names(A)=1:n;colnames(A)=1:n
data <- data.frame(tbcw=pheno$bew,length=pheno$length,
                    bcw_d_length2=pheno$HD,BW=pheno$BW, gid=1:n,x=x1)
row.names(A)=1:n;colnames(A)=1:n
registerDoParallel(cl)
system.time({
  GBLUP<-foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {
      bcw_test <- data
      bcw_test$tbcw[id == i] <- NA
      res <- kin.blup(bcw_test, K=A, geno="gid", pheno="tbcw")
      cor(data$tbcw[id==i],res$pred[id==i])
    }
  }
})
stopImplicitCluster()
mean(unlist(GBLUP))

gblup_snpdensity<-function(mkrsiz){
  gblup_list<-c()
  for (m in 1:50){
    set.seed(100+3*m+1)
    x1<-x[,sample(dim(x)[2],mkrsiz)]
    A<- A.mat(x1)
    row.names(A)=1:n;colnames(A)=1:n
    data <- data.frame(tbcw=pheno$bew,length=pheno$length,
                        bcw_d_length2=pheno$HD,BW=pheno$BW, gid=1:n,x=x1)
    row.names(A)=1:n;colnames(A)=1:n
    registerDoParallel(cl)
    system.time({
      GBLUP<-foreach(j=1:repeats,.combine = "rbind") %do% {
        set.seed(100+3*j+1)
```

```

id <- sample(1:n.sample %% n.fold) + 1
foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {
  bcw_test <- data
  bcw_test$tbcw[id == i] <- NA
  res <- kin.blup(bcw_test, K=A, geno="gid", pheno="tbcw")
  cor(data$tbcw[id==i],res$pred[id==i])
}
}
})
stopImplicitCluster()
gblup_list<-c(gblup_list, mean(unlist(GLBLUP)))
}
return(gblup_list)
}

gblup_merge<-list()
markers<-c(12548, 10000, 7500, 5000, 2500, 1000, 500, 100, 50)
for (i in 1:length(markers)){
  gblup_merge[[i]]<-gblup_snpdensity(markers[i])
}

markers<-c(12548, 10000, 7500, 5000, 2500, 1000, 500, 100, 50)
data_box<- data.frame(PA=unlist(gblup_merge)[51:450],SNPs=rep(markers[c(-1)],each=50))
write.csv(data_box,"data_boxHC.csv")

options(repr.plot.width=6, repr.plot.height=3)
thred<-0.258
line_th_unit<-0.6
p<-ggplot(data=data_box, aes(y=PA,x=SNPs))+geom_boxplot(aes(group=SNPs))+
  ggpubr::theme_pubr(base_family="Times New Roman",base_size = 12,border = TRUE)+ 
  labs(x="Number of SNPs", y="Predictive abilities for HC")+
  theme(legend.position="none",
        axis.text.y=element_text(size=12),
        axis.text.x=element_text(size=12),
        text=element_text(size=12,
                          family="Times New Roman"))+
  geom_hline(yintercept=thred, linetype="dashed", color = "red")+
  geom_text(aes(0,0.258,label = 0.258, vjust = 1.2),color = "black",family="Times New Roman")

```

```

p$theme$line$size<-line_th_unit
p$theme$line$size<-line_th_unit
p$theme$axis.ticks$size<-line_th_unit
p$theme$axis.line$size<-line_th_unit
p$theme$panel.grid$colour<-"black"
p$theme$axis.ticks$colour<-"black"
p

```

```
ggsave("GPSNPs2.pdf",device="pdf",units="in",width =14, height = 7 )
```

```

library(dplyr)
df_grp_snps= data_box %>% group_by(SNPs) %>%
  summarise(ave_PA = mean(PA),
            sd_PA= sd(PA),
            se = sd(PA)/sqrt(50))
View(round(df_grp_snps,3)[,c(1,2,3,4)])

```

```

gblup_snpdensity<-function(mkrsize){
  gblup_list<-c()
  for (m in 1:50){
    set.seed(100+3*m+1)
    x1<-x[,sample(dim(x)[2],mkrsize)]
    A<- A.mat(x1)
    row.names(A)=1:n;colnames(A)=1:n
    data <- data.frame(tbcw=pheno$bew,length=pheno$length,
                        bcw_d_length2=pheno$HD,BW=pheno$BW, gid=1:n,x=x1)
    registerDoParallel(cl)
    system.time({
      GBLUP<-foreach(j=1:repeats,.combine = "rbind") %do% {
        set.seed(100+3*j+1)
        id <- sample(1:n.sample %% n.fold) + 1
        foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {
          bcw_test <- data
          bcw_test$length[id == i] <- NA
          res <- kin.blup(bcw_test, K=A, geno="gid", pheno="length")
          cor(data$length[id==i],res$pred[id==i])
        }
      }
    })
  }
}

```

```

        }
    }
}

stopImplicitCluster()
gblup_list<-c(gblup_list, mean(unlist(GLBLUP)))
}

return(gblup_list)
}

gblup_mergeSL<-list()
markers<-c(10000, 7500, 5000, 2500, 1000, 500, 100, 50)
for (i in 1:length(markers)){
  gblup_mergeSL[[i]]<-gblup_snpdensity(markers[i])
}

markers<-c(10000, 7500, 5000, 2500, 1000, 500, 100, 50)
data_boxSL<- data.frame(PA=unlist(gblup_mergeSL),SNPs=rep(markers,each=50))
write.csv(data_boxSL,"data_boxSL.csv")

library(dplyr)
df_grp_snps= data_boxSL %>% group_by(SNPs) %>%
  summarise(ave_PA = mean(PA),
            sd_PA= sd(PA),
            se = sd(PA)/sqrt(50))
View(round(df_grp_snps,3)[,c(1,2,3,4)])

View(round(df_grp_snps,3)[,c(2,3,4)])

options(repr.plot.width=6, repr.plot.height=3)
thred<-0.520
line_th_unit<-0.6
p1<-ggplot(data=data_boxSL, aes(y=PA,x=SNPs))+geom_boxplot(aes(group=SNPs))+
  ggpubr::theme_pubr(base_family="Times New Roman",base_size = 12,border = TRUE)+
  labs(x="Number of SNPs", y="Predictive abilities for SL")+
  theme(legend.position="none",
        axis.text.y=element_text(size=12),
        axis.text.x=element_text(size=12),

```

```

text=element_text(size=12,
family="Times New Roman"))+
geom_hline(yintercept=thred, linetype="dashed", color = "red")+
geom_text(aes(0,0.520,label = 0.520, vjust = 1.2),color = "black",family="Times New Roman")
p1$theme$line$size<-line_th_unit
p1$theme$line$size<-line_th_unit
p1$theme$axis.ticks$size<-line_th_unit
p1$theme$axis.line$size<-line_th_unit
p1$theme$panel.grid$colour<-"black"
p1$theme$axis.ticks$colour<-"black"
p1
options(repr.plot.width=6, repr.plot.height=6)
figure <- ggpubr::ggarrange(p, p1,
labels = c("a", "b"),
ncol = 1,
nrow = 2,
font.label = list(size = 14,
color = "black",
face = "plain",
family = "Times New Roman"))
figure
ggsave("Fig. 2merge_snp.pdf",device="pdf",units="in",width =6, height =8 )

```

Section 3.1

3.1.1 R script for feeding trial and statistical tests

```
library(tidyverse)

pheno_0515blood<-read.csv("190515_afterbloodtest_reform.csv")
pheno_0517feeding_trial<-read.csv("190517jst_data_reform.csv")
pheno_0621hetero<-read.csv("190621JST_final_reform.csv")

str(pheno_0515blood)

str(pheno_0517feeding_trial)

str(pheno_0621hetero)

# id0214 dead

trail_SL_BW<-pheno_0517feeding_trial%>%filter(id0214!=2)%>%
dplyr::select(sample_id0515,diet,tank,SL0214,SL0318,SL0416,SL0515,BW0214,BW0318,BW0416,BW0515)
%>%arrange(sample_id0515)

trail_SL_BW$diet<-gsub("1","FM",trail_SL_BW$diet)
trail_SL_BW$diet<-gsub("2","LY",trail_SL_BW$diet)
trail_SL_BW$diet<-gsub("3","PP",trail_SL_BW$diet)
trail_SL_BW$diet<-gsub("4","HY",trail_SL_BW$diet)
trail_SL_BW$diet<-gsub("5","BAC",trail_SL_BW$diet)

trail_SL_BW$diet<-factor(trail_SL_BW$diet, levels=c("FM","PP","BAC","LY","HY"))
trail_SL_BW$tank<-factor(trail_SL_BW$tank, levels=c("1_1","1_2","1_3",
"2_1","2_2","2_3",
"3_1","3_2","3_3",
"4_1","4_2","4_3",
"5_1","5_2","5_3"))

trail_SL_BW["SL3s2d2"]<-((trail_SL_BW$SL0318-trail_SL_BW$SL0214)/trail_SL_BW$SL0214)*100
trail_SL_BW["BW3s2d2"]<-((trail_SL_BW$BW0318-trail_SL_BW$BW0214)/trail_SL_BW$BW0214)*100

trail_SL_BW["SL4s3d3"]<-((trail_SL_BW$SL0416-trail_SL_BW$SL0318)/trail_SL_BW$SL0318)*100
trail_SL_BW["BW4s3d3"]<-((trail_SL_BW$BW0416-trail_SL_BW$BW0318)/trail_SL_BW$BW0318)*100
```

```

trail_SL_BW["SL5s4d4"]<-((trail_SL_BW$SL0515-trail_SL_BW$SL0416)/trail_SL_BW$SL0416)*100
trail_SL_BW["BW5s4d4"]<-((trail_SL_BW$BW0515-trail_SL_BW$BW0416)/trail_SL_BW$BW0416)*100

trail_SL_BW["SL4s2d2"]<-((trail_SL_BW$SL0416-trail_SL_BW$SL0214)/trail_SL_BW$SL0214)*100
trail_SL_BW["BW4s2d2"]<-((trail_SL_BW$BW0416-trail_SL_BW$BW0214)/trail_SL_BW$BW0214)*100

trail_SL_BW["SL5s3d3"]<-((trail_SL_BW$SL0515-trail_SL_BW$SL0318)/trail_SL_BW$SL0318)*100
trail_SL_BW["BW5s3d3"]<-((trail_SL_BW$BW0515-trail_SL_BW$BW0318)/trail_SL_BW$BW0318)*100

trail_SL_BW["SL5s2d2"]<-((trail_SL_BW$SL0515-trail_SL_BW$SL0214)/trail_SL_BW$SL0214)*100
trail_SL_BW["BW5s2d2"]<-((trail_SL_BW$BW0515-trail_SL_BW$BW0214)/trail_SL_BW$BW0214)*100

```

```

#suppli
trail_SL_BW2<-pheno_0517feeding_trial%>%filter(id0214!=2)%>%
  dplyr::select(sample_id0515,diet,tank,SL0214,SL0318,SL0416,SL0515,BW0214,BW0318,BW0416,BW0515)
%>%arrange(sample_id0515)

trail_SL_BW2$diet<-gsub("1","FM",trail_SL_BW2$diet)
trail_SL_BW2$diet<-gsub("2","LY",trail_SL_BW2$diet)
trail_SL_BW2$diet<-gsub("3","PP",trail_SL_BW2$diet)
trail_SL_BW2$diet<-gsub("4","HY",trail_SL_BW2$diet)
trail_SL_BW2$diet<-gsub("5","BAC",trail_SL_BW2$diet)

trail_SL_BW2$diet<-factor(trail_SL_BW2$diet, levels=c("FM","PP","BAC","LY","HY"))
trail_SL_BW2$tank<-factor(trail_SL_BW2$tank, levels=c("1_1","1_2","1_3",
  "2_1","2_2","2_3",
  "3_1","3_2","3_3",
  "4_1","4_2","4_3",
  "5_1","5_2","5_3"))

suppli<-trail_SL_BW2[,c(1,2,3,4,5,6,7,8,9,10,11)]
colnames(suppli)<-c("ID",
  "Diet","Tank","SL1(cm)","SL2(cm)","SL3(cm)","SL4(cm)","BW1(g)","BW2(g)","BW3(g)","BW4(g)")
suppli$Tank <- gsub("1_1", "FM1", suppli$Tank)
suppli$Tank <- gsub("1_2", "FM2", suppli$Tank)
suppli$Tank <- gsub("1_3", "FM3", suppli$Tank)
suppli$Tank <- gsub("2_1", "LY1", suppli$Tank)

```

```

suppli$Tank <- gsub("2_2", "LY2", suppli$Tank)
suppli$Tank <- gsub("2_3", "LY3", suppli$Tank)
suppli$Tank <- gsub("3_1", "PP1", suppli$Tank)
suppli$Tank <- gsub("3_2", "PP2", suppli$Tank)
suppli$Tank <- gsub("3_3", "PP3", suppli$Tank)
suppli$Tank <- gsub("4_1", "HY1", suppli$Tank)
suppli$Tank <- gsub("4_2", "HY2", suppli$Tank)
suppli$Tank <- gsub("4_3", "HY3", suppli$Tank)
suppli$Tank <- gsub("5_1", "BAC1", suppli$Tank)
suppli$Tank <- gsub("5_2", "BAC2", suppli$Tank)
suppli$Tank <- gsub("5_3", "BAC3", suppli$Tank)
suppli$Tank <- gsub("5_3", "BAC3", suppli$Tank)

head(suppli)
str(suppli)
write.csv(suppli,"feeding_suppli.csv")

# summary of phenotypes among all samples
summary(trail_SL_BW,maxsum = 15)
str(trail_SL_BW)

# mean, sd and pvalue of normtest for all samples
pvalue<-function(x){
  m<-round(shapiro.test(x)[2]$p.value,3)
  return(m)
}

p_diet<-function(x){
  res.aov <- aov(x ~ diet,trail_SL_BW)
  p<-unlist(summary(res.aov))[9]
  return(p)
}

trial_SL_BW_basic<-cbind(apply(trail_SL_BW[,c(-1,-2,-3)],2,mean),apply(trail_SL_BW[,c(-1,-2,-3)],2,sd),
  apply(trail_SL_BW[,c(-1,-2,-3)],2,pvalue),apply(trail_SL_BW[,c(-1,-2,-3)],2,p_diet))
colnames(trial_SL_BW_basic)<-c("mean","sd","p_norm","p_ANOVA_diet")
trial_SL_BW_basic

```

```

library("qpcR") #1.4.1
library("emmeans")

gg<-function(p){
  txt_size<-12
  pg<- p+
  #scale_y_discrete(limits = rev(c("FM","PP","BAC","LY","HY")))+ 
  coord_flip()+
  theme_bw()+
  theme(axis.text.y= element_text(size=txt_size, family="Times New Roman"),
        axis.text.x = element_text(size=txt_size, family="Times New Roman"),
        axis.title= element_text(size=txt_size, family="Times New Roman"),
        text=element_text(size=txt_size, family="Times New Roman"))

}

padjustment<-function(p1){
  p1$theme$line$size<-line_th_unit
  p1$theme$line$size<-line_th_unit
  p1$theme$axis.ticks$size<-line_th_unit
  p1$theme$axis.line$size<-line_th_unit
  #p1$theme$panel.grid$colour<-"black"
  p1$theme$axis.ticks$colour<-"black"
  return(p1)
}

# Starting point SL and BW
m1 <- glm(SL0214 ~ 1, family = gaussian, data = trail_SL_BW)
m2 <- glm(SL0214 ~ 1, family = Gamma, data = trail_SL_BW)
m3 <- glm(SL0214 ~ diet, family = gaussian, data = trail_SL_BW)
m4 <- glm(SL0214 ~ diet,family = Gamma, data = trail_SL_BW)
aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4))
df<-c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1])
data.frame(df,round(aiclist,1),round(data.frame(akaike.weights(aiclist)),[1],1),
           round(data.frame(akaike.weights(aiclist))[3],3))

## https://github.com/rvlenth/emmeans/issues/288 pair-wise student t-test
pigs.emm.s <- emmeans(m3, "diet")

```

```

summary(pigs.emm.s)
pairs(pigs.emm.s)
p1<-gg(plot(pigs.emm.s))+
  ylab("Diet")+
  xlab(substitute(paste("EMM of SL"[italic("1")]))))
# ggsave("SL41.png",dpi = 300,units="in",font="Helvetica",width=8, height=4)

p1
p1<-padjustment(p1)

# Starting point SL and BW

m1 <- glm(BW0214 ~ 1, family = gaussian, data = trail_SL_BW)
m2 <- glm(BW0214 ~ 1, family = Gamma, data = trail_SL_BW)
m3 <- glm(BW0214 ~ diet, family = gaussian, data = trail_SL_BW)
m4 <- glm(BW0214 ~ diet,family = Gamma, data = trail_SL_BW)

aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4))
df<-c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1])
data.frame(df,round(aiclist,1),round(data.frame(akaike.weights(aiclist)),[1],1),
           round(data.frame(akaike.weights(aiclist)),[3],3))

## https://github.com/rvlenth/emmeans/issues/288 pair-wise student t-test
pigs.emm.s <- emmeans(m3, "diet")
print("-----")
summary(pigs.emm.s)
pairs(pigs.emm.s)
p2<-gg(plot(pigs.emm.s))+
  ylab("Diet")+
  xlab(substitute(paste("EMM of BW"[italic("1")]))))
p2<-padjustment(p2)
p2

library("lme4")
line_th_unit<-0.3

m1 <- glm(SL5s2d2~ 1, family = gaussian, data = trail_SL_BW)
m2 <- glm(SL5s2d2 ~ 1, family = Gamma, data = trail_SL_BW)

```

```

m3 <- glm(SL5s2d2 ~ diet, family = gaussian, data = trail_SL_BW)
m4 <- glm(SL5s2d2~ diet,family = Gamma, data = trail_SL_BW)
m5 <- glm(SL5s2d2 ~ tank, family = gaussian, data = trail_SL_BW)
m6 <- glm(SL5s2d2~ tank,family = Gamma, data = trail_SL_BW)
m7 <- glm(SL5s2d2 ~ diet+tank, family = gaussian, data = trail_SL_BW)
m8 <- glm(SL5s2d2~ diet+tank,family = Gamma, data = trail_SL_BW)
#m7 <- lmer(SL5s2d2~ diet+(1|tank), data = trail_SL_BW)
#m8 <- glmer(SL5s2d2~ diet+(1|tank),family = Gamma, data = trail_SL_BW,nAGQ=0)

aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4),AIC(m5),AIC(m6),AIC(m7),AIC(m8))
df<-
c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1],extractAIC(m5)[1],extractAIC(
m6)[1],extractAIC(m7)[1],extractAIC(m8)[1])
data.frame(df,round(aiclist,1),round(data.frame(akaike.weights(aiclist))[1],1),
round(data.frame(akaike.weights(aiclist))[,3],3))

pigs.emm.s <- emmeans(m3, "diet")
print("-----")
summary(pigs.emm.s)
pairs(pigs.emm.s)
options(repr.plot.width=8, repr.plot.height=4)

p5<-gg(plot(pigs.emm.s))+  

  ylab("Diet")+
  xlab(substitute(paste("EMM of SL"[italic("41")])))+  

  geom_segment(aes(x = 22, y = 1, xend = 22, yend =5),size=line_th_unit)+  

  geom_segment(aes(x = 21.9,y = 1, xend = 22.1,yend=1),size=line_th_unit)+  

  geom_segment(aes(x = 21.9,y = 5, xend = 22.1, yend =5),size=line_th_unit)+  

  annotate("text",x = 22.3,y=3, label = "italic(p)~`=~0.031", parse =TRUE,size =rel(5), family="Times New  

Roman")  

# ggsave("SL41.png",dpi = 300,units="in",font="Helvetica",width=8, height=4)  

p5<-padjustment(p5)  

p5

m1 <- glm(BW5s2d2~ 1, family = gaussian, data = trail_SL_BW)
m2 <- glm(BW5s2d2 ~ 1, family = Gamma, data = trail_SL_BW)
m3 <- glm(BW5s2d2 ~ diet, family = gaussian, data = trail_SL_BW)

```

```

m4 <- glm(BW5s2d2~ diet,family = Gamma, data = trail_SL_BW)
m5 <- glm(BW5s2d2 ~ tank, family = gaussian, data = trail_SL_BW)
m6 <- glm(BW5s2d2~ tank,family = Gamma, data = trail_SL_BW)
m7 <- glm(BW5s2d2 ~ diet+tank, family = gaussian, data = trail_SL_BW)
m8 <- glm(BW5s2d2~ diet+tank,family = Gamma, data = trail_SL_BW)
#m7 <- lmer(BW5s2d2~ diet+(1|tank), data = trail_SL_BW)
#m8 <- glmer(BW5s2d2~ diet+(1|tank),family = Gamma, data = trail_SL_BW,nAGQ=0)

aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4),AIC(m5),AIC(m6),AIC(m7),AIC(m8))
df<-
c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1],extractAIC(m5)[1],extractAIC(
m6)[1],extractAIC(m7)[1],extractAIC(m8)[1])
data.frame(df,round(aiclist,1),round(data.frame(akaike.weights(aiclist)),[1],1),
round(data.frame(akaike.weights(aiclist))[,3],3))

pigs.emm.s <- emmeans(m4, "diet")
print("-----")
summary(regrid(pigs.emm.s))
contrast(regrid(pigs.emm.s),method = "pairwise")
options(repr.plot.width=8, repr.plot.height=4)
p6<-gg(plot(regrid(pigs.emm.s))+
  ylab("Diet")+
  xlab(substitute(paste("EMM of BW"[italic("41")])))+
  geom_segment(aes(x = 80, y = 1, xend = 80, yend =5),size=line_th_unit)+
  geom_segment(aes(x = 79.5,y = 1, xend = 80.5,yend=1),size=line_th_unit)+
  geom_segment(aes(x = 79.5,y = 5, xend = 80.5, yend =5),size=line_th_unit)+
  annotate("text",x = 81,y=3, label = "italic(p)~`=~0.008", parse =TRUE,size = rel(5), family="Times New
Roman")
p6
# ggsave("BW41.png",dpi = 300,units="in",font="Helvetica",width=8, height=4)

figure <- ggpubr::ggarrange(p1, p2,p5,p6,
  labels = c("a", "b","c","d"),
  ncol = 2, nrow = 2,font.label = list(size = 14,
  color = "black", face = "plain", family = "Times New Roman"))
options(repr.plot.width=8, repr.plot.height=8)
figure

```

```

ggsave("feeding_emmean.pdf",font="Times New Roman",width=7.2, height=7.2)

m1 <- glm(SL0515 ~ diet, family = gaussian, data = trail_SL_BW)
m2 <- glm(SL0515 ~ diet+tank, family = gaussian, data = trail_SL_BW)
m3 <- glm(SL0515 ~ diet, family = Gamma, data = trail_SL_BW)
m4 <- glm(SL0515 ~ diet+tank, family = Gamma, data = trail_SL_BW)
m5 <- glm(SL0515 ~ diet+SL0214, family = gaussian, data = trail_SL_BW)
m6 <- glm(SL0515 ~ diet+SL0214, family = Gamma, data = trail_SL_BW)
aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4),AIC(m5),AIC(m6))
df<-
c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1],extractAIC(m4)[1],extractAIC(
m4)[1])
data.frame(df,round(aiclist,1),round(data.frame(akaike.weights(aiclist)),[1],1),
round(data.frame(akaike.weights(aiclist)),[3],3))

## https://github.com/rvlenth/emmeans/issues/288 pair-wise student t-test
pigs.emm.s <- emmeans(m5, "diet")
print("-----")
summary(pigs.emm.s)
pairs(pigs.emm.s)
p3<-gg(plot(pigs.emm.s))+
ylab("Diet")+
xlab(substitute(paste("EMM of SL"[italic("t4")])))
#p3

m1 <- glm(BW0515 ~ diet, family = gaussian, data = trail_SL_BW)
m2 <- glm(BW0515 ~ diet+tank, family = gaussian, data = trail_SL_BW)
m3 <- glm(BW0515 ~ diet, family = Gamma, data = trail_SL_BW)
m4 <- glm(BW0515 ~ diet+tank, family = Gamma, data = trail_SL_BW)
m5 <- glm(BW0515 ~ diet+SL0214, family = gaussian, data = trail_SL_BW)
m6 <- glm(BW0515 ~ diet+SL0214, family = Gamma, data = trail_SL_BW)
aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4),AIC(m5),AIC(m6))
df<-
c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1],extractAIC(m4)[1],extractAIC(
m4)[1])
data.frame(df,round(aiclist,1),round(data.frame(akaike.weights(aiclist)),[1],1),
round(data.frame(akaike.weights(aiclist)),[3],3))

```

```

pigs.emm.s <- emmeans(m6, "diet")
summary(regrid(pigs.emm.s))
contrast(regrid(pigs.emm.s))
options(repr.plot.width=8, repr.plot.height=4)
p4<-gg(plot(regrid(pigs.emm.s)))+
  ylab("Diet")+
  xlab(substitute(paste("EMM of BW"[italic("t4")])))
#p4
#ggsave("BW41.png",dpi = 300,units="in",font="Helvetica",width=8, height=4)

m1 <- glm(SL5s2d2 ~ diet, family = gaussian, data = trail_SL_BW)
m2 <- glm(SL5s2d2 ~ diet+tank, family = gaussian, data = trail_SL_BW)
m3 <- glm(SL5s2d2 ~ diet, family = Gamma, data = trail_SL_BW)
m4 <- glm(SL5s2d2 ~ diet+tank, family = Gamma, data = trail_SL_BW)
m5 <- glm(SL5s2d2 ~ diet+SL0214, family = gaussian, data = trail_SL_BW)
m6 <- glm(SL5s2d2 ~ diet+SL0214, family = Gamma, data = trail_SL_BW)
aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4),AIC(m5),AIC(m6))
df<-
c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1],extractAIC(m4)[1],extractAIC(
m4)[1])
data.frame(df,round(aiclist,1),round(data.frame(akaike.weights(aiclist))[,1],1),
  round(data.frame(akaike.weights(aiclist))[,3],3))

```

```

pigs.emm.s <- emmeans(m5, "diet")
print("-----")
summary(pigs.emm.s)
pairs(pigs.emm.s)
options(repr.plot.width=8, repr.plot.height=4)
p5<-gg(plot(pigs.emm.s))+
  ylab("Diet")+
  xlab(substitute(paste("EMM of SL"[italic("41")])))
# p5
#ggsave("SL41.png",dpi = 300,units="in",font="Helvetica",width=8, height=4)

```

```

m1 <- glm(BW5s2d2 ~ diet, family = gaussian, data = trail_SL_BW)
m2 <- glm(BW5s2d2 ~ diet+tank, family = gaussian, data = trail_SL_BW)
m3 <- glm(BW5s2d2 ~ diet, family = Gamma, data = trail_SL_BW)

```

```

m4 <- glm(BW5s2d2 ~ diet+tank, family = Gamma, data = trail_SL_BW)
m5 <- glm(BW5s2d2 ~ diet+BW0214, family = gaussian, data = trail_SL_BW)
m6 <- glm(BW5s2d2 ~ diet+BW0214, family = Gamma, data = trail_SL_BW)
aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4),AIC(m5),AIC(m6))
df<-
c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1],extractAIC(m4)[1],extractAIC(m4)[1])
data.frame(df,round(aiclist,1),round(data.frame(akaike.weights(aiclist))[,1],1),
           round(data.frame(akaike.weights(aiclist))[,3],3))

pigs.emm.s <- emmeans(m6, "diet")
print("-----")
summary(regrid(pigs.emm.s))
contrast(regrid(pigs.emm.s),method = "pairwise")
options(repr.plot.width=8, repr.plot.height=4)
p6<-gg(plot(regrid(pigs.emm.s)))+
  ylab("Diet")+
  xlab(substitute(paste("EMM of BW"[italic("41")])))
# p6
# ggsave("BW41.png",dpi = 300,units="in",font="Helvetica",width=8, height=4)

figure <- ggpubr::ggarrange(p1, p2,p3,p4,p5,p6,
                           labels = c("a", "b","c","d","e","f"),
                           ncol = 2, nrow = 3)
options(repr.plot.width = 12, repr.plot.height = 10)
figure

ggsave("feeding_emmean.png",dpi = 300,units="in",font="Helvetica",width=12, height=10)

m2 <- glm(SL4s3d3 ~ diet+SL0214, family = gaussian, data = trail_SL_BW)
## https://github.com/rvlenth/emmeans/issues/288 pair-wise student t-test
pigs.emm.s <- emmeans(m2, "diet")

```

```

print("-----")
summary(pigs.emm.s)
pairs(pigs.emm.s)

m1 <- glm(SL5s2d2 ~ diet, family = gaussian, data = trail_SL_BW)
m2 <- glm(SL5s2d2 ~ diet+SL0214, family = gaussian, data = trail_SL_BW)
m3 <- glm(SL5s2d2 ~ diet, family = Gamma, data = trail_SL_BW)
m4 <- glm(SL5s2d2 ~ diet+SL0214, family = Gamma, data = trail_SL_BW)
aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4))
df<-c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1])
data.frame(df,round(aiclist,3),round(data.frame(akaike.weights(aiclist)),3))

m1 <- glm(BW5s3d3 ~ diet, family = gaussian, data = trail_SL_BW)
m2 <- glm(BW5s3d3 ~ diet+SL0214, family = gaussian, data = trail_SL_BW)
m3 <- glm(BW5s3d3 ~ diet, family = Gamma, data = trail_SL_BW)
m4 <- glm(BW5s3d3 ~ diet+SL0214, family = Gamma, data = trail_SL_BW)
aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4))
df<-c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1])
data.frame(df,round(aiclist,3),round(data.frame(akaike.weights(aiclist)),3))

m1 <- glm(BW5s2d2 ~ diet, family = gaussian, data = trail_SL_BW)
m2 <- glm(BW5s2d2 ~ diet+SL0214, family = gaussian, data = trail_SL_BW)
m3 <- glm(BW5s2d2 ~ diet, family = Gamma, data = trail_SL_BW)
m4 <- glm(BW5s2d2 ~ diet+SL0214, family = Gamma, data = trail_SL_BW)
aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4))
df<-c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1])
data.frame(df,round(aiclist,3),round(data.frame(akaike.weights(aiclist)),3))

m3 <- glm(SL5s2d2 ~ diet+tank, family = gaussian, data = trail_SL_BW)
AIC(m1);AIC(m2);AIC(m3)

m2 <- glm(SL5s2d2 ~ diet+SL0214, family = gaussian, data = trail_SL_BW)
pigs.emm.s <- emmeans(m2, "diet")
pairs(pigs.emm.s)

m2 <- glm(SL5s3d3 ~ diet+SL0214, family = gaussian, data = trail_SL_BW)
pigs.emm.s <- emmeans(m2, "diet")
pairs(pigs.emm.s)

```

```
m2 <- glm(SL5s4d4 ~ diet+SL0214, family = gaussian, data = trail_SL_BW)
pigs.emm.s <- emmeans(m2, "diet")
pairs(pigs.emm.s)
```

```
m2 <- glm(SL4s2d2 ~ diet+SL0214, family = gaussian, data = trail_SL_BW)
pigs.emm.s <- emmeans(m2, "diet")
pairs(pigs.emm.s)
```

```
m2 <- glm(BW5s2d2 ~ diet+BW0214, family = gaussian, data = trail_SL_BW)
pigs.emm.s <- emmeans(m2, "diet")
pairs(pigs.emm.s)
```

```
m2 <- glm(BW5s3d3 ~ diet+BW0214, family = gaussian, data = trail_SL_BW)
pigs.emm.s <- emmeans(m2, "diet")
pairs(pigs.emm.s)
```

```
m2 <- glm(BW5s4d4 ~ diet+BW0214, family = gaussian, data = trail_SL_BW)
pigs.emm.s <- emmeans(m2, "diet")
pairs(pigs.emm.s)
```

```
m2 <- glm(BW4s3d3 ~ diet+BW0214, family = gaussian, data = trail_SL_BW)
pigs.emm.s <- emmeans(m2, "diet")
pairs(pigs.emm.s)
```

```
m2 <- glm(BW4s2d2 ~ diet+BW0214, family = gaussian, data = trail_SL_BW)
pigs.emm.s <- emmeans(m2, "diet")
pairs(pigs.emm.s)
```

```
m2 <- glm(BW3s2d2 ~ diet+BW0214, family = gaussian, data = trail_SL_BW)
pigs.emm.s <- emmeans(m2, "diet")
pairs(pigs.emm.s)
```

```
library("emmeans")
```

```
pigs.emm.s <- emmeans(m1, "diet")
pairs(pigs.emm.s)
```

```

options(repr.plot.width=8, repr.plot.height=4)
plot(pigs.emm.s, comparisons = TRUE)

options(repr.plot.width=8, repr.plot.height=4)
pwpp(pigs.emm.s)

# mean, sd and pvalue of normtest for each diet
SL_mean_sd_test<-group_by(trail_SL_BW,diet)%>%summarise_at(vars("SL5s3d3","BW5s3d3"),
list(~mean(.),~sd(.),~pvalue(.)))

SL_mean_sd_test

# mean, sd and pvalue of normtest for each diet
SL_mean_sd_test<-
group_by(trail_SL_BW,diet)%>%summarise_at(vars("SL0214","SL0318","SL0416","SL0515",
"BW0214","BW0318","BW0416","BW0515"),
list(~round(mean(.),2)))

t(SL_mean_sd_test)

# mean, sd and pvalue of normtest for each diet
SL_mean_sd_test<-
group_by(trail_SL_BW,diet)%>%summarise_at(vars("SL0214","SL0318","SL0416","SL0515",
"BW0214","BW0318","BW0416","BW0515"),
list(~round(sd(.),2)))

t(SL_mean_sd_test)

# mean, sd and pvalue of normtest for each diet
SL_mean_sd_test<-
group_by(trail_SL_BW,diet)%>%summarise_at(vars("SL0214","SL0318","SL0416","SL0515",
"BW0214","BW0318","BW0416","BW0515"),
list(~round(pvalue(.),3)))

t(SL_mean_sd_test)

# mean, sd and pvalue of normtest for each diet
SL_mean_sd_test<-group_by(trail_SL_BW,diet)%>%summarise_at(vars("SL3s2d2","SL4s3d3","SL5s4d4",
"SL4s2d2","SL5s3d3","SL5s2d2",
"BW3s2d2","BW4s3d3","BW5s4d4",
"BW4s2d2","BW5s3d3","BW5s2d2"),
list(~round(mean(.),2)))

```

```

t(SL_mean_sd_test)

# mean, sd and pvalue of normtest for each diet
SL_mean_sd_test<-group_by(trail_SL_BW,diet)%>%summarise_at(vars("SL3s2d2","SL4s3d3","SL5s4d4",
"SL4s2d2","SL5s3d3","SL5s2d2",
"BW3s2d2","BW4s3d3","BW5s4d4",
"BW4s2d2","BW5s3d3","BW5s2d2"),
list(~round(sd(.),2)))

t(SL_mean_sd_test)

# mean, sd and pvalue of normtest for each diet
SL_mean_sd_test<-group_by(trail_SL_BW,diet)%>%summarise_at(vars("SL3s2d2","SL4s3d3","SL5s4d4",
"SL4s2d2","SL5s3d3","SL5s2d2",
"BW3s2d2","BW4s3d3","BW5s4d4",
"BW4s2d2","BW5s3d3","BW5s2d2"),
list(~round(pvalue(.),2)))

t(SL_mean_sd_test)

var_t_test<-function(CG, TG){
  f<-round(unlist(var.test(CG, TG)[3]),3)
  w<-round(wilcox.test(CG,TG)$p.value,3)
  if(f>0.05){
    t<-round(unlist(t.test(CG, TG, var.equal = TRUE)[3]),3)
    return(c(f,t,w))
  }
  else{
    t<-round(unlist(t.test(CG, TG, var.equal = FALSE)[3]),3)
    return(cbind(f,t,w))
  }
}

result_SL_g<-c()
CG <- trail_SL_BW[trail_SL_BW$diet==1,]$SL5s2d2
for (i in c(2,3,4,5)){
  TG <- trail_SL_BW[trail_SL_BW$diet==i,]$SL5s2d2
  result_SL_g<-rbind(result_SL_g, var_t_test(CG,TG))
}

```

```

result_SL_g

result_BW_g<-c()
CG <- trail_SL_BW[trail_SL_BW$diet==1,]$BW5s2d2
for (i in c(2,3,4,5)){
  TG <- trail_SL_BW[trail_SL_BW$diet==i,]$BW5s2d2
  result_BW_g<-rbind(result_BW_g, var_t_test(CG,TG))
}

result_BW_g

library("reshape2")
library("ggplot2")

boxplot_SL<-melt(trail_SL_BW,id.vars = c("sample_id0515","diet"),
  measure.vars = c("SL0214", "SL0318","SL0416","SL0515"))
boxplot_SL$variable<-gsub("SL0214","1st",boxplot_SL$variable)
boxplot_SL$variable<-gsub("SL0318","2nd",boxplot_SL$variable)
boxplot_SL$variable<-gsub("SL0416","3rd",boxplot_SL$variable)
boxplot_SL$variable<-gsub("SL0515","4th",boxplot_SL$variable)
boxplot_SL$diet<-gsub("1","FM",boxplot_SL$diet)
boxplot_SL$diet<-gsub("2","LY",boxplot_SL$diet)
boxplot_SL$diet<-gsub("3","PP",boxplot_SL$diet)
boxplot_SL$diet<-gsub("4","HY",boxplot_SL$diet)
boxplot_SL$diet<-gsub("5","BAC",boxplot_SL$diet)

boxplot_SL$diet<-factor(boxplot_SL$diet, levels=c("FM","PP","BAC","LY","HY"))

str(boxplot_SL)

library("extrafont")
# https://github.com/wch/extrafont

packageVersion("Rttf2pt1")

packageVersion("Rttf2pt1")
# font_import()

```

```

fonts()

loadfonts(device = "pdf",quiet = TRUE)

line_th_unit<-0.6
txt_size<-12

padjustment<-function(p1){
  p1$theme$line$size<-line_th_unit
  p1$theme$line$size<-line_th_unit
  p1$theme$axis.ticks$size<-line_th_unit
  p1$theme$axis.line$size<-line_th_unit
  #p1$theme$panel.grid$colour<-"black"
  p1$theme$axis.ticks$colour<-"black"
  return(p1)
}

options(repr.plot.width=8, repr.plot.height=4)
p_SL <- ggplot(boxplot_SL, aes(x=variable, y=value,fill=diet)) +
  # geom_boxplot()+
  geom_boxplot()+
  labs(x = "Time of measure",y="SL")+
  # scale_fill_manual(values=c("red", "blue","limegreen","purple","yellow"),name="Diet") +
  scale_fill_brewer(palette="Accent",name="Diet")+
  theme_bw(base_family="Times New Roman",base_line_size = line_th_unit,base_size = txt_size )+
  theme(axis.text.y= element_text(size = txt_size),
        axis.text.x = element_text(size = txt_size),
        text=element_text(size = txt_size,colour = "black"))

p_SL<-padjustment(p_SL)

p_SL

boxplot_BW<-melt(trail_SL_BW,id.vars = c("sample_id0515","diet"),
  measure.vars = c("BW0214", "BW0318","BW0416","BW0515"))

```

```

boxplot_BW$variable<-gsub("BW0214","1st",boxplot_BW$variable)
boxplot_BW$variable<-gsub("BW0318","2nd",boxplot_BW$variable)
boxplot_BW$variable<-gsub("BW0416","3rd",boxplot_BW$variable)
boxplot_BW$variable<-gsub("BW0515","4th",boxplot_BW$variable)
boxplot_BW$diet<-gsub("1","FM",boxplot_BW$diet)
boxplot_BW$diet<-gsub("2","LY",boxplot_BW$diet)
boxplot_BW$diet<-gsub("3","PP",boxplot_BW$diet)
boxplot_BW$diet<-gsub("4","HY",boxplot_BW$diet)
boxplot_BW$diet<-gsub("5","BAC",boxplot_BW$diet)
boxplot_BW$diet<-factor(boxplot_BW$diet, levels=c("FM","PP","BAC","LY","HY"))

options(repr.plot.width=8, repr.plot.height=4)
p_BW <- ggplot(boxplot_BW, aes(x=variable, y=value, fill=diet)) +
  geom_boxplot()+
  #geom_violin(draw_quantiles = c(0.25, 0.5, 0.75))+ 
  labs(x = "Time of measure",y="BW") +
  #scale_fill_manual(values=c("red", "blue","limegreen","purple","yellow"),name="Diet")+
  scale_fill_brewer(palette="Accent",name="Diet")+
  theme_bw(base_family="Times New Roman",base_line_size = line_th_unit,base_size = txt_size )+
  theme(axis.text.y= element_text(size = txt_size),
        axis.text.x = element_text(size = txt_size),
        text=element_text(size = txt_size, colour = "black"))

p_BW<-padjustment(p_BW)

## options(repr.plot.width = 8, repr.plot.height = 8)
# p_all<-plot_grid(p_SL,p_BW ,ncol = 1, align = 'v', axis = 'lr')
# p_all

figure <- ggpubr::ggarrange(p_SL, p_BW,
  labels = c("a", "b"),
  ncol = 1, nrow = 2,
  font.label = list(size = 14,
    color = "black", face = "bold", family = "Times New Roman"))
options(repr.plot.width = 12, repr.plot.height = 12)
figure

```

```
#ggsave("plot_SL_BW.png",dpi = 300)

figure
ggsave("Fig. 1.pdf",device="pdf",units="in",width =7.2)

embed_fonts("Fig. 1.pdf", outfile="Fig. 1_embed.pdf")
```

Section 3.2

3.2.1 R script for blood test and statistical tests

```
library(tidyverse)

pheno_0515blood<-read.csv("190515_afterbloodtest_reform.csv")
pheno_0517feeding_trial<-read.csv("190517jst_data_reform.csv")

str(pheno_0515blood)

trial_SL_BW<-pheno_0517feeding_trial%>%filter(id0214!=2)%>%

dplyr::select(sample_id0515,diet,tank,SL0214,SL0318,SL0416,SL0515,BW0214,BW0318,BW0416,BW0515)
%>%arrange(sample_id0515)

pheno_0515blood$diet<-gsub("1","FM",pheno_0515blood$diet)
pheno_0515blood$diet<-gsub("2","LY",pheno_0515blood$diet)
pheno_0515blood$diet<-gsub("3","PP",pheno_0515blood$diet)
pheno_0515blood$diet<-gsub("4","HY",pheno_0515blood$diet)
pheno_0515blood$diet<-gsub("5","BAC",pheno_0515blood$diet)

pheno_0515blood$diet<-factor(pheno_0515blood$diet, levels=c("FM","PP","BAC","LY","HY"))
pheno_0515blood$tank<-factor(pheno_0515blood$tank, levels=c("1_1","1_2","1_3",
                "2_1","2_2","2_3",
                "3_1","3_2","3_3",
                "4_1","4_2","4_3",
                "5_1","5_2","5_3"))

g_liver<-c()
n=0

g_liver_idlist<-which(pheno_0515blood[c(1:4,6:151)]$remarks=="g_liver")
for (i in 1:150){
  if (i %in% g_liver_idlist){
    g_liver<-c(g_liver,1)
  } else{
    g_liver<-c(g_liver, 0)
  }
}
```

```

blood_test<-pheno_0515blood[c(1:4,6:151),c(1,3,4,8,9,10,13,15,16,17,18)]
blood_test["Gender"]<-factor(blood_test$Gender)
blood_test$diet<-gsub("1","FM",blood_test$diet)
blood_test$diet<-gsub("2","LY",blood_test$diet)
blood_test$diet<-gsub("3","PP",blood_test$diet)
blood_test$diet<-gsub("4","HY",blood_test$diet)
blood_test$diet<-gsub("5","BAC",blood_test$diet)

blood_test$diet<-factor(blood_test$diet, levels=c("FM","PP","BAC","LY","HY"))
pheno_trail_blood<-c()
for (i in blood_test$sample_id){
  pheno_trail_blood<-rbind(pheno_trail_blood,trail_SL_BW[trail_SL_BW$sample_id0515==i,])
}
blood_test["SL5s2d2"]<-(pheno_trail_blood$SL0515-
pheno_trail_blood$SL0214)/pheno_trail_blood$SL0214*100
blood_test["BW5s2d2"]<-(pheno_trail_blood$BW0515-
pheno_trail_blood$BW0214)/pheno_trail_blood$BW0214*100
blood_test["green_liver"]<-factor(g_liver)
blood_test["LSI"]<-blood_test$LSI
summary(blood_test, maxsum=15)
str(blood_test)

```

```

suppli<-cbind(pheno_0515blood[c(1:4,6:151),c(1,4,3,11,12,15,16,17,18,8,9,10)],pheno_trail_blood$id0214)
colnames(suppli)<-c("ID", "Diet","Tank","Ht1", "Ht2", "TCHO","TG","TP","GLU","Gender","LW","LSI","ID
of feeding trail")
suppli$Tank <- gsub("1_1", "FM1", suppli$Tank)
suppli$Tank <- gsub("1_2", "FM2", suppli$Tank)
suppli$Tank <- gsub("1_3", "FM3", suppli$Tank)
suppli$Tank <- gsub("2_1", "LY1", suppli$Tank)
suppli$Tank <- gsub("2_2", "LY2", suppli$Tank)
suppli$Tank <- gsub("2_3", "LY3", suppli$Tank)
suppli$Tank <- gsub("3_1", "PP1", suppli$Tank)
suppli$Tank <- gsub("3_2", "PP2", suppli$Tank)
suppli$Tank <- gsub("3_3", "PP3", suppli$Tank)
suppli$Tank <- gsub("4_1", "HY1", suppli$Tank)
suppli$Tank <- gsub("4_2", "HY2", suppli$Tank)
suppli$Tank <- gsub("4_3", "HY3", suppli$Tank)

```

```

suppli$Tank <- gsub("5_1", "BAC1", suppli$Tank)
suppli$Tank <- gsub("5_2", "BAC2", suppli$Tank)
suppli$Tank <- gsub("5_3", "BAC3", suppli$Tank)
suppli$Tank <- gsub("5_3", "BAC3", suppli$Tank)
str(suppli)

write.csv(suppli,"bloodtest.csv")

# Diet Difference
group_by(blood_test,diet)%>%summarise_at(vars("LW","LSI","Ht_ave","TCHO","TG","TP","GLU","SL5s2d2
","BW5s2d2"),list(~round(mean(.),2)))

# Diet Difference
group_by(blood_test,diet)%>%summarise_at(vars("LW","LSI","Ht_ave","TCHO","TG","TP","GLU","SL5s2d2
","BW5s2d2"),list(~round(sd(.),2)))

head(blood_test)

res<-cor(blood_test[,c(-1,-2,-3,-4,-14)])
library(corrplot)
options(repr.plot.width=8, repr.plot.height=8)

corrplot.mixed(res, upper = "square", tl.col = "black",insig = "p-value")

y<-list(blood_test$Ht_ave,blood_test$TCHO,blood_test$TG, blood_test$TP, blood_test$GLU, blood_test$LW,
blood_test$LSI)
ylab<-c("Ht","TCHO","TG", "TP", "GLU", "LW", "HSI")

library(extrafont)

options(repr.plot.width = 4, repr.plot.height = 4)
line_th_unit<-0.6
txt_size<-12

padjustment<-function(p1){
  p1$theme$line$size<-line_th_unit
  p1$theme$line$size<-line_th_unit
  p1$theme$axis.ticks$size<-line_th_unit
}

```

```

p1$theme$axis.line$size<-line_th_unit
#p1$theme$panel.grid$colour<-"black"
p1$theme$axis.ticks$colour<-"black"
return(p1)
}

boxplotgg<-function(i){
  p1 <- ggplot(blood_test,aes(y=unlist(y[i]),x=diet)) +
    labs(x = "Diet",y=ylab[i]) +
    # scale_fill_manual(values=c"red", "blue","limegreen","purple","yellow"),name="Diet") +
    geom_boxplot()+
    theme_bw(base_family="Times New Roman",base_line_size = line_th_unit,base_size = txt_size )+
    theme(axis.text.y= element_text(size=txt_size),
          axis.text.x = element_text(size=txt_size),
          axis.title= element_text(size=txt_size),
          text=element_text(size=txt_size))
  p1<-padjustment(p1)
  return(p1)
}

```

```

figure <- ggpubr::ggarrange(boxplotgg(1),boxplotgg(2),boxplotgg(3),boxplotgg(4),
                           boxplotgg(5),boxplotgg(6),boxplotgg(7),
                           labels = c("a", "b","c","d","e","f","g"),
                           ncol = 2, nrow = 4, font.label = list(size = 14, face="plain",
                           color = "black",family = "Times New Roman"))
options(repr.plot.width = 12, repr.plot.height = 14)
figure
# ggsave("blood_box.png",dpi = 300,units="in",font="Helvetica",width=12, height=14)

ggsave("Fig. 2.pdf",device="pdf",units="in",width =7.2,height = 9.6)
embed_fonts("Fig. 2.pdf", outfile="Fig. 2_embed.pdf")

```

```

library("emmeans")
library("qpcR") #1.4.1

```

```
gg<-function(p){
```

```

txt_size<-12
line_th_unit=0.6
pg<- p+
#scale_y_discrete(limits = rev(c("FM","PP","BAC","LY","HY")))+ 
coord_flip()+
theme_bw(base_family="Times New Roman",base_line_size = line_th_unit,base_size = txt_size )+
theme(axis.text.y= element_text(size=txt_size, family="Times New Roman"),
      axis.text.x = element_text(size=txt_size, family="Times New Roman"),
      axis.title= element_text(size=txt_size, family="Times New Roman"),
      text=element_text(size=txt_size, family="Times New Roman"))
return(padjustment(pg))
}

```

```
library("lme4")
```

```

m1 <- glm(Ht_ave~ 1, family = gaussian, data = blood_test)
m2 <- glm(Ht_ave ~ 1, family = Gamma, data = blood_test)
m3 <- glm(Ht_ave ~ diet, family = gaussian, data = blood_test)
m4 <- glm(Ht_ave~ diet,family = Gamma, data = blood_test)
m5 <- glm(Ht_ave ~ tank, family = gaussian, data = blood_test)
m6 <- glm(Ht_ave~ tank,family = Gamma, data = blood_test)
m7 <- glm(Ht_ave ~ diet+tank, family = gaussian, data = blood_test)
m8 <- glm(Ht_ave~ diet+tank,family = Gamma, data = blood_test)
#m7 <- lmer(Ht_ave~ diet+(1|tank), data = blood_test)
#m8 <- glmer(Ht_ave~ diet+(1|tank),family = Gamma, data = blood_test,nAGQ=0)

aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4),AIC(m5),AIC(m6),AIC(m7),AIC(m8))
df<-
c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1],extractAIC(m5)[1],extractAIC(
m6)[1],extractAIC(m7)[1],extractAIC(m8)[1])
data.frame(round(aiclist,1),round(data.frame(akaike.weights(aiclist))[,1],1),
           round(data.frame(akaike.weights(aiclist))[,3],3))

## https://github.com/rvlenth/emmeans/issues/288 pair-wise student t-test
pigs.emm.s <- emmeans(m3, "diet")
print("-----")
summary(pigs.emm.s)
pairs(pigs.emm.s)

```

```

p1<-gg(plot(pigs.emm.s))+  

  ylab("Diet")+
  xlab(substitute(paste("EMM of Ht")))+  

  geom_segment(aes(x = 32.5, y = 1, xend = 32.5, yend =5))+  

  geom_segment(aes(x = 33,y = 1, xend = 32,yend=1))+  

  geom_segment(aes(x = 33,y = 5, xend = 32, yend =5))+  

  annotate("text",x = 33,y=3, label = "italic(p)~`=~0.009", parse =TRUE,size = rel(5), family="Times New  

Roman")  

# ggsave("SL41.png",dpi = 300,units="in",font="Helvetica",width=8, height=4)  

p1

m1 <- glm(TCHO~ 1, family = gaussian, data = blood_test)  

m2 <- glm(TCHO ~ 1, family = Gamma, data = blood_test)  

m3 <- glm(TCHO ~ diet, family = gaussian, data = blood_test)  

m4 <- glm(TCHO~ diet,family = Gamma, data = blood_test)  

m5 <- glm(TCHO ~ tank, family = gaussian, data = blood_test)  

m6 <- glm(TCHO~ tank,family = Gamma, data = blood_test)  

m7 <- glm(TCHO ~ diet+tank, family = gaussian, data = blood_test)  

m8 <- glm(TCHO~ diet+tank,family = Gamma, data = blood_test)  

#m7 <- lmer(TCHO~ diet+(1|tank), data = blood_test)  

#m8 <- glmer(TCHO~ diet+(1|tank),family = Gamma, data = blood_test,nAGQ=0)

aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4),AIC(m5),AIC(m6),AIC(m7),AIC(m8))  

df<-
c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1],extractAIC(m5)[1],extractAIC(
m6)[1],extractAIC(m7)[1],extractAIC(m8)[1])
data.frame(round(aiclist,1),round(data.frame(akaike.weights(aiclist))[1],1),
           round(data.frame(akaike.weights(aiclist))[3],3))

## https://github.com/rvlenth/emmeans/issues/288 pair-wise student t-test
pigs.emm.s <- emmeans(m3, "diet")
print("-----")
summary(pigs.emm.s)
pairs(pigs.emm.s)
p2<-gg(plot(pigs.emm.s))+  

  ylab("Diet")+
  xlab(substitute(paste("EMM of TCHO")))+  

  geom_segment(aes(x = 225, y = 1, xend = 225, yend =3))+
```

```

geom_segment(aes(x = 223,y = 1, xend = 227,yend=1))+  

  geom_segment(aes(x = 223,y = 3, xend = 227, yend =3))+  

  annotate("text",x = 230,y=2, label = "italic(p)~`=~0.015", parse =TRUE,size = rel(5),family="Times New  

Roman")+  
  

  geom_segment(aes(x = 240, y = 1, xend = 240, yend =4))+  

  geom_segment(aes(x = 238,y = 1, xend = 242,yend=1))+  

  geom_segment(aes(x = 238,y = 4, xend = 242, yend =4))+  

  annotate("text",x = 245,y=2.5, label = "italic(p)~`=~0.007", parse =TRUE,size = rel(5),family="Times New  

Roman")+  
  

  geom_segment(aes(x = 255, y = 1, xend = 255, yend =5))+  

  geom_segment(aes(x = 253,y = 1, xend = 257,yend=1))+  

  geom_segment(aes(x = 253,y = 5, xend = 257, yend =5))+  

  annotate("text",x = 260,y=3, label = "italic(p)~`=~0.047", parse =TRUE,size = rel(5),family="Times New  

Roman")  
  

# ggsave("SL41.png",dpi = 300,units="in",font="Helvetica",width=8, height=4)  

p2  
  

m1 <- glm(TG~ 1, family = gaussian, data = blood_test)  

m2 <- glm(TG ~ 1, family = Gamma, data = blood_test)  

m3 <- glm(TG ~ diet, family = gaussian, data = blood_test)  

m4 <- glm(TG~ diet,family = Gamma, data = blood_test)  

m5 <- glm(TG ~ tank, family = gaussian, data = blood_test)  

m6 <- glm(TG~ tank,family = Gamma, data = blood_test)  

m7 <- glm(TG ~ diet+tank, family = gaussian, data = blood_test)  

m8 <- glm(TG~ diet+tank,family = Gamma, data = blood_test)  
  

#m7 <- lmer(TG~ diet+(1|tank), data = blood_test)  

#m8 <- glmer(TG~ diet+(1|tank),family = Gamma, data = blood_test,nAGQ=0)  
  

aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4),AIC(m5),AIC(m6),AIC(m7),AIC(m8))  

df<-  

c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1],extractAIC(m5)[1],extractAIC(  

m6)[1],extractAIC(m7)[1],extractAIC(m8)[1])  

data.frame(round(aiclist,1),round(data.frame(akaike.weights(aiclist)),[1],1),  

  round(data.frame(akaike.weights(aiclist))[3],3))

```

```

## https://github.com/rvlenth/emmeans/issues/288 pair-wise student t-test
pigs.emm.s<- emmeans(m7, "diet")
print("-----")
summary(pigs.emm.s)
pairs(pigs.emm.s)
p3<-gg(plot(pigs.emm.s))+
  ylab("Diet")+
  xlab(substitute(paste("EMM of TG")))
# ggsave("SL41.png",dpi = 300,units="in",font="Helvetica",width=8, height=4)

m1 <- glm(TP~ 1, family = gaussian, data = blood_test)
m2 <- glm(TP ~ 1, family = Gamma, data = blood_test)
m3 <- glm(TP ~ diet, family = gaussian, data = blood_test)
m4 <- glm(TP~ diet,family = Gamma, data = blood_test)
m5 <- glm(TP ~ tank, family = gaussian, data = blood_test)
m6 <- glm(TP~ tank,family = Gamma, data = blood_test)
m7 <- glm(TP ~ diet+tank, family = gaussian, data = blood_test)
m8 <- glm(TP~ diet+tank,family = Gamma, data = blood_test)

#m7 <- lmer(TP~ diet+(1|tank), data = blood_test)
#m8 <- glmer(TP~ diet+(1|tank),family = Gamma, data = blood_test,nAGQ=0)

aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4),AIC(m5),AIC(m6),AIC(m7),AIC(m8))
df<-
c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1],extractAIC(m5)[1],extractAIC(
m6)[1],extractAIC(m7)[1],extractAIC(m8)[1])
data.frame(round(aiclist,1),round(data.frame(akaike.weights(aiclist))[,1],1),
  round(data.frame(akaike.weights(aiclist))[,3],3))

pigs.emm.s<- emmeans(m4, "diet")
print("-----")
summary(regrid(pigs.emm.s))
contrast(regrid(pigs.emm.s),method="pairwise")
options(repr.plot.width=8, repr.plot.height=4)
p4<-gg(plot(regrid(pigs.emm.s)))+
  ylab("Diet")+
  xlab(substitute(paste("EMM of TP")))

```

```

# p6
# ggsave("BW41.png",dpi = 300,units="in",font="Helvetica",width=8, height=4)

m1 <- glm(GLU~ 1, family = gaussian, data = blood_test)
m2 <- glm(GLU ~ 1, family = Gamma, data = blood_test)
m3 <- glm(GLU ~ diet, family = gaussian, data = blood_test)
m4 <- glm(GLU~ diet,family = Gamma, data = blood_test)
m5 <- glm(GLU ~ tank, family = gaussian, data = blood_test)
m6 <- glm(GLU~ tank,family = Gamma, data = blood_test)
m7 <- glm(GLU ~ diet+tank, family = gaussian, data = blood_test)
m8 <- glm(GLU~ diet+tank,family = Gamma, data = blood_test)

#m7 <- lmer(GLU~ diet+(1|tank), data = blood_test)
#m8 <- glmer(GLU~ diet+(1|tank),family = Gamma, data = blood_test,nAGQ=0)

aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4),AIC(m5),AIC(m6),AIC(m7),AIC(m8))
df<-
c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1],extractAIC(m5)[1],extractAIC(
m6)[1],extractAIC(m7)[1],extractAIC(m8)[1])
data.frame(round(aiclist,1),round(data.frame(akaike.weights(aiclist)),[1],1),
           round(data.frame(akaike.weights(aiclist))[,3],3))

pigs.emm.s <- emmeans(m8, "diet")
print("-----")
summary(regrid(pigs.emm.s))
contrast(regrid(pigs.emm.s),method="pairwise")
options(repr.plot.width=8, repr.plot.height=4)
p5<-gg(plot(regrid(pigs.emm.s))+
  ylab("Diet")+
  xlab(substitute(paste("EMM of GLU"))))
# p6
# ggsave("BW41.png",dpi = 300,units="in",font="Helvetica",width=8, height=4)

m1 <- glm(LW~ 1, family = gaussian, data = blood_test)
m2 <- glm(LW ~ 1, family = Gamma, data = blood_test)
m3 <- glm(LW ~ diet, family = gaussian, data = blood_test)
m4 <- glm(LW~ diet,family = Gamma, data = blood_test)
m5 <- glm(LW ~ tank, family = gaussian, data = blood_test)

```

```

m6 <- glm(LW~ tank,family = Gamma, data = blood_test)
m7 <- glm(LW ~ diet+tank, family = gaussian, data = blood_test)
m8 <- glm(LW~ diet+tank,family = Gamma, data = blood_test)

#m7 <- lmer(LW~ diet+(1|tank), data = blood_test)
#m8 <- glmer(LW~ diet+(1|tank),family = Gamma, data = blood_test,nAGQ=0)

aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4),AIC(m5),AIC(m6),AIC(m7),AIC(m8))
df<-
c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1],extractAIC(m5)[1],extractAIC(
m6)[1],extractAIC(m7)[1],extractAIC(m8)[1])
data.frame(round(aiclist,1),round(data.frame(akaike.weights(aiclist)),[1],1),
           round(data.frame(akaike.weights(aiclist))[,3],3))

## https://github.com/rvlenth/emmeans/issues/288 pair-wise student t-test
pigs.emm.s <- emmeans(m3, "diet")
print("-----")
summary(pigs.emm.s)
pairs(pigs.emm.s)
p6<-gg(plot(pigs.emm.s))+
  ylab("Diet")+
  xlab(substitute(paste("EMM of LW")))
# ggsave("SL41.png",dpi = 300,units="in",font="Helvetica",width=8, height=4)
p6

m1 <- glm(LSI~ 1, family = gaussian, data = blood_test)
m2 <- glm(LSI ~ 1, family = Gamma, data = blood_test)
m3 <- glm(LSI ~ diet, family = gaussian, data = blood_test)
m4 <- glm(LSI~ diet,family = Gamma, data = blood_test)
m5 <- glm(LSI ~ tank, family = gaussian, data = blood_test)
m6 <- glm(LSI~ tank,family = Gamma, data = blood_test)
m7 <- glm(LSI ~ diet+tank, family = gaussian, data = blood_test)
m8 <- glm(LSI~ diet+tank,family = Gamma, data = blood_test)

#m7 <- lmer(LSI~ diet+(1|tank), data = blood_test)
#m8 <- glmer(LSI~ diet+(1|tank),family = Gamma, data = blood_test,nAGQ=0)

aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4),AIC(m5),AIC(m6),AIC(m7),AIC(m8))

```

```

df<-
c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1],extractAIC(m5)[1],extractAIC(
m6)[1],extractAIC(m7)[1],extractAIC(m8)[1])
data.frame(round(aiclist,1),round(data.frame(akaike.weights(aiclist)),[1],1),
           round(data.frame(akaike.weights(aiclist)),[3],3))

pigs.emm.s <- emmeans(m4, "diet")
print("-----")
summary(regrid(pigs.emm.s))
contrast(regrid(pigs.emm.s),method="pairwise")
options(repr.plot.width=8, repr.plot.height=4)
p7<-gg(plot(regrid(pigs.emm.s)))+
  ylab("Diet")+
  xlab(substitute(paste("EMM of HSI")))+
  geom_segment(aes(x = 13, y = 1, xend = 13, yend =3))+ 
  geom_segment(aes(x = 13.1,y = 1, xend = 12.9,yend=1))+ 
  geom_segment(aes(x = 13.1,y = 3, xend = 12.9, yend =3))+ 
  annotate("text",x = 13.3,y=2, label = "italic(p)~`=~0.002", parse =TRUE,size = rel(5),family="Times New
Roman")
# p6
# ggsave("BW41.png",dpi = 300,units="in",font="Helvetica",width=8, height=4)
p7

library("cowplot")
options(repr.plot.width=7.2, repr.plot.height=9.6)
all<- ggdraw() +
  draw_plot(p1, x = 0, y = 0.75, width = 0.5, height = 0.25) +
  draw_plot(p2, x = 0.5, y = 0.75, width = 0.5, height = 0.25) +
  draw_plot(p3, x = 0, y = 0.5, width = 0.5, height = 0.25) +
  draw_plot(p4, x = 0.5, y = 0.5, width = 0.5, height = 0.25) +
  draw_plot(p5, x = 0, y = 0.25, width = 0.5, height = 0.25) +
  draw_plot(p6, x = 0.5, y = 0.25, width = 0.5, height = 0.25) +
  draw_plot(p7, x = 0, y = 0, width = 0.5, height = 0.25) +
  draw_plot_label(label=c("a", "b", "c", "d", "e", "f", "g"), size=14, fontface = "plain", family="Times New Roman",
  x = c(0., 0.5, 0, 0.5, 0, 0.5, 0), y = c(1, 1, 0.75,0.75,0.5,0.5,0.25))
all
# ggsave("kegg_ora.pdf",device="pdf",units="in",font="Helvetica",width=16, height=12)
ggsave("blood_emmean.pdf",units="in",font="Times New Roman",width=7.2, height=9.6,bg="white")

```

```
figure <- ggpubr::ggarrange(p1,p2,p3,p4,p5,p6,p7,p8,
  labels = c("a", "b","c","d","e","f","",""),
  ncol = 2, nrow = 4)
options(repr.plot.width = 12, repr.plot.height = 14)
figure
ggsave("blood_emmean.png",dpi = 300,units="in",font="Helvetica",width=12, height=14)
```

Section 3.3

3.3.1 R script for artificial infection and statistical tests

```
library(tidyverse)
library("ggplot2")
library("ggridges")

pheno_0621hetero<-read.csv("190621JST_final_reform.csv")
str(pheno_0621hetero)

suppli<-pheno_0621hetero[,c(2,5,4,9,10,12,13,16,19,1)]
colnames(suppli)<-c("ID", "Diet", "Tank", "SL5", "BW5", "Ht1", "Ht2", "HC", "Tank of exposure", "ID of feeding
trail")
suppli$Diet<-gsub("1","FM",suppli$Diet)
suppli$Diet<-gsub("2","LY",suppli$Diet)
suppli$Diet<-gsub("3","PP",suppli$Diet)
suppli$Diet<-gsub("4","HY",suppli$Diet)
suppli$Diet<-gsub("5","BAC",suppli$Diet)
suppli$Tank <- gsub("1_1", "FM1", suppli$Tank)
suppli$Tank <- gsub("1_2", "FM2", suppli$Tank)
suppli$Tank <- gsub("1_3", "FM3", suppli$Tank)
suppli$Tank <- gsub("2_1", "LY1", suppli$Tank)
suppli$Tank <- gsub("2_2", "LY2", suppli$Tank)
suppli$Tank <- gsub("2_3", "LY3", suppli$Tank)
suppli$Tank <- gsub("3_1", "PP1", suppli$Tank)
suppli$Tank <- gsub("3_2", "PP2", suppli$Tank)
suppli$Tank <- gsub("3_3", "PP3", suppli$Tank)
suppli$Tank <- gsub("4_1", "HY1", suppli$Tank)
suppli$Tank <- gsub("4_2", "HY2", suppli$Tank)
suppli$Tank <- gsub("4_3", "HY3", suppli$Tank)
suppli$Tank <- gsub("5_1", "BAC1", suppli$Tank)
suppli$Tank <- gsub("5_2", "BAC2", suppli$Tank)
suppli$Tank <- gsub("5_3", "BAC3", suppli$Tank)
suppli$Tank <- gsub("5_3", "BAC3", suppli$Tank)
str(suppli)

write.csv(suppli,"AI.csv")

head(pheno_0621hetero,10)
```

```

hetero_data<-pheno_0621hetero[,c(4,5,6,7,9,10,14,16,17,18,19)]
hetero_data["SL6s5d5"]<-(pheno_0621hetero$SL_0621-
pheno_0621hetero$SL_0515)/pheno_0621hetero$SL_0515*100
hetero_data["BW6s5d5"]<-(pheno_0621hetero$BW_0621-
pheno_0621hetero$BW_0515)/pheno_0621hetero$BW_0515*100
hetero_data["TankPC"]<-factor(hetero_data$TankPC)
hetero_data["BCWdSL6s2"]<-pheno_0621hetero$BCW/pheno_0621hetero$SL_0621^2*100

hetero_data$diet<-gsub("1","FM",hetero_data$diet)
hetero_data$diet<-gsub("2","LY",hetero_data$diet)
hetero_data$diet<-gsub("3","PP",hetero_data$diet)
hetero_data$diet<-gsub("4","HY",hetero_data$diet)
hetero_data$diet<-gsub("5","BAC",hetero_data$diet)
hetero_data$diet<-factor(hetero_data$diet, levels=c("FM","PP","BAC","LY","HY"))
hetero_data["FM_proportion"]<-gsub("FM",0.8,hetero_data$diet)
hetero_data["FM_proportion"]<-gsub("PP",0.5,hetero_data$FM_proportion)
hetero_data["FM_proportion"]<-gsub("BAC",0.5,hetero_data$FM_proportion)
hetero_data["FM_proportion"]<-gsub("LY",0.5,hetero_data$FM_proportion)
hetero_data["FM_proportion"]<-gsub("HY",0.3,hetero_data$FM_proportion)
hetero_data["FM_proportion"]<-as.numeric(hetero_data$FM_proportion)

hetero_data["source"]<-gsub("LY","Y",hetero_data$diet)
hetero_data["source"]<-gsub("HY","Y",hetero_data$source)
hetero_data$source<-factor(hetero_data$source, levels=c("FM","PP","BAC","Y"))

hetero_data["tank"]<-gsub("1_","FM_t",hetero_data$tank)
hetero_data["tank"]<-gsub("2_","LY_t",hetero_data$tank)
hetero_data["tank"]<-gsub("3_","PP_t",hetero_data$tank)
hetero_data["tank"]<-gsub("4_","HY_t",hetero_data$tank)
hetero_data["tank"]<-gsub("5_","BAC_t",hetero_data$tank)
hetero_data$tank<-factor(hetero_data$tank, levels=c("FM_t1","FM_t2","FM_t3",
"PP_t1","PP_t2","PP_t3",
"BAC_t1","BAC_t2","BAC_t3",
"LY_t1","LY_t2","LY_t3",
"HY_t1","HY_t2","HY_t3"))

pheno_0621hetero[is.na(pheno_0621hetero$Ht2),]

```

```

head(hetero_data)

cor(hetero_data[,c(3,4,5,6,7,8,12,13,14,15)]]

pvalue<-function(x){
  m<-round(shapiro.test(x)[2]$p.value,3)
  return(m)
}

t(group_by(hetero_data,diet)%>%summarise_at(vars("BCWdSL6s2"),list(~round(mean(.),1),~round(sd(.),1),~round(pvalue(.),3)))) 

t(group_by(hetero_data,diet)%>%summarise_at(vars("BCW"),list(~round(mean(.),1),~round(sd(.),1),~round(pv
alue(.),3)))) 

t(group_by(hetero_data,diet)%>%summarise_at(vars("Ht_ave"),list(~round(mean(.),1),~round(sd(.),1),~round(
pvalue(.),3)))) 

t(group_by(hetero_data,diet)%>%summarise_at(vars("SL_0515"),list(~round(mean(.),1),~round(sd(.),1),~round
(pvalue(.),3)))) 

t(group_by(hetero_data,diet)%>%summarise_at(vars("SL_0621"),list(~round(mean(.),1),~round(sd(.),1),~round
(pvalue(.),3)))) 

t(group_by(hetero_data,diet)%>%summarise_at(vars("BW_0515"),list(~round(mean(.),1),~round(sd(.),1),~roun
d(pvalue(.),3)))) 

t(group_by(hetero_data,diet)%>%summarise_at(vars("BW_0621"),list(~round(mean(.),1),~round(sd(.),1),~roun
d(pvalue(.),3)))) 

options(repr.plot.width = 6, repr.plot.height = 8)
ggplot(hetero_data, aes(x = BCW, y = TankPC)) + geom_density_ridges(scale = 0.9,)+ 
  geom_jitter(aes(color=diet),height=0.1)+ 
  scale_fill_brewer(palette="Accent",name="Diet")

library(extrafont)

y<-list(hetero_data$BCW,hetero_data$BCWdSL6s2,hetero_data$Ht_ave,                                     hetero_data$SL6s5d5,
hetero_data$BW6s5d5)
ylab<-c("HC","HD","Ht", substitute(paste("SL"[italic("54")])), substitute(paste("BW"[italic("54")])))

```

```
ylab[4]
```

```
options(repr.plot.width = 4, repr.plot.height = 4)  
line_th_unit<-0.6  
txt_size<-12
```

```
padjustment<-function(p1){  
  p1$theme$line$size<-line_th_unit  
  p1$theme$line$size<-line_th_unit  
  p1$theme$axis.ticks$size<-line_th_unit  
  p1$theme$axis.line$size<-line_th_unit  
  #p1$theme$panel.grid$colour<-"black"  
  p1$theme$axis.ticks$colour<-"black"  
  return(p1)  
}
```

```
boxplotgg<-function(i){  
  p1 <- ggplot(hetero_data,aes(y=unlist(y[i]),x=diet)) +  
    labs(x = "Diet",y=ylab[i]) +  
    # scale_fill_manual(values=c"red", "blue","limegreen","purple","yellow"),name="Diet") +  
    geom_boxplot() +  
    theme_bw(base_family="Times New Roman",base_line_size = line_th_unit,base_size = txt_size) +  
    theme(axis.text.y= element_text(size=txt_size),  
          axis.text.x = element_text(size=txt_size),  
          axis.title= element_text(size=txt_size),  
          text=element_text(size=txt_size))  
  p1<-padjustment(p1)  
  return(p1)  
}
```

```
psl<-boxplotgg(4)+ labs(x = "Diet",y=substitute(paste("SL"[italic("54")])))  
pbw<-boxplotgg(5)+ labs(x = "Diet",y=substitute(paste("BW"[italic("54")])))
```

```
figure <- ggpubr::ggarrange(boxplotgg(1),boxplotgg(2),boxplotgg(3),psl,  
                             pbw,  
                             labels = c("a", "b","c","d","e"),  
                             ncol = 2, nrow = 3, font.label = list(size = 14, face="plain",
```

```

color = "black",family = "Times New Roman"))
options(repr.plot.width = 12, repr.plot.height = 12)
figure
# ggsave("blood_box.png",dpi = 300,units="in",font="Helvetica",width=12, height=14)

ggsave("Fig. 3.pdf",device="pdf",units="in",width =7.2,height = 7.2)
embed_fonts("Fig. 3.pdf", outfile="Fig. 3_embed.pdf")

library("cowplot")
options(repr.plot.width=12, repr.plot.height=8)
all<- ggdraw() +
  draw_plot(p_diet, x = 0, y = 0.66, width = 0.5, height = 0.33) +
  # draw_plot(p_FM_proportion, x = 0.32, y = 0.64, width = 0.16, height = 0.32) +
  draw_plot(p_HD, x = 0.5, y = 0.66, width = 0.5, height = 0.33) +
  draw_plot(p_ht, x = 0, y = 0.33, width = 0.5, height = 0.33) +
  draw_plot(p_sl, x = 0.5, y = 0.33, width = 0.5, height = 0.33) +
  draw_plot(p_bw, x = 0, y = 0, width = 0.5, height = 0.33) +
  # draw_plot(p_fedtank, x = 0, y = 0, width = 1, height = 0.5) +
  draw_plot_label(label = c("a", "b", "c", "d", "e"), size =txt_size,family="serif",
                 x = c(0., 0.5, 0.,0.5, 0), y = c(1, 1, 0.66,0.66,0.33))
all
#ggsave("Fig_HC.pdf",device="pdf",units="in",font="Helvetica",width=16, height=10)
ggsave("Fig_HC.png",device="png",dpi = 300,units="in",font="Helvetica",width=12, height=10)

hetero_data$diet<-factor(hetero_data$diet, levels=c("HY","LY","BAC","PP","FM"))
options(repr.plot.width = 6, repr.plot.height =10)
p_HD_group <- ggplot(hetero_data,aes(x=BCWdSL6s2,y=diet)) +
  labs(x = "HD",y="Diet") +
  # scale_fill_manual(values=c"red", "blue","limegreen","purple","yellow"),name="Diet") +
  #geom_density()+
  geom_density_ridges(scale = 0.5,quantile_lines = TRUE)+
  geom_jitter(height=0.1)+

  #geom_jitter(height=0.1,aes(color=TankPC))+
  #facet_wrap(~diet,scale="free")+
  theme_bw()+
  #scale_fill_brewer(palette="Accent",name="Diet")+

```

```

theme(axis.text.y= element_text(size=txt_size),
      axis.text.x = element_text(size=txt_size),
      axis.title= element_text(size=txt_size),
      text=element_text(size=txt_size, family="serif"))

hetero_data$diet<-factor(hetero_data$diet, levels=c("FM","PP","BAC","LY","HY"))

p_HD_group

#ggsave("Fig_HD.png",device="png",dpi = 300,units="in",font="Helvetica",width=12, height=10)

#hetero_data$diet<-factor(hetero_data$diet, levels=c("HY","LY","BAC","PP","FM"))
options(repr.plot.width = 15, repr.plot.height = 4)
p_Hc_group <- ggplot(hetero_data,aes(x=BCW)) +
  labs(x = "HC",y="Density") +
  # scale_fill_manual(values=c"red", "blue","limegreen","purple","yellow"),name="Diet") +
  geom_density()+
  #geom_density_ridges(scale = 0.5,)+
  geom_jitter(height=0, aes(x=BCW, y=0))+ 
  #geom_jitter(height=0.1)+ 
  facet_wrap(~diet,nrow=1)+ 
  theme_bw()+
  #scale_fill_brewer(palette="Accent",name="Diet")+
  theme(axis.text.y= element_text(size=txt_size),
        axis.text.x = element_text(size=txt_size),
        axis.title= element_text(size=txt_size),
        text=element_text(size=txt_size, family="serif"))

#hetero_data$diet<-factor(hetero_data$diet, levels=c("FM","PP","BAC","LY","HY"))


```

```

p_Hc_group

#ggsave("Fig_HD.png",device="png",dpi = 300,units="in",font="Helvetica",width=12, height=8)

#hetero_data$diet<-factor(hetero_data$diet, levels=c("HY","LY","BAC","PP","FM"))
options(repr.plot.width = 15, repr.plot.height = 4)
p_HD_group <- ggplot(hetero_data,aes(x=BCWdSL6s2)) +
  labs(x = "HD",y="Density") +
  # scale_fill_manual(values=c"red", "blue","limegreen","purple","yellow"),name="Diet") +
  geom_density()+
  #geom_density_ridges(scale = 0.5,)+
  geom_jitter(height=0, aes(x=BCWdSL6s2, y=0))+


```

```

#geom_jitter(height=0.1) +
facet_wrap(~diet,nrow=1) +
theme_bw() +
#scale_fill_brewer(palette="Accent",name="Diet") +
theme(axis.text.y= element_text(size=txt_size),
axis.text.x = element_text(size=txt_size),
axis.title= element_text(size=txt_size),
text=element_text(size=txt_size, family="serif"))

#hetero_data$diet<-factor(hetero_data$diet, levels=c("FM","PP","BAC","LY","HY"))

p_HD_group
#ggsave("Fig_HD.png",device="png",dpi = 300,units="in",font="Helvetica",width=12, height=8)

#hetero_data$diet<-factor(hetero_data$diet, levels=c("HY","LY","BAC","PP","FM"))
options(repr.plot.width = 15, repr.plot.height = 4)
p_Ht_group <- ggplot(hetero_data,aes(x=Ht_ave)) +
  labs(x = "Ht",y="Density") +
  # scale_fill_manual(values=c"red", "blue","limegreen","purple","yellow"),name="Diet") +
  geom_density()+
  #geom_density_ridges(scale = 0.5,)+
  geom_jitter(height=0, aes(x=Ht_ave, y=0))+ 
  #geom_jitter(height=0.1) +
  facet_wrap(~diet,nrow=1) +
  theme_bw() +
  #scale_fill_brewer(palette="Accent",name="Diet") +
  theme(axis.text.y= element_text(size=txt_size),
axis.text.x = element_text(size=txt_size),
axis.title= element_text(size=txt_size),
text=element_text(size=txt_size, family="serif"))

#hetero_data$diet<-factor(hetero_data$diet, levels=c("FM","PP","BAC","LY","HY"))

p_Ht_group
#ggsave("Fig_HD.png",device="png",dpi = 300,units="in",font="Helvetica",width=12, height=8)

library("cowplot")
options(repr.plot.width=12, repr.plot.height=8)
all<- ggdraw() +
  draw_plot(p_Hc_group, x = 0, y = 0.66, width = 1, height = 0.33) +

```

```

# draw_plot(p_FM_proportion, x = 0.32, y = 0.64, width = 0.16, height = 0.32) +
draw_plot(p_HD_group, x = 0, y = 0.33, width = 1, height = 0.33) +
draw_plot(p_Ht_group, x = 0, y = 0, width = 1, height = 0.33) +

draw_plot_label(label = c("a", "b", "c"), size = txt_size,family="serif",
x = c(0,0, 0), y = c(1, 0.66, 0.33))

all

#ggsave("Fig_HC.pdf",device="pdf",units="in",font="Helvetica",width=16, height=10)
ggsave("Fig_HC_density.png",device="png",dpi = 300,units="in",font="Helvetica",width=12, height=10)

library("car")
# mean, sd and pvalue of normtest for each diet
SL_mean_sd_test<-group_by(hetero_data,TankPC)%>%summarise_at(vars("BCW"),list(~round(mean(.),3),
~round(sd(.),3),
~round(pvalue(.),3)
))

SL_mean_sd_test
leveneTest(BCW~TankPC,hetero_data)
bartlett.test(BCW~TankPC,hetero_data)

packageVersion("emmeans")

library("qpcR") #1.4.1

m1 <- glm(BCWdSL6s2+1 ~ diet,family = gaussian, data = hetero_data)
m2 <- glm(BCWdSL6s2+1 ~ diet+TankPC ,family = gaussian, data = hetero_data)
m3 <- glm(BCWdSL6s2+1 ~ diet,family = Gamma, data = hetero_data)
m4 <- glm(BCWdSL6s2+1 ~ diet+TankPC ,family = Gamma, data = hetero_data)

aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4))
df<-c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1])
data.frame(df,round(aiclist,1),round(data.frame(akaike.weights(aiclist))[,1],1),
round(data.frame(akaike.weights(aiclist))[,3],3))

## https://github.com/rvlenth/emmeans/issues/288 pair-wise student t-test
pigs.emm.s<- emmeans::emmeans(m2, "diet")
summary(pigs.emm.s)

```

```

pairs(pigs.emm.s)
phr<-gg(plot(pigs.emm.s))+  

  ylab("Tanks for exposure")+
  xlab(substitute(paste("EMM of (HR+1)")))
phr
# ggsave("SL41.png",dpi = 300,units="in",font="Helvetica",width=8, height=4)

bartlett.test(BCW ~ diet, data = hetero_data[hetero_data$diet == c("FM","PP"),])
bartlett.test(BCW ~ diet, data = hetero_data[hetero_data$diet == c("FM","BAC"),])
bartlett.test(BCW ~ diet, data = hetero_data[hetero_data$diet == c("FM","LY"),])
bartlett.test(BCW ~ diet, data = hetero_data[hetero_data$diet == c("FM","HY"),])

bartlett.test(BCWdSL6s2 ~ diet, data = hetero_data[hetero_data$diet == c("FM","PP"),])
bartlett.test(BCWdSL6s2 ~ diet, data = hetero_data[hetero_data$diet == c("FM","BAC"),])
bartlett.test(BCWdSL6s2~ diet, data = hetero_data[hetero_data$diet == c("FM","LY"),])
bartlett.test(BCWdSL6s2 ~ diet, data = hetero_data[hetero_data$diet == c("FM","HY"),])

ggplot2()

m1 <- glm(BCW+1~ 1, family = gaussian, data = hetero_data)
m2 <- glm(BCW+1 ~ 1, family = Gamma, data = hetero_data)
m3 <- glm(BCW+1 ~ diet, family = gaussian, data = hetero_data)
m4 <- glm(BCW+1~ diet,family = Gamma, data = hetero_data)
m5 <- glm(BCW+1 ~ TankPC, family = gaussian, data = hetero_data)
m6 <- glm(BCW+1~ TankPC,family = Gamma, data = hetero_data)
m7 <- glm(BCW+1 ~ diet+TankPC, family = gaussian, data = hetero_data)
m8 <- glm(BCW+1~ diet+TankPC,family = Gamma, data = hetero_data)
#m8 <- glm(BCW+1 ~ diet+TankPC+SL_0621,family = gaussian, data = hetero_data)

#m7 <- lmer(BCW+1~ diet+(1|tank), data = hetero_data)
#m8 <- glmer(BCW+1~ diet+(1|tank),family = Gamma, data = hetero_data,nAGQ=0)

aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4),AIC(m5),AIC(m6),AIC(m7),AIC(m8))
df<-
c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1],extractAIC(m5)[1],extractAIC(
m6)[1],extractAIC(m7)[1],extractAIC(m8)[1])
data.frame(round(aiclist,1),round(data.frame(akaike.weights(aiclist))[1],1),
  round(data.frame(akaike.weights(aiclist))[3],3))

```

```

m1 <- glm(BCW~ 1, family = poisson, data = hetero_data)
#m2 <- glm(BCW+1 ~ 1, family = Gamma, data = hetero_data)
m2 <- glm.nb(BCW~ 1, data = hetero_data)

m3 <- glm(BCW ~ diet, family = poisson, data = hetero_data)
#m4 <- glm(BCW+1~ diet,family = Gamma, data = hetero_data)
m4 <- glm.nb(BCW~ diet, data = hetero_data)

m5 <- glm(BCW ~ TankPC, family = poisson, data = hetero_data)
# m6 <- glm(BCW+1~ TankPC,family = Gamma, data = hetero_data)
m6 <- glm.nb(BCW~ TankPC, data = hetero_data)
m7 <- glm(BCW ~ diet+TankPC, family = poisson, data = hetero_data)
#m8 <- glm(BCW+1~ diet+TankPC,family = Gamma, data = hetero_data)
m8 <- glm.nb(BCW~ diet+TankPC, data = hetero_data)

#m8 <- glm(BCW+1 ~ diet+TankPC+SL_0621,family = gaussian, data = hetero_data)

#m7 <- lmer(BCW+1~ diet+(1|tank), data = hetero_data)
#m8 <- glmer(BCW+1~ diet+(1|tank),family = Gamma, data = hetero_data,nAGQ=0)

aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4),AIC(m5),AIC(m6),AIC(m7),AIC(m8))
df<-
c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1],extractAIC(m5)[1],extractAIC(
m6)[1],extractAIC(m7)[1],extractAIC(m8)[1])
data.frame(round(aiclist,1),round(data.frame(akaike.weights(aiclist)),[1],1),
           round(data.frame(akaike.weights(aiclist))[,3],3))

gg<-function(p){
  txt_size<-20
  pg<- p+
    #scale_y_discrete(limits = rev(c("FM","PP","BAC","LY","HY")))+ 
    coord_flip()+
    theme_bw()+
    theme(axis.text.y= element_text(size=txt_size),
          axis.text.x = element_text(size=txt_size),
          axis.title= element_text(size=txt_size),
          text=element_text(size=txt_size, family="serif"))
}

```

```
}
```

```
## https://github.com/rvlenth/emmeans/issues/288 pair-wise student t-test
pigs.emm.s <- emmeans::emmeans(m6, "TankPC")
summary(pigs.emm.s)
pairs(pigs.emm.s)
pt<-gg(plot(pigs.emm.s))+
  ylab("Tanks for exposure")+
  xlab(substitute(paste("EMM of HC")))
# pt
# ggsave("SL41.png",dpi = 300,units="in",font="Helvetica",width=8, height=4)
pt
```

```
m1 <- glm(BCWdSL6s2~ 1, family = gaussian, data = hetero_data)
#m2 <- glm(BCWdSL6s2 ~ 1, family = Gamma, data = hetero_data)
m3 <- glm(BCWdSL6s2~ diet, family = gaussian, data = hetero_data)
#m4 <- glm(BCWdSL6s2+1~ diet,family = Gamma, data = hetero_data)
m5 <- glm(BCWdSL6s2 ~ TankPC, family = gaussian, data = hetero_data)
#m6 <- glm(BCWdSL6s2+1~ TankPC,family = Gamma, data = hetero_data)
m7 <- glm(BCWdSL6s2 ~ diet+TankPC, family = gaussian, data = hetero_data)
#m8 <- glm(BCWdSL6s2+1~ diet+TankPC,family = Gamma, data = hetero_data)
#m7 <- lmer(BCWdSL6s2+1~ diet+(1|tank), data = hetero_data)
#m8 <- glmer(BCWdSL6s2+1~ diet+(1|tank),family = Gamma, data = hetero_data,nAGQ=0)
```

```
aiclist<-c(AIC(m1),AIC(m3),AIC(m5),AIC(m7))
df<-c(extractAIC(m1)[1],extractAIC(m3)[1],extractAIC(m5)[1],extractAIC(m7)[1])
data.frame(round(aiclist,1),round(data.frame(akaike.weights(aiclist))[,1],1),
  round(data.frame(akaike.weights(aiclist))[,3],3))
```

```
## https://github.com/rvlenth/emmeans/issues/288 pair-wise student t-test
pigs.emm.s <- emmeans::emmeans(m5, "TankPC")
summary(pigs.emm.s)
pairs(pigs.emm.s)
pt<-gg(plot(pigs.emm.s))+
  ylab("Tanks for exposure")+
```

```

xlab(substitute(paste("EMM of (HC+1)")))
# pt
# ggsave("SL41.png",dpi = 300,units="in",font="Helvetica",width=8, height=4)
pt

m1 <- glm(Ht_ave~ 1, family = gaussian, data = hetero_data)
m2 <- glm(Ht_ave ~ 1, family = Gamma, data = hetero_data)
m3 <- glm(Ht_ave ~ diet, family = gaussian, data = hetero_data)
m4 <- glm(Ht_ave~ diet,family = Gamma, data = hetero_data)
m5 <- glm(Ht_ave ~ TankPC, family = gaussian, data = hetero_data)
m6 <- glm(Ht_ave~ TankPC,family = Gamma, data = hetero_data)
m7 <- glm(Ht_ave ~ diet+TankPC, family = gaussian, data = hetero_data)
m8 <- glm(Ht_ave~ diet+TankPC,family = Gamma, data = hetero_data)
#m7 <- lmer(BCWdSL6s2+1~ diet+(1|tank), data = hetero_data)
#m8 <- glmer(BCWdSL6s2+1~ diet+(1|tank),family = Gamma, data = hetero_data,nAGQ=0)

aiclist<-c(AIC(m1),AIC(m2),AIC(m3),AIC(m4),AIC(m5),AIC(m6),AIC(m7),AIC(m8))
df<-
c(extractAIC(m1)[1],extractAIC(m2)[1],extractAIC(m3)[1],extractAIC(m4)[1],extractAIC(m5)[1],extractAIC(
m6)[1],extractAIC(m7)[1],extractAIC(m8)[1])
data.frame(round(aiclist,1),round(data.frame(akaike.weights(aiclist))[,1],1),
           round(data.frame(akaike.weights(aiclist))[,3],3))

m1 <- glm(SL6s5d5~ 1, family = gaussian, data = hetero_data)
#m2 <- glm(SL6s5d5+1 ~ 1, family = Gamma, data = hetero_data)
m3 <- glm(SL6s5d5 ~ diet, family = gaussian, data = hetero_data)
#m4 <- glm(SL6s5d5+1~ diet,family = Gamma, data = hetero_data)
m5 <- glm(SL6s5d5 ~ tank, family = gaussian, data = hetero_data)
#m6 <- glm(SL6s5d5+1~ tank,family = Gamma, data = hetero_data)
m7 <- glm(SL6s5d5 ~ diet+tank, family = gaussian, data = hetero_data)
#m8 <- glm(SL6s5d5+1~ diet+tank,family = Gamma, data = hetero_data)
#m7 <- lmer(BCWdSL6s2+1~ diet+(1|tank), data = hetero_data)
#m8 <- glmer(BCWdSL6s2+1~ diet+(1|tank),family = Gamma, data = hetero_data,nAGQ=0)

aiclist<-c(AIC(m1),AIC(m3),AIC(m5),AIC(m7))
df<-c(extractAIC(m1)[1],extractAIC(m3)[1],extractAIC(m5)[1],extractAIC(m7)[1])
data.frame(round(aiclist,1),round(data.frame(akaike.weights(aiclist))[,1],1),
           round(data.frame(akaike.weights(aiclist))[,3],3))

```

```

round(data.frame(akaike.weights(aiclist)[,3],3))

sum(hetero_data$SL6s5d5<=0)
sum(hetero_data$BW6s5d5<=0)
hetero_data$BW6s5d5[hetero_data$BW6s5d5<=0]

library("emmeans")
## https://github.com/rvlenth/emmeans/issues/288 pair-wise student t-test
pigs.emm.s <- emmeans(m3, "diet")
print("-----")
summary(pigs.emm.s)
pairs(pigs.emm.s)

m1 <- glm(BW6s5d5~ 1, family = gaussian, data = hetero_data)
#m2 <- glm(BW6s5d5+10 ~ 1, family = Gamma, data = hetero_data)
m3 <- glm(BW6s5d5 ~ diet, family = gaussian, data = hetero_data)
#m4 <- glm(BW6s5d5+10~ diet,family = Gamma, data = hetero_data)
m5 <- glm(BW6s5d5 ~ tank, family = gaussian, data = hetero_data)
#m6 <- glm(BW6s5d5+10~ tank,family = Gamma, data = hetero_data)
m7 <- glm(BW6s5d5~ diet+tank, family = gaussian, data = hetero_data)
#m8 <- glm(BW6s5d5+10~ diet+tank,family = Gamma, data = hetero_data)
#m7 <- lmer(BCWdSL6s2+1~ diet+(1|tank), data = hetero_data)
#m8 <- glmer(BCWdSL6s2+1~ diet+(1|tank),family = Gamma, data = hetero_data,nAGQ=0)

aiclist<-c(AIC(m1),AIC(m3),AIC(m5),AIC(m7))
df<-c(extractAIC(m1)[1],extractAIC(m3)[1],extractAIC(m5)[1],extractAIC(m7)[1])
data.frame(round(aiclist,1),round(data.frame(akaike.weights(aiclist))[,1],1),
           round(data.frame(akaike.weights(aiclist))[,3],3))

library("emmeans")
## https://github.com/rvlenth/emmeans/issues/288 pair-wise student t-test
pigs.emm.s <- emmeans(m7, "diet")
print("-----")
summary(pigs.emm.s)
pairs(pigs.emm.s)

figure <- ggpubr::ggarrange(pt,p1,p2,p3,p4,phr,

```

```

labels = c("a", "b","c","d","e","f"),
ncol = 2, nrow = 3)

options(repr.plot.width = 10, repr.plot.height = 10)
figure
ggsave("ai_emmean.png",dpi = 300,units="in",font="Helvetica",width=10, height=10,bg="white")

options(repr.plot.width = 7, repr.plot.height = 13)
#plot a simple scatter plot
p1 <- ggplot(hetero_data,aes(x=Ht_ave,y=BCW,color=factor(diet))) +
  geom_point(shape=19) +
  geom_smooth(method=lm, se=FALSE) +
  ylim(0,80) +
  labs(x = "Ht (%)",y="HC") +
  #scale_fill_manual(values=c("red", "blue","limegreen","purple","yellow"),name="Diet")+
  scale_colour_brewer(palette="Accent",name="Diet")+

  theme(axis.text.y= element_text(size = rel(1.2)),
        axis.text.x = element_text(size = rel(1.2)),
        legend.position = "none",
        text=element_text(size=10, family="serif"))+
  guides(colour = guide_legend())+
  theme_bw()

#plot a simple scatter plot
p2 <- ggplot(hetero_data,aes(x=Ht_ave,y=BCW)) +
  geom_point(shape=19) +
  geom_smooth(method=lm,se=FALSE) +
  ylim(0,80) +
  labs(x = "Ht (%)",y="HC") +
  #scale_fill_manual(values=c("red", "blue","limegreen","purple","yellow"),name="Diet")+
  scale_fill_brewer(palette="Accent",name="Diet")+
  theme(axis.text.y= element_text(size = rel(1.2)),
        axis.text.x = element_text(size = rel(1.2)),
        text=element_text(size=10, family="serif"))+
  geom_text(parse=TRUE, aes(35,70,label = 'paste(italic("r = "), "0.467")'),color = "black")+
  theme_bw()

p_all<-plot_grid(p2,p1,nrow = 2, labels = c("a","b"), align = 'v', axis = 'lr')

```

p_all

```
library("emmeans")
## https://github.com/rvlenth/emmeans/issues/288 pair-wise student t-test
pigs.emm.s<- emmeans(m4, "diet")
print("-----")
summary(pigs.emm.s)
pairs(pigs.emm.s)

options(repr.plot.width=8, repr.plot.height=4)
plot(pigs.emm.s, comparisons = TRUE, ylab="Diet", xlab= "emmmean of HC",)

options(repr.plot.width=8, repr.plot.height=4)
pwpp(pigs.emm.s, ylab="HC")

pigs.emm.s<- emmeans(m4, "TankPC")
print("-----")
summary(pigs.emm.s)
pairs(pigs.emm.s)

gg<-function(p){
  txt_size<-20
  pg<- p+
    theme_bw()+
    theme(axis.text.y= element_text(size=txt_size),
          axis.text.x = element_text(size=txt_size),
          axis.title= element_text(size=txt_size),
          text=element_text(size=txt_size, family="serif"))}
```

```
}

pigs.emm.s<- emmeans(m1, "diet")
print("-----")
summary(pigs.emm.s)
pairs(pigs.emm.s)
options(repr.plot.width=8, repr.plot.height=4)
p1<-gg(plot(pigs.emm.s,comparisons = TRUE))+  
  ylab("Diet")+
  xlab(substitute(paste("Estimated marginal mean of SL"[italic("t1 ")])))
p1
options(repr.plot.width=8, repr.plot.height=4)
p1p<-gg(pwpp(pigs.emm.s))+
  ylab(substitute(paste("SL"[italic("t1 ")])))
p1p
```

Section 3.4

3.4.1 Bash script for NGS analysis

```
# download fa and gtf file  
conda install -c bioconda -y ucsc-genepredtobed  
conda install -c bioconda -y ucsc-genepredtogtf  
conda install -c bioconda -y ucsc-gftogenepred  
conda install -c bioconda -y ucsc-bedtogenepred  
conda install -c bioconda -y ucsc-gff3togenepred  
conda install -c bioconda -y genomepy  
genomepy search -p UCSC FUGU5  
genomepy install fr3 --annotation
```

```
gzip -d $DIR/fr3/fr3.annotation.gtf.gz
```

```
# index  
cd ~/brbseq20210526  
mkdir STAR_Index/  
STAR --runMode genomeGenerate --genomeDir STAR_Index/ --genomeFastaFiles ./fr3/fr3.fa --  
sjdbGTFfile ./fr3/fr3.annotation.gtf
```

```
# trim & alignment  
Fastaq_dir1=~/210401_NB551362_0125_AH3M57BGXJ/Unaligned_210426  
Fastaq_dir2=~/210408_NB551362_0126_AH3M3NBGXJ/Unaligned_210427  
for i in `ls $Fastaq_dir1| grep Pool |grep R2 |grep 001.fastq.gz`; do java -jar BRBseqTools-1.6.jar Trim -f  
$Fastaq_dir1/$i; m=`echo $i|sed s/001.fastq/001.trimmed.fastq/g`; STAR --runMode alignReads --genomeDir  
STAR_Index/ --outFilterMultimapNmax 1 --readFilesCommand zcat --outSAMtype BAM Unsorted --  
outFileNamePrefix BAM/ --readFilesIn $Fastaq_dir1/$m --runThreadN 10; n=`echo $i|sed  
s/001.fastq.gz/001.bam/g`; mv BAM/Aligned.out.bam BAM/$n; done  
for i in `ls $Fastaq_dir2| grep Pool |grep R2 |grep 001.fastq.gz`; do java -jar BRBseqTools-1.6.jar Trim -f  
$Fastaq_dir2/$i; m=`echo $i|sed s/001.fastq/001.trimmed.fastq/g`; STAR --runMode alignReads --genomeDir  
STAR_Index/ --outFilterMultimapNmax 1 --readFilesCommand zcat --outSAMtype BAM Unsorted --  
outFileNamePrefix BAM/ --readFilesIn $Fastaq_dir2/$m --runThreadN 10; n=`echo $i|sed  
s/001.fastq.gz/001.bam/g`; mv BAM/Aligned.out.bam BAM/$n; done
```

```
# demultiplex and matrix  
for i in `ls $Fastaq_dir1| grep Pool |grep R2 |grep 001.fastq.gz`  
do
```

```

j=`echo $i|sed s/R2/R1/g`
n=`echo $i|sed s/001.fastq.gz/001.bam/g`
v=`echo $i|sed s/_R2_001.fastq.gz//g`
mkdir $v/
java -jar BRBseqTools-1.6.jar CreateDGEMatrix -f $Fastaq_dir1/$j -b BAM/$n -c barcodes_16p051mRF02L.txt
-gtf ./fr3/fr3.annotation.gtf -p BU -UMI 10 -o $v
done

```

```

for i in `ls $Fastaq_dir2| grep Pool |grep R2 |grep 001.fastq.gz`'
do
j=`echo $i|sed s/R2/R1/g`
n=`echo $i|sed s/001.fastq.gz/001.bam/g`
v=`echo $i|sed s/_R2_001.fastq.gz//g`
mkdir $v/
java -jar BRBseqTools-1.6.jar CreateDGEMatrix -f $Fastaq_dir2/$j -b BAM/$n -c barcodes_16p053mRF02L.txt
-gtf ./fr3/fr3.annotation.gtf -p BU -UMI 10 -o $v
done

```

```

# files names for post analysis
for i in `ls | grep Pool`'
do
echo $i >> files.txt
done

```

```

Fastaq_dir3=~/BRBseq_JP-KNI-20200731-06
for i in `ls $Fastaq_dir3| grep R2.fastq.gz `; do java -jar BRBseqTools-1.6.jar Trim -f $Fastaq_dir3/$i; m=`echo
$ii|sed s/.fastq/.trimmed.fastq/g`; STAR --runMode alignReads --genomeDir STAR_Index/ --
outFilterMultimapNmax 1 --readFilesCommand zcat --outSAMtype BAM Unsorted --outFileNamePrefix
BAM_20/ --readFilesIn $Fastaq_dir3/$m ; n=`echo $i|sed s/.fastq.gz/.bam/g`; mv BAM_20/Aligned.out.bam
BAM_20/$n; done

```

```

for i in `ls $Fastaq_dir3| grep R2.fastq.gz `'
do
j=`echo $i|sed s/R2/R1/g`
n=`echo $i|sed s/.fastq.gz/.bam/g`
v=`echo $i|sed s/_R2.fastq.gz//g`
mkdir $v/

```

```
java -jar BRBseqTools-1.6.jar CreateDGEMatrix -f $Fastaq_dir3/$j -b BAM_20/$n -c barcodes_20.txt -  
gtf ./fr3/fr3.annotation.gtf -p BU -UMI 10 -o $v  
done
```

3.4.2 R script for NGS analysis

```
# r=3.6.1
library(tidyverse)

pheno <- read_table2("pheno.txt")
pheno$diet<-as.character(pheno$diet)

filename<-read_table2("files.txt",col_names=FALSE)

reads <- list()
UMIs <- list()
for (i in filename$X1[c(1,2,3,4,17,18,19,20)]){
  #set filepath
  readsfp <- paste("./",i,"/output.dge.reads.txt", sep = "")
  umisfp <- paste("./",i,"/output.dge.umis.txt",sep= "")
  #import data
  reads[[i]] <- read_table2(readsfp)
  UMIs[[i]] <- read_table2(umisfp)
}

## make sure all IDs are in pheno and BRBLibReads/UMIs objects and its order be consistent
#check <- function(x){
#  if(all(pheno>Name %in% x)) print("OK") else print("not OK")
#}
#check(colnames(reads[[1]][-1]))
checkIDs <- function(){
  print("check data consistency")
  check <- function(x){
    if(all(pheno>Name %in% x)) print("OK") else print("not OK")
  }
  for (i in 1:length(reads)){
    readscol <- eval(parse(text = paste("colnames(reads[",i,""])[-1]",sep = "")))
    UMIscol <- eval(parse(text = paste("colnames(UMIs[",i,""])[-1]",sep = "")))
    paste("library: L",i,sep = "") %>% print()
    print("Checking reads")
    check(readscol)
    print("Checking UMIs")
    check(UMIscol)
  }
}
```

```

}

}

checkIDs()

## Re-order count data according to row names of pheno object
for (i in 1:length(reads)){
  eval(parse(text = paste("reads[[" , i, "]] <- reads[[" , i, "]] %>% select(Gene_id, as.character(pheno$Name))", sep = " ")))
  eval(parse(text = paste("UMIs[[" , i, "]] <- UMIs[[" , i, "]] %>% select(Gene_id, as.character(pheno$Name))", sep = " ")))
}
checkIDs()

for (i in 1:8){
  print(paste("L",i,"mean:",round(mean(colSums(reads[[i]][,-1])),2),"sd:",round(sd(colSums(reads[[i]][,-1])),2),
             "length",length(colSums(reads[[i]][,-1]))))
}

m=cbind(colSums(reads[[1]][,-1]),colSums(reads[[2]][,-1]),colSums(reads[[3]][,-1]),colSums(reads[[4]][,-1]),
         colSums(reads[[5]][,-1]),colSums(reads[[6]][,-1]),colSums(reads[[7]][,-1]),colSums(reads[[8]][,-1]))

write_csv(data.frame(m),"readcount.csv")

m=cbind(colSums(reads[[1]][,-1]),colSums(reads[[2]][,-1]),colSums(reads[[3]][,-1]),colSums(reads[[4]][,-1]),
         colSums(reads[[5]][,-1]),colSums(reads[[6]][,-1]),colSums(reads[[7]][,-1]),colSums(reads[[8]][,-1]))

add_all<-reads[[1]][,-1]+reads[[2]][,-1]+reads[[3]][,-1]+reads[[4]][,-1]+
  reads[[5]][,-1]+reads[[6]][,-1]+reads[[7]][,-1]+reads[[8]][,-1]
colSums(add_all)

reads <- list()
UMIs <- list()
for (i in filesname$X1[c(25,26,27,28,29,30)]){
  #set filepath
  readsfp <- paste("./",i,"/output.dge.reads.txt", sep = "")
  umisfp <- paste("./",i,"/output.dge.umis.txt",sep= "")
  #import data
  reads[[i]] <- read_table2(readsfp)
  UMIs[[i]] <- read_table2(umisfp)
}

```

```

}

pheno <- read_table2("JST_lowfishmeal_lin.txt")
pheno$sample_id<-as.character(pheno$sample_id)
str(pheno)

## make sure all IDs are in pheno and BRBLibReads/UMIs objects and its order be consistent

#check <- function(x){
#  if(all(pheno>Name %in% x)) print("OK") else print("not OK")
#}

#check(colnames(reads[[1]][-1]))

checkIDs <- function(){
  print("check data consistency")
  check <- function(x){
    if(all(pheno$sample_id %in% x)) print("OK") else print("not OK")
  }
  for (i in 1:length(reads)){
    readscol <- eval(parse(text = paste("colnames(reads[",i,""])[-1]",sep = "")))
    UMIscol <- eval(parse(text = paste("colnames(UMIs[",i,""])[-1]",sep = "")))
    paste("library: L",i,sep = "") %>% print()
    print("Checking reads")
    check(readscol)
    print("Checking UMIs")
    check(UMIscol)
  }
}
checkIDs()

## Re-order count data according to row names of pheno object

for (i in 1:length(reads)){
  eval(parse(text = paste("reads[",i,""] <- reads[["i,""]] %>% select(Gene_id, as.character(pheno$sample_id))",
sep = "")))
  eval(parse(text = paste("UMIs[",i,""] <- UMIs[["i,""]] %>% select(Gene_id, as.character(pheno$sample_id))",
sep = "")))
}
checkIDs()

m=cbind(colSums(reads[[1]][,-1]),colSums(reads[[2]][,-1]),colSums(reads[[3]][,-1]),colSums(reads[[4]][,-1]),
colSums(reads[[5]][,-1]),colSums(reads[[6]][,-1]))
write_csv(data.frame(m),"readcount_20.csv")

```

3.4.3 R script for data preprocessing, differential gene expression analysis and gene set analysis

```
# https://xueyidong.github.io/RNAseq123CN/articles/limmaWorkflow.html#examining-the-number-of-de-
genes-1

library(tidyverse)

# r=3.6.1

pheno <- read_table("pheno.txt",

  col_types = cols(Name = col_character(),

    tank = col_character(),

    diet = col_character()))


filename <- read.table("files.txt")

reads <- list()

UMIs <- list()

for (i in filename$V1[c(1,2,3,4,17,18,19,20,25)]){

  #set filepath

  readsfp <- paste("./",i,"/output.dge.reads.txt", sep = "")

  umisfp <- paste("./",i,"/output.dge.umis.txt",sep= "")

  #import data

  reads[[i]] <- read_table2(readsfp,col_types = cols(.default = "?",

    Gene_id = col_character()))

  UMIs[[i]] <- read_table2(umisfp,col_types = cols(.default = "?",

    Gene_id = col_character()))

}

## make sure all IDs are in pheno and BRBLibReads/UMIs objects and its order be consistent

#check <- function(x){

#  if(all(pheno>Name %in% x)) print("OK") else print("not OK")
```

```

# }

#check(colnames(reads[[1]][-1]))

checkIDs <- function(){

  print("check data consistency")

  check <- function(x){

    if(all(x %in% pheno$Name)) print("OK") else print("not OK")

  }

  for (i in 1:length(reads)){

    readscol <- eval(parse(text = paste("colnames(reads[\"",i,"\"])-1]",sep = "")))

    UMIscol <- eval(parse(text = paste("colnames(UMIs[\"",i,"\"])-1]",sep = "")))

    paste("library: L",i,sep = "") %>% print()

    print("Checking reads")

    check(readscol)

    print("Checking UMIs")

    check(UMIscol)

  }

}

checkIDs()

```

```

sample_id1<-colnames(reads[[1]])[-1]

sample_id2<-colnames(reads[[5]])[-1]

sample_id3<-colnames(reads[[9]])[-1]

sample_id1_tank<-c()

sample_id1_diet<-c()

for (i in sample_id1){

  sample_id1_tank<-c(sample_id1_tank,pheno[which(pheno$Name==i),][1,2]$tank)

  sample_id1_diet<-c(sample_id1_diet,pheno[which(pheno$Name==i),][1,3]$diet)
}

```

```

}

sample_id2_tank<-c()

sample_id2_diet<-c()

for (i in sample_id2){

sample_id2_tank<-c(sample_id2_tank,pheno[which(pheno$Name==i),][1,2]$tank)

sample_id2_diet<-c(sample_id2_diet,pheno[which(pheno$Name==i),][1,3]$diet)

}

sample_id3_tank<-c()

sample_id3_diet<-c()

for (i in sample_id3){

sample_id3_tank<-c(sample_id3_tank,pheno[which(pheno$Name==i),][1,2]$tank)

sample_id3_diet<-c(sample_id3_diet,pheno[which(pheno$Name==i),][1,3]$diet)

}

length(reads[[1]][-1])

length(reads[[5]][-1])

length(reads[[9]][-1])

(length(reads[[1]][-1]+length(reads[[5]][-1]))*4+length(reads[[9]][-1]))

# colSums(reads[[9]][-1])

# read count per fihs

mean(colSums(reads[[9]][-1]))

mean(colSums((reads[[1]][-1]+reads[[2]][-1]+reads[[3]][-1]+reads[[4]][-1])))

mean(colSums((reads[[5]][-1]+reads[[6]][-1]+reads[[7]][-1]+reads[[8]][-1])))

# read count per fihs

sd(colSums(reads[[9]][-1]))

```

```

sd(colSums((reads[[1]][-1]+reads[[2]][-1]+reads[[3]][-1]+reads[[4]][-1])))

sd(colSums((reads[[5]][-1]+reads[[6]][-1]+reads[[7]][-1]+reads[[8]][-1])))

m = cbind((reads[[1]][-1]+reads[[2]][-1]+reads[[3]][-1]+reads[[4]][-1]),
           (reads[[5]][-1]+reads[[6]][-1]+reads[[7]][-1]+reads[[8]][-1]),reads[[9]][-1])

n = cbind((UMIs[[1]][-1]+UMIs[[2]][-1]+UMIs[[3]][-1]+UMIs[[4]][-1]),
           (UMIs[[5]][-1]+UMIs[[6]][-1]+UMIs[[7]][-1]+UMIs[[8]][-1]),UMIs[[9]][-1])

length(m)

length(n)

head(m)

head(n)

str(m)

sum(colSums(m))

mean(colSums(m))

sd(colSums(m))

batch1=1

batch2=1

data<-data.frame(merged_id=as.numeric(colnames(m)),
                  sample_id= as.numeric(c(rep(sample_id1, batch1),rep(sample_id2,batch2),sample_id3)),

library=as.factor(c(rep("Lib2",length(sample_id1)*batch1),rep("Lib3",length(sample_id2)*batch2),rep("Lib1",length(sample_id3)))),

```

```

diet= as.factor(c(rep(sample_id1_diet, batch1),rep(sample_id2_diet,batch2),sample_id3_diet)),
tank= as.factor(c(rep(sample_id1_tank, batch1),rep(sample_id2_tank,batch2),sample_id3_tank)))

str(data)

summary(data)

#ddbj

id_list<-c()
for (i in data$merged_id){
  id_list<-rbind(id_list,paste("LFMrna_",i))
}
write.csv(id_list,"ddbj_id.csv")

library(edgeR)

readmats<-m %>% as.matrix()

rownames(readmats) <- reads[[1]]$Gene_id

x <- DGEList(counts=readmats,group=factor(data$diet))

x$samples$library<-data$library

x$samples$sample_id<-data$sample_id

# remove sample id = 125, which has low reads count

x<-x[,data$sample_id != 125]

# Removing genes that are lowly expressed

```

```

keep.exprs <- filterByExpr(x, group=data$diet)

d_f <- x[keep.exprs, keep.lib.sizes=TRUE]

dim(d_f)

str(d_f)

sum(d_f$count)

cpm <- cpm(x)

lcpm <- cpm(x, log=TRUE)

cpm_f <- cpm(d_f)

lcpm_f <- cpm(d_f, log=TRUE)

# for plotting

L <- mean(d_f$samples$lib.size) * 1e-6

M <- median(d_f$samples$lib.size) * 1e-6

c(L, M)

(lcpm.cutoff <- log2(10/M + 2/L))

library(RColorBrewer)

options(repr.plot.width = 18, repr.plot.height = 6)

png(file="~/brbseq20210526/raw_filtered_norm.png",width=10, height=4, units = "in",res = 300)

test_factor<-x$samples$library

col <- brewer.pal(length(levels(test_factor)), "Set2")

par(mfrow=c(1,3))

plot(density(lcpm[,1]), col=col[as.numeric(test_factor[1])], ylim=c(0,1.5), lwd=0.5, las=2, main="", xlab="")

```

```

title(main="a. Raw data", xlab="Log-cpm")

abline(v=lcpm.cutoff, lty=3)

for (i in 2:length(test_factor)){
  den <- density(lcpm[,i])
  lines(den$x, den$y, col=col[as.numeric(test_factor[i])], lwd=0.5)
}

for (i in 1:length(levels(test_factor))){
  # legend_date<-factor(c("Library_210401","Library_210408","Library_200731"))
  legend_date<-factor(c("Lib2","Lib3","Lib1"))

  legend("topright", levels(legend_date), text.col=col, bty="n",y.intersp=1.5)

  # legend("topright", levels(test_factor), text.col=col, bty="n")
}

test_factor<-d_f$samples$library

plot(density(lcpm_f[,1]), col=col[as.numeric(test_factor[1])],ylim=c(0,1.5), lwd=0.5, las=2, main="", xlab="")

title(main="b. Filtered data", xlab="Log-cpm")

abline(v=lcpm.cutoff, lty=3)

for (i in 2:length(test_factor)){
  den <- density(lcpm_f[,i])
  lines(den$x, den$y, col=col[as.numeric(test_factor[i])], lwd=0.5)
}

for (i in 1:length(levels(test_factor))){
  legend("topright", levels(legend_date), text.col=col, bty="n",y.intersp=1.5)
}

```

```

#legend("topright", levels(test_factor), text.col=col, bty="n")
}

# Norm

d_n <- calcNormFactors(d_f, method = "TMM")

lcpm_fn<-cpm(d_n, log=TRUE)

test_factor<-x$samples$library

plot(density(lcpm_fn[,1]), col=col[as.numeric(test_factor[1])], ylim=c(0,1.5), lwd=0.5, las=2, main="", xlab="")

title(main="c. Filterd and normalized data", xlab="Log-cpm")

abline(v=lcpm.cutoff, lty=3)

for (i in 2:length(test_factor)){
  den <- density(lcpm_fn[,i])
  lines(den$x, den$y, col=col[as.numeric(test_factor[i])], lwd=0.5)
}

for (i in 1:length(levels(test_factor))){
  # legend_date<-factor(c("Library_210401","Library_210408","Library_200731"))

  legend_date<-factor(c("Lib2","Lib3","Lib1"))

  legend("topright", levels(legend_date), text.col=col, bty="n",y.intersp=1.5)
}

dev.off()

python_gmatrix<-t(lcpm_fn)

row.names(python_gmatrix)<-colnames(lcpm_fn)

```

```

str(python_gmatrix)

write.table(python_gmatrix,"python_2.gmatrix")

python_pmatrix<-data[data$sample_id != 125,]

str(python_pmatrix)

write.table(python_pmatrix,"python_2.pmatrix")

lcpm <- cpm(d_f, log=TRUE)

par(mfrow=c(1,2))

col.group <- data$library[data$sample_id != 125]

levels(col.group) <- brewer.pal(nlevels(col.group), "Set1")

col.group <- as.character(col.group)

col.lane <- data$diet[data$sample_id != 125]

levels(col.lane) <- brewer.pal(nlevels(col.lane), "Set2")

col.lane <- as.character(col.lane)

plotMDS(lcpm, labels=data$library[data$sample_id != 125], col=col.group)

title(main="A. Sample groups")

plotMDS(lcpm, labels=data$diet[data$sample_id != 125], col=col.lane, dim=c(3,4))

title(main="B. Sequencing lanes")

#suppli

suppli<-data[,c(2,3,4,5)]

colnames(suppli)<-c("NGS_ID","Library","Diet","Tank")

suppli$Diet <- gsub("1", "FM", suppli$Diet)

suppli$Diet <- gsub("2", "LY", suppli$Diet)

suppli$Diet <- gsub("3", "PP", suppli$Diet)

suppli$Diet <- gsub("4", "HY", suppli$Diet)

```

```

suppli$Diet <- gsub("5", "BAC", suppli$Diet)

suppli$Tank <- gsub("1_1", "FM1", suppli$Tank)
suppli$Tank <- gsub("1_2", "FM2", suppli$Tank)
suppli$Tank <- gsub("1_3", "FM3", suppli$Tank)
suppli$Tank <- gsub("2_1", "LY1", suppli$Tank)
suppli$Tank <- gsub("2_2", "LY2", suppli$Tank)
suppli$Tank <- gsub("2_3", "LY3", suppli$Tank)
suppli$Tank <- gsub("3_1", "PP1", suppli$Tank)
suppli$Tank <- gsub("3_2", "PP2", suppli$Tank)
suppli$Tank <- gsub("3_3", "PP3", suppli$Tank)
suppli$Tank <- gsub("4_1", "HY1", suppli$Tank)
suppli$Tank <- gsub("4_2", "HY2", suppli$Tank)
suppli$Tank <- gsub("4_3", "HY3", suppli$Tank)
suppli$Tank <- gsub("5_1", "BAC1", suppli$Tank)
suppli$Tank <- gsub("5_2", "BAC2", suppli$Tank)
suppli$Tank <- gsub("5_3", "BAC3", suppli$Tank)
str(suppli)
suppli<-suppli%>%arrange(Library)%>%arrange(NGS_ID)

```

```
write.csv(suppli,"suppli.csv")
```

```
# write.table(python_pmatrix,"python_2.pmatrix")
```

```
diet<-data$diet[data$sample_id != 125]
```

```
library<-data$library[data$sample_id != 125]
```

```
diet <- gsub("1", "FM", diet)
```

```

diet <- gsub("2", "LY", diet)
diet <- gsub("3", "PP", diet)
diet <- gsub("4", "HY", diet)
diet <- gsub("5", "BAC", diet)
design <- model.matrix(~0+diet)

colnames(design) <- gsub("diet", "", colnames(design))
colnames(design) <- gsub("batch", "", colnames(design))

str(design)

contr.matrix <- makeContrasts(
  PP.vs.FM = PP-FM,
  BAC.vs.FM = BAC-FM ,
  LY.vs.FM = LY-FM ,
  HY.vs.FM = HY-FM,
  # diet2v4 = diet2 - diet4,
  levels = colnames(design))
#levels = c("FM","PP","BAC","LY","HY"))

contr.matrix

library("limma")

packageVersion("limma")

# Transform count data to log2-counts per million (logCPM),
# estimate the mean-variance relationship and use this
# to compute appropriate observation-level weights.
# The data are then ready for linear modelling.

```

```

options(repr.plot.width = 8, repr.plot.height = 8)

v <- voom(d_n, design, plot=TRUE,)

# fit lm

vfit <- lmFit(v, design)

vfit <- contrasts.fit(vfit, contrasts=contr.matrix)

# empirical Bayes moderation is carried out by borrowing information

# across all the genes to obtain more precise estimates of gene-wise variability (Smyth 2004).

efit <- eBayes(vfit)

plotSA(efit, main="Final model: Mean-variance trend")

options(repr.plot.width = 8, repr.plot.height = 8)

# Examining the number of DE genes

# Significance is defined using an adjusted p-value cutoff that is set at 5% by default.

def<-decideTests(efit)

summary(def)

# The treat method (McCarthy and Smyth 2009) can be used to calculate p-values from

# empirical Bayes moderated t-statistics with a minimum log-FC requirement.

tfit <- treat(vfit, lfc=1.2)

dt <- decideTests(tfit)

summary(dt)

# install.packages("ggvenn")          # Install & load ggvenn

library("ggvenn")

```

```

packageVersion("ggvenn")

flist<-c()

for (i in c(1,2,3,4)) {

  test1<-def[,1:4]@.Data[,i]

  flist[[i]]<-unlist(def@dimnames[1])[!(test1==0)]

}

line_th_unit<-0.6

txt_size<-12

padjustment<-function(p1){

  p1$theme$line$size<-line_th_unit

  p1$theme$line$size<-line_th_unit

  #p1$theme$axis.ticks$size<-line_th_unit

  #p1$theme$axis.line$size<-line_th_unit

  #p1$theme$panel.grid$colour<-"black"

  #p1$theme$axis.ticks$colour<-"black"

  return(p1)

}

library(extrafont)

loadfonts(device = "pdf",quiet = TRUE)

ggvenn2 <- function(data, columns = NULL,

  show_elements = FALSE,

  show_percentage = TRUE,

```

```

digits = 1,
fill_color = c("blue", "yellow", "green", "red"),
fill_alpha = .5,
stroke_color = "black",
stroke_alpha = 1,
stroke_size = 1,
stroke_linetype = "solid",
set_name_color = "black",
set_name_size = 6,
text_color = "black",
text_size = 4,
label_sep = ",",
count_column = NULL,
show_outside = c("auto", "none", "always"),
auto_scale = FALSE) {
show_outside <- match.arg(show_outside)
venn <- prepare_venn_data(data, columns, show_elements, show_percentage, digits,
label_sep, count_column, show_outside, auto_scale)
g <- venn$shapes %>%
mutate(group = LETTERS[group]) %>%
ggplot() +
geom_polygon(aes(x = x, y = y, group = group, fill = group),
alpha = fill_alpha)
if (nrow(venn$labels) > 0) {
g <- g +
geom_text(data = venn$labels,
aes(x = x, y = y, label = text),

```

```

color = set_name_color,
size = 12,
family = "Times New Roman"

)

#geom_polygon(aes(x = x, y = y, group = group),
#      fill = NA,
#      color = stroke_color,
#      size = stroke_size,
#      alpha = stroke_alpha,
#      linetype = stroke_linetype)
}

if (nrow(venn$texts) > 0) {
  g <- g +
    geom_text(data = venn$texts,
              aes(x = x, y = y, label = text),
              color = text_color,
              size = 12,
              family = "Times New Roman"
)
}

if (nrow(venn$segs) > 0) {
  g <- g +
    geom_segment(data = venn$segs,
                 aes(x = x, y = y, xend = xend, yend = yend),
                 color = text_color,
                 size = 0.5)
}

```

```

}

g <- g +
  scale_fill_manual(values = fill_color) +
  guides(fill = "none") +
  #coord_fixed() +
  theme_void()

return(g)
}

```

```

gen_element_df_4 <- function() {

  df <- tribble(~name, ~A, ~B, ~C, ~D,
    "A", TRUE, FALSE, FALSE, FALSE,
    "B", FALSE, TRUE, FALSE, FALSE,
    "C", FALSE, FALSE, TRUE, FALSE,
    "D", FALSE, FALSE, FALSE, TRUE,
    "AB", TRUE, TRUE, FALSE, FALSE,
    "BC", FALSE, TRUE, TRUE, FALSE,
    "CD", FALSE, FALSE, TRUE, TRUE,
    "AC", TRUE, FALSE, TRUE, FALSE,
    "BD", FALSE, TRUE, FALSE, TRUE,
    "AD", TRUE, FALSE, FALSE, TRUE,
    "ABC", TRUE, TRUE, TRUE, FALSE,
    "BCD", FALSE, TRUE, TRUE, TRUE,
    "ACD", TRUE, FALSE, TRUE, TRUE,
    "ABD", TRUE, TRUE, FALSE, TRUE,
    "ABCD", TRUE, TRUE, TRUE, TRUE,
  )
}

```

```

"-", FALSE, FALSE, FALSE, FALSE)

stopifnot(all((df %>% count(A, B, C, D) %>% with(n)) == 1))

return(df %>% mutate(n = 0, text = ""))
}

```

```

gen_circle <- function(group, x_offset = 0, y_offset = 0, radius = 1,
                       radius_b = radius, theta_offset = 0, length.out = 100) {

  tibble(group = group,
         theta = seq(0, 2 * pi, length.out = length.out)) %>%
    mutate(x_raw = radius * cos(theta),
          y_raw = radius_b * sin(theta),
          x = x_offset + x_raw * cos(theta_offset) - y_raw * sin(theta_offset),
          y = y_offset + x_raw * sin(theta_offset) + y_raw * cos(theta_offset))
}

}

```

```

calc_scale_info_4 <- function(auto_scale, n_sets, max_scale_diff = 5) {
  if (auto_scale) {
    stop("Error: 'auto_scale' parameter is supported for only two set venn so far.")
  }
  return(NULL)
}

```

```
min_overlap_for_text <- 0.2
```

```

gen_circle_4 <- function() {
  rbind(gen_circle(1L, -.7, -1/2, .75, 1.5, pi/4),
    gen_circle(2L, -.72+2/3, -1/6, .75, 1.5, pi/4),
    gen_circle(3L, .72-2/3, -1/6, .75, 1.5, -pi/4),
    gen_circle(4L, .7, -1/2, .75, 1.5, -pi/4))
}

```

```

gen_text_pos_4 <- function() {
  tribble(~name, ~x, ~y, ~hjust, ~vjust,
    "A", -1.5, 0, 0.5, 0.5,
    "B", -0.6, 0.7, 0.5, 0.5,
    "C", 0.6, 0.7, 0.5, 0.5,
    "D", 1.5, 0, 0.5, 0.5,
    "AB", -0.9, 0.3, 0.5, 0.5,
    "BC", 0, 0.4, 0.5, 0.5,
    "CD", 0.9, 0.3, 0.5, 0.5,
    "AC", -0.8, -0.9, 0.5, 0.5,
    "BD", 0.8, -0.9, 0.5, 0.5,
    "AD", 0, -1.4, 0.5, 0.5,
    "ABC", -0.5, -0.2, 0.5, 0.5,
    "BCD", 0.5, -0.2, 0.5, 0.5,
    "ACD", -0.3, -1.1, 0.5, 0.5,
    "ABD", 0.3, -1.1, 0.5, 0.5,
    "ABCD", 0, -0.7, 0.5, 0.5,
    "-",
    "0", -1.9, 0.5, 0.5)
}

```

```

gen_seg_pos_4 <- function(scale_info) {
  df <- tibble(x = 0, y = 0, xend = 0, yend = 0)[-1,]

```

```

return(df)
}

gen_label_pos_4 <- function() {
  tribble(~name, ~x, ~y, ~hjust, ~vjust,
  "A", -1.5, -1.3, 1, 1,
  "B", -0.8, 1.2, 0.5, 0,
  "C", 0.8, 1.2, 0.5, 0,
  "D", 1.5, -1.3, 0, 1)
}

prepare_venn_data <- function(data, columns = NULL,
  show_elements = FALSE, show_percentage = TRUE, digits = 1,
  label_sep = ",", count_column = NULL,
  show_outside = "auto", auto_scale = FALSE) {
  if (is.data.frame(data)) {
    if (is.null(columns)) {
      columns = data %>% select_if(is.logical) %>% names
    }
    if (!identical(show_elements, FALSE)) {
      stopifnot(is.character(show_elements))
      show_elements <- show_elements[[1]]
      if (!(show_elements %in% names(data))) {
        stop("`show_elements` should be one column name of the data frame")
      }
    }
    if (length(columns) == 2) {
      stopifnot(is.logical(as_tibble(data)[,columns[[1]]], drop = TRUE))
    }
  }
}

```

```

stopifnot(is.logical(as_tibble(data)[,columns[[2]]], drop = TRUE))

df_element <- gen_element_df_2()

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

  idx <- ((!xor(df_element$A[[i]], as_tibble(data)[,columns[[1]]])) &
    (!xor(df_element$B[[i]], as_tibble(data)[,columns[[2]]])))

  if (is.null(count_column)) {

    df_element$n[[i]] <- sum(idx)

  } else {

    df_element$n[[i]] <- sum(as_tibble(data)[,count_column][idx,])

  }

  if (!identical(show_elements, FALSE)) {

    df_element$text[[i]] <- paste(unlist(as_tibble(data)[idx,show_elements]), collapse = label_sep)

  }

}

scale_info <- calc_scale_info_2(auto_scale, df_element$n)

df_shape <- gen_circle_2(scale_info)

df_text <- gen_text_pos_2(scale_info) %>% inner_join(df_element, by = "name")

df_label <- gen_label_pos_2(scale_info)

df_seg <- gen_seg_pos_2(scale_info)

} else if (length(columns) == 3) {

  stopifnot(is.logical(as_tibble(data)[,columns[[1]]], drop = TRUE))

  stopifnot(is.logical(as_tibble(data)[,columns[[2]]], drop = TRUE))

  stopifnot(is.logical(as_tibble(data)[,columns[[3]]], drop = TRUE))

  df_element <- gen_element_df_3()

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

    idx <- ((!xor(df_element$A[[i]], as_tibble(data)[,columns[[1]]])) &
      (!xor(df_element$B[[i]], as_tibble(data)[,columns[[2]]]))) &
      (!xor(df_element$C[[i]], as_tibble(data)[,columns[[3]]])))
  }
}

```

```

(!xor(df_element$C[[i]], as_tibble(data)[,columns[[3]]])))

if (is.null(count_column)) {

  df_element$n[[i]] <- sum(idx)

} else {

  df_element$n[[i]] <- sum(as_tibble(data)[,count_column][idx,])

}

if (!identical(show_elements, FALSE)) {

  df_element$text[[i]] <- paste(unlist(as_tibble(data)[idx,show_elements]), collapse = label_sep)

}

scale_info <- calc_scale_info_3(auto_scale, df_element$n)

df_shape <- gen_circle_3()

df_text <- gen_text_pos_3() %>% inner_join(df_element, by = "name")

df_label <- gen_label_pos_3()

df_seg <- gen_seg_pos_3(scale_info)

} else if (length(columns) == 4) {

  stopifnot(is.logical(as_tibble(data)[,columns[[1]]], drop = TRUE))

  stopifnot(is.logical(as_tibble(data)[,columns[[2]]], drop = TRUE))

  stopifnot(is.logical(as_tibble(data)[,columns[[3]]], drop = TRUE))

  stopifnot(is.logical(as_tibble(data)[,columns[[4]]], drop = TRUE))

  df_element <- gen_element_df_4()

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

    idx <- ((df_element$A[[i]] == as_tibble(data)[,columns[[1]]], drop = TRUE) &

             (df_element$B[[i]] == as_tibble(data)[,columns[[2]]], drop = TRUE) &

             (df_element$C[[i]] == as_tibble(data)[,columns[[3]]], drop = TRUE) &

             (df_element$D[[i]] == as_tibble(data)[,columns[[4]]], drop = TRUE))

    if (is.null(count_column)) {

```

```

df_element$n[[i]] <- sum(idx)

} else {

df_element$n[[i]] <- sum(as_tibble(data)[,count_column][idx,])

}

if (!identical(show_elements, FALSE)) {

df_element$text[[i]] <- paste(unlist(as_tibble(data)[idx,show_elements])), collapse = label_sep)

}

scale_info <- calc_scale_info_4(auto_scale, df_element$n)

df_shape <- gen_circle_4()

df_text <- gen_text_pos_4() %>% inner_join(df_element, by = "name")

df_label <- gen_label_pos_4()

df_seg <- gen_seg_pos_4(scale_info)

} else {

stop("logical columns in data.frame `data` or vector `columns` should be length between 2 and 4")

}

df_label <- df_label %>% mutate(text = columns)

show_elements <- !identical(show_elements, FALSE)

} else if (is.list(data)) {

if (is.null(columns)) {

columns <- names(data) %>% head(4)

}

a2 <- unique(unlist(data[columns]))

if (length(columns) == 2) {

df_element <- gen_element_df_2()

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

idx <- ((!xor(df_element$A[[i]], a2 %in% data[,columns[[1]]]))) &

```

```

(!xor(df_element$B[[i]], a2 %in% data[[columns[[2]]]])))

df_element$n[[i]] <- sum(idx)

df_element$text[[i]] <- paste(a2[idx], collapse = label_sep)

}

scale_info <- calc_scale_info_2(auto_scale, df_element$n)

df_shape <- gen_circle_2(scale_info)

df_text <- gen_text_pos_2(scale_info) %>% inner_join(df_element, by = "name")

df_label <- gen_label_pos_2(scale_info)

df_seg <- gen_seg_pos_2(scale_info)

} else if (length(columns) == 3) {

df_element <- gen_element_df_3()

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

idx <- ((!xor(df_element$A[[i]], a2 %in% data[[columns[[1]]]])) &

(!xor(df_element$B[[i]], a2 %in% data[[columns[[2]]]])) &

(!xor(df_element$C[[i]], a2 %in% data[[columns[[3]]]])))

df_element$n[[i]] <- sum(idx)

df_element$text[[i]] <- paste(a2[idx], collapse = label_sep)

}

scale_info <- calc_scale_info_3(auto_scale, df_element$n)

df_shape <- gen_circle_3()

df_text <- gen_text_pos_3() %>% inner_join(df_element, by = "name")

df_label <- gen_label_pos_3()

df_seg <- gen_seg_pos_3(scale_info)

} else if (length(columns) == 4) {

df_element <- gen_element_df_4()

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

idx <- ((!xor(df_element$A[[i]], a2 %in% data[[columns[[1]]]])) &

```

```

(!xor(df_element$B[[i]], a2 %in% data[[columns[[2]]]])) &
(!xor(df_element$C[[i]], a2 %in% data[[columns[[3]]]])) &
(!xor(df_element$D[[i]], a2 %in% data[[columns[[4]]]]))

df_element$n[[i]] <- sum(idx)

df_element$text[[i]] <- paste(a2[idx], collapse = label_sep)

}

scale_info <- calc_scale_info_4(auto_scale, df_element$n)

df_shape <- gen_circle_4()

df_text <- gen_text_pos_4() %>% inner_join(df_element, by = "name")

df_label <- gen_label_pos_4()

df_seg <- gen_seg_pos_4(scale_info)

} else {

stop("list `data` or vector `column` should be length between 2 and 4")

}

df_label <- df_label %>% mutate(text = columns)

} else {

stop("`data` should be either a list or a data.frame")

}

if ((show_outside == "none") || (show_outside == "auto" & df_text$n[[nrow(df_text)]] == 0)) {

if (df_text$n[[nrow(df_text)]] > 0)

warning("Although not display in plot, outside elements are still count in percentages.")

df_text <- df_text[-nrow(df_text), ]

}

if (!show_elements) {

if (show_percentage) {

fmt <- sprintf("%%.%d\n(%%.%.%df%%.%%)", digits)

df_text <- df_text %>% mutate(text = sprintf(fmt, n, 100 * n / sum(n)))

```

```

} else {

df_text <- df_text %>% mutate(text = sprintf("%d", n))

}

}

list(shapes = df_shape, texts = df_text, labels = df_label, segs = df_seg)
}

```

mvenn\$labels

c("PPFM", "BAC-FM", "LY-FM", "HY-FM")

col <- brewer.pal(5, "Accent")

length(col)

col

fill_alpha=0.5

set_name_color="black"

text_color="black"

fill_color=c("blue", "yellow", "green", "red")

fill_color = c(col[2],col[3],col[4],col[5])

options(repr.plot.width = 10, repr.plot.height = 8)

g <- mvenn\$shapes %>%

mutate(group = LETTERS[group]) %>%

ggplot() +

geom_polygon(aes(x = x, y = y, group = group, fill = group),

alpha = fill_alpha) +

```

geom_polygon(aes(x = x, y = y, group = group),
             fill = NA,
             color = "black",
             size = 0.6,
           )+

```

```

#geom_text(data = mvenn$labels,
           #aes(x = x, y = y, label = text),
           # color = set_name_color,
           #size = 12,
           # family = "Arial"
         #)+
```

```

#geom_text(data = mvenn$texts,
           #     aes(x = x, y = y, label = n),
           #     color = text_color,
           #     size = 12,
           #     family = "Times New Roman"
         #     )+
```

```

scale_fill_manual(values = fill_color) +
guides(fill = "none") +
coord_fixed() +
theme_void()
```

g

mvenn\$texts

f<-ggvenn2(a,

```

show_percentage=FALSE,fill_alpha = 0.2)+

#theme_bw(base_family="Times New Roman",base_line_size = line_th_unit,base_size = txt_size )+  

theme_void()

f

c("PPFM", "BAC-FM", "LY-FM", "HY-FM")

ggsave("Fig. 4.pdf",device="pdf",units="in",width = 4,height = 4)

embed_fonts("Fig. 4.pdf", outfile="Fig. 4_embed.pdf")

save(a, file="a.RData")

load("a.RData")

a<- list(`PP-FM` = flist[[1]],

`BAC-FM` = flist[[2]],

`LY-FM` = flist[[3]],

`HY-FM` = flist[[4]])

f<-ggvenn(a, c("PP-FM", "BAC-FM", "LY-FM", "HY-FM"),

show_percentage=FALSE, fill_alpha = 0.3,stroke_size = line_th_unit)+

theme(legend.text =element_text(size=txt_size,family="Times New Roman"))

#theme_bw(base_family="Times New Roman",base_line_size = line_th_unit,base_size = txt_size )+  

  

# f<-padjustment(f)# draw two-set venn  

  

#ggvenn(a, c("Set 1", "Set 2", "Set 3")) # draw three-set venn  

  

#ggvenn(a)

```

f

```
# the number of genes that are DE in all comparisons  
  
#de.common <- which(dt[,1]!=0 & dt[,3]!=0)  
  
#length(de.common)  
  
d<-vennDiagram(def[,1:4], circle.col=c("turquoise", "salmon","yellow","purple"))
```

```
PP.vs.FM <- topTreat(efit, coef=1, n=Inf)
```

```
BAC.vs.FM <- topTreat(efit, coef=2, n=Inf)
```

```
LY.vs.FM <- topTreat(efit, coef=3, n=Inf)
```

```
HY.vs.FM <- topTreat(efit, coef=4, n=Inf)
```

```
head(PP.vs.FM,n=1)
```

```
head(BAC.vs.FM,n=20)#582
```

```
head(LY.vs.FM, n=8)
```

```
head(HY.vs.FM, n=17)
```

```
packageVersion("ggvenn")
```

```
options(repr.plot.width = 12, repr.plot.height = 12)
```

```
# graphical representation of DE results
```

```
png(file="~/brbseq20210526/SA_DE_genes.png",width=10, height=10, units = "in",res = 300)
```

```
par(mfrow=c(2,2))
```

```
plotMD(efit, column=1, status=def[,1], main=paste("a.", colnames(efit)[1]))
```

```
plotMD(efit, column=2, status=def[,2], main=paste("b.", colnames(efit)[2]))
```

```
plotMD(efit, column=3, status=def[,3], main=paste("c.", colnames(efit)[3]))
```

```
plotMD(efit, column=4, status=def[,4], main=paste("d.", colnames(efit)[4]))
```

```

dev.off()

options(repr.plot.width = 12, repr.plot.height = 12)

# graphical representation of DE results

par(mfrow=c(2,2))

plotMD(tfit, column=1, status=dt[,1], main=colnames(tfit)[1])

plotMD(tfit, column=2, status=dt[,2], main=colnames(tfit)[2])

plotMD(tfit, column=3, status=dt[,3], main=colnames(tfit)[3])

plotMD(tfit, column=4, status=dt[,4], main=colnames(tfit)[4])

# ensemble id

gene_id<-gsub(".2","",row.names(d_n$count),fixed=TRUE)

gene_id<-data.frame(gsub(".1","",gene_id,fixed=TRUE))

colnames(gene_id)<-"ensembl_gene_id"

gene_id$ensembl_gene_id<-as.character(gene_id$ensembl_gene_id)

str(gene_id$ensembl_gene_id)

library("biomaRt")

packageVersion("biomaRt")

listEnsembl(version = 98)

ensembl98 <- useEnsembl(biomart = 'ensembl',
                         version = 98)

ensembl104 <- useEnsembl(biomart = 'ensembl',
                         version = 104)

```

```
version = 104)
```

```
searchDatasets(mart = ensembl98, pattern = "FUGU")
```

```
searchDatasets(mart=ensembl104,pattern = "zebra")
```

```
fugumart <- useDataset("trubripes_gene_ensembl",mart=ensembl98)
```

```
zebramart<-useDataset("mzebra_gene_ensembl",mart=ensembl104)
```

```
searchAttributes(mart = fugumart, pattern = "symbol")
```

```
head(searchAttributes(mart = fugumart, pattern = "go"))
```

```
head(searchAttributes(mart = fugumart, pattern = "chr"))
```

```
head(searchAttributes(mart = fugumart, pattern = "pos"))
```

```
hg_symbols<-
getBM(attributes=c('ensembl_gene_id',"zfin_id_symbol","goslim_goa_description","entrezgene_id","description","go_id","start_position","end_position"),
      filters= 'ensembl_gene_id',
      values= gene_id$ensembl_gene_id,
      mart = fugumart)
```

```
head(hg_symbols,10)
```

```
# goslim
```

```
biomart_t2g = hg_symbols%>% dplyr::select(goslim_goa_description,entrezgene_id) %>% as.data.frame()
```

```

str(biomart_t2g)

biomart_go= hg_symbols%>% dplyr::select(goslim_goa_description,go_id) %>% as.data.frame()

str(biomart_go)

head(biomart_go)

# https://cran.r-project.org/web/packages/msigdbr/vignettes/msigdbr-intro.html

#library("msigdbr")

#cgp_gene_sets = msigdbr(species = "Danio rerio",category = "C5")

#msigdbr_t2g = cgp_gene_sets %>% dplyr::distinct(gs_name, entrez_gene) %>% as.data.frame()

library("rvcheck")

packageVersion("rvcheck")

# http://www.bioconductor.org/packages/release/bioc/vignettes/clusterProfiler/inst/doc/clusterProfiler.html

library("clusterProfiler")

library("enrichplot")

packageVersion("clusterProfiler")

packageVersion("enrichplot")

zfin_id_symbol_list<-c()

zfin_entrez_list<-c()

zebra_entrez<-function(gene_list){

  gene_list<-gsub(".2","",gene_list,fixed=TRUE)

  gene_list<-gsub(".1","",gene_list,fixed=TRUE)
}

```

```

zfin_id_symbol_list<- getBM(attributes=c('ensembl_gene_id',"zfin_id_symbol","entrezgene_id"),
  filters= 'ensembl_gene_id',
  values= gene_list,
  mart = fugumart)

zfin_enrez_list<-getBM(attributes=c('zfin_id_symbol',"entrezgene_id"),
  filters= 'zfin_id_symbol',
  values= zfin_id_symbol_list$zfin_id_symbol,
  mart = zebramart)

return(zfin_enrez_list$entrezgene_id)
}


```

```

gl_rd_entrez<-function(gene_list){

  gene_list<-gsub(".2","",gene_list,fixed=TRUE)
  gene_list<-gsub(".1","",gene_list,fixed=TRUE)

  entrez_list<- getBM(attributes=c('ensembl_gene_id',"entrezgene_id"),
    filters= 'ensembl_gene_id',
    values= gene_list,
    mart = fugumart)

  returnList<-entrez_list[!duplicated(entrez_list$ensembl_gene_id),]

  returnList<-returnList[!duplicated(returnList$entrezgene_id),]

  return(returnList)
}


```

```

de_summary<-function(gene_list){

  gene_list<-gsub(".2","",gene_list,fixed=TRUE)

```

```

gene_list<-gsub(".1","",gene_list,fixed=TRUE)

entrez_list<- getBM(attributes=c('ensembl_gene_id',"zfin_id_symbol","hgnc_symbol","description"),
filters= 'ensembl_gene_id',
values= gene_list,
mart = fugumart)

#returnList<-entrez_list[!duplicated(entrez_list$ensembl_gene_id),]

#returnList<-returnList[!duplicated(returnList$entrezgene_id),]

return(entrez_list)

}

```

```

DEgenes<-rbind(cbind(row.names(head(PP.vs.FM,n=1)),"PP-FM"),
cbind(row.names(head(BAC.vs.FM,n=582)),rep("BAC-FM",582)),
cbind(row.names(head(LY.vs.FM,n=8)),rep("LY-FM",8)),
cbind(row.names(head(HY.vs.FM,n=17)),rep("HY-FM",17)))

DEgenes<-data.frame(DEgenes)

DEgenes[,1]<-gsub(".2","",DEgenes[,1],fixed=TRUE)

DEgenes[,1]<-gsub(".1","",DEgenes[,1],fixed=TRUE)

de_sumarry_result<-de_summary(DEgenes[,1])

head(de_sumarry_result)

```

```

logfc_adjP<-rbind(
head(PP.vs.FM,n=1)[,c(1,5)],
head(BAC.vs.FM,n=582)[,c(1,5)],
head(LY.vs.FM,n=8)[,c(1,5)],
head(HY.vs.FM,n=17)[,c(1,5)])

```

```
mdelist<-c()
```

```

for (i in 1:length(DEgenes[,1])){
  mde<-
  cbind(DEgenes[i,2],round(logfc_adjP[i,1:2],3),de_summary_result[de_summary_result$ensembl_gene_id==DEgenes[i,1],])
  mdelist<-rbind(mdelist,mde)
}
head(mdelist)

```

```
write.csv(mdelist,"degenes_2109017.csv")
```

```

de_gs<-function(gene_list){

  gene_list<-gsub(".2","",gene_list,fixed=TRUE)

  gene_list<-gsub(".1","",gene_list,fixed=TRUE)

  entrez_list<-
  getBM(attributes=c('ensembl_gene_id',"zfin_id_symbol","hgnc_symbol","description","chromosome_name","start_position","end_position"),
  filters= 'ensembl_gene_id',
  values= gene_list,
  mart = fugumart)

  #returnList<-entrez_list[!duplicated(entrez_list$ensembl_gene_id),]

  #returnList<-returnList[!duplicated(returnList$entrezgene_id),]

  return(entrez_list)
}


```

```
gsDEgenes<-cbind(row.names(head(HY.vs.FM,n=17)),rep("HY-FM",17))
```

```

gsDEgenes<-data.frame(gsDEgenes)

gsDEgenes[,1]<-gsub(".2","",gsDEgenes[,1],fixed=TRUE)

gsDEgenes[,1]<-gsub(".1","",gsDEgenes[,1],fixed=TRUE)

gsde_sumarry_result<-de_gs(gsDEgenes[,1])

head(gsde_sumarry_result)

gsdelist<-c()

gsdelist<-cbind(gsDEgenes[,2],round(logfc_adjP[c(-1:-591),1:2],3),gsde_sumarry_result)

head(gsdelist)

write.csv(mdelist,"degenes_210902.csv")

#PP.vs.FM <- topTreat(efit, coef=1, n=Inf)

#BAC.vs.FM <- topTreat(efit, coef=2, n=Inf)

#LY.vs.FM <- topTreat(efit, coef=3, n=Inf)

#HY.vs.FM <- topTreat(efit, coef=4, n=Inf)

summary(def)

# https://www.biostars.org/p/220465/

# BgRatio, M/N.

# M = size of the geneset (eg size of the E2F_targets); (is the number of genes within that distribution that are annotated (either directly or indirectly) to the node of interest).

# N = size of all of the unique genes in the collection of genesets (example the HALLMARK collection); (is the total number of genes in the background distribution (universe))

# GeneRatio is k/n.

# k = size of the overlap of 'a vector of gene id' you input with the specific geneset (eg E2F_targets), only unique genes; (the number of genes within that list n, which are annotated to the node).

```

```
# n = size of the overlap of 'a vector of gene id' you input with all the members of the collection of genesets (eg  
the HALLMARK collection),only unique genes; is the size of the list of genes of interest
```

```
txt_size<-12
```

```
genelist_DE<-list(PP.vs.FM[1,],BAC.vs.FM[1:582,],LY.vs.FM[1:8,],HY.vs.FM[1:17,])
```

```
genelist_all<-list(PP.vs.FM,BAC.vs.FM,LY.vs.FM,HY.vs.FM)
```

```
genelistGSEA<-function(test_gene_list, entrez_list){
```

```
rownames(test_gene_list)<-gsub(".2","",rownames(test_gene_list),fixed=TRUE)
```

```
rownames(test_gene_list)<-gsub(".1","",rownames(test_gene_list),fixed=TRUE)
```

```
geneList<-2^test_gene_list[entrez_list$ensembl_gene_id,]$logFC
```

```
names(geneList) <- as.character(entrez_list$entrezgene_id)
```

```
geneList <- sort(geneList, decreasing = TRUE)
```

```
return(geneList)
```

```
}
```

```
Erez_ora<-function(contrast,method){
```

```
test_gene_list<-as.data.frame(genelist_DE[contrast])
```

```
gene_list<-row.names(test_gene_list)
```

```
entrez_list<-gl_rd_entrez(gene_list)
```

```
set.seed(100)
```

```
if(method == "GO"){
```

```
re <- enricher(entrez_list$entrezgene_id, TERM2GENE=biomart_t2g)
```

```
}
```

```
if(method == "KEGG"){
```

```

re<-enrichKEGG(entrez_list$entrezgene_id,organism ="tru")
}

re@result$p.adjust<-round(re@result$p.adjust,3)

re@result$pvalue<-round(re@result$pvalue,3)

re@result$qvalue<-round(re@result$qvalue,3)

return(re)
}

```

```

Erez_gsea<-function(contrast,method){

test_gene_list<-as.data.frame(genelist_all[contrast])

gene_list<-row.names(test_gene_list)

entrez_list<-gl_rd_entrez(gene_list)

geneList<-genelistGSEA(test_gene_list, entrez_list)

set.seed(100)

if(method == "GO"){

re<-GSEA(geneList, TERM2GENE = biomart_t2g, scoreType = 'pos', nPermSimple = 100000, eps=0)

}

if(method == "KEGG"){

re <- gseKEGG(geneList    = geneList,
              organism   = 'tru',
              scoreType  = "pos",
              pvalueCutoff = 0.05,
              verbose    = FALSE,
              eps = 0)

}

```

```

re@result$p.adjust<-round(re@result$p.adjust,3)

re@result$pvalue <-round(re@result$pvalue,3)

re@result$qvalue <-round(re@result$qvalue,3)

return(re)
}

barplot_gg<-function(re){

  p<-barplot(re, showCategory=100)+

  # scale_fill_manual(values=c("red", "blue","limegreen","purple","yellow"),name="Diet") +

  theme(axis.text.y= element_text(size = txt_size),
        axis.text.x = element_text(size = txt_size),
        text=element_text(size=txt_size, family="serif"))

  return(p)
}

padjustment<-function(p1){

  p1$theme$line$size<-line_th_unit

  p1$theme$line$size<-line_th_unit

  p1$theme$axis.ticks$size<-line_th_unit

  p1$theme$axis.line$size<-line_th_unit

  #p1$theme$panel.grid$colour<-"black"

  p1$theme$axis.ticks$colour<-"black"

  return(p1)
}

library("DOSE")

```

```

barplot_gg_gsea<-function(re){

  p<-ggplot(re, showCategory=100, aes(setSize, fct_reorder(Description, setSize), fill=p.adjust)) +
    geom_col() +
    scale_fill_gradientn(colours=c("#b3eebe", "#46bac2", "#371ea3"),
    guide=guide_colorbar(reverse=TRUE)) + theme_dose(txt_size)+ xlab("Gene count")+ ylab(NULL) +
    # + ggtitle("GO-GSEA")

  # scale_fill_manual(values=c("red", "blue","limegreen","purple","yellow"),name="Diet") +
  theme(
    axis.text.y= element_text(size = txt_size, family="Times New Roman"),
    axis.text.x = element_text(size = txt_size, family="Times New Roman"),
    text=element_text(size=txt_size, family="Times New Roman"))

  p<-padjustment(p)

  return(p)
}

```

```

barplot_gg_ora<-function(re){

  p<-ggplot(re, showCategory=100, aes(Count, fct_reorder(Description, Count), fill=p.adjust)) +
    geom_col() +
    scale_fill_gradientn(colours=c("#b3eebe", "#46bac2", "#371ea3"),
    guide=guide_colorbar(reverse=TRUE)) + theme_dose(txt_size)+ xlab("Gene count")+ ylab(NULL) +
    # + ggtitle("GO-GSEA")

  # scale_fill_manual(values=c("red", "blue","limegreen","purple","yellow"),name="Diet") +
  theme(
    axis.text.y= element_text(size = txt_size, family="Times New Roman"),
    axis.text.x = element_text(size = txt_size, family="Times New Roman"),
    text=element_text(size=txt_size, family="Times New Roman"))

  p<-padjustment(p)
}

```

```

return(p)

}

go_ora<-list()
p_go_ora<-list()
kegg_ora<-list()
p_kegg_ora<-list()

for (i in seq(4)){
  #go_ora[[i]]<-Erez_ora(i,"GO")
  #p_go_ora[[i]]<-barplot_gg(go_ora[[i]])
  kegg_ora[[i]]<-Erez_ora(i,"KEGG")
  p_kegg_ora[[i]]<-barplot_gg_ora(kegg_ora[[i]])
}

go_gsea<-list()
p_go_gsea<-list()
kegg_gsea<-list()
p_kegg_gsea<-list()

for (i in seq(4)){
  #go_gsea[[i]]<-Erez_gsea(i,"GO")
  #p_go_gsea[[i]]<-dotplot_gg(go_gsea[[i]])
  kegg_gsea[[i]]<-Erez_gsea(i,"KEGG")
  #p_kegg_gsea[[i]]<-dotplot_gg(kegg_gsea[[i]])
}

```

```

for (i in seq(4)){
  #go_gsea[[i]]<-Erez_gsea(i,"GO")
  #p_go_gsea[[i]]<-barplot_gg_gsea(go_gsea[[i]])
  #kegg_gsea[[i]]<-Erez_gsea(i,"KEGG")
  p_kegg_gsea[[i]]<-barplot_gg_gsea(kegg_gsea[[i]])
}

library("cowplot")
options(repr.plot.width=22, repr.plot.height=16)
all<- ggdraw() +
  draw_plot(p_go_gsea[[1]], x = 0, y = 0.5, width = 0.50, height = 0.5) +
  draw_plot(p_go_gsea[[2]], x = 0.5, y = 0.50, width = 0.5, height = 0.50) +
  draw_plot(p_go_gsea[[3]], x = 0, y = 0, width = 0.4, height = 0.50) +
  draw_plot(p_go_gsea[[4]], x = 0.4, y = 0, width = 0.6, height = 0.50) +
  draw_plot_label(label = c("a. PP.vs.FM", "b. BAC.vs.FM", "c. LY.vs.FM", "d. HY.vs.FM"), size = txt_size, family = "serif", x = c(0., 0.5, 0, 0.4), y = c(1, 1, 0.50, 0.5))
all
# ggsave("kegg_ora.pdf", device = "pdf", units = "in", font = "Helvetica", width = 16, height = 12)
ggsave("go_gsea.png", device = "png", dpi = 300, units = "in", font = "Helvetica", width = 22, height = 16)

library("cowplot")
options(repr.plot.width = 12, repr.plot.height = 6.6)
all2<- ggdraw() +
  draw_plot(p_kegg_ora[[1]], x = 0, y = .7, width = 0.4, height = 0.3) +
  draw_plot(p_kegg_ora[[2]], x = 0.4, y = 0, width = 0.60, height = 1) +

```

```

draw_plot(p_kegg_ora[[3]], x = 0, y = 0.35, width = 0.4, height = 0.35) +
draw_plot(p_kegg_ora[[4]], x = 0, y = 0, width = 0.4, height = 0.35) +
draw_plot_label(label = c("a", "b", "c", "d"), size = 14, family = "Times New Roman", fontface = "plain", x = c(0, 0.4, 0, 0), y = c(1, 1, 0.7, 0.35))

all2

ggsave("kegg_ora.pdf", device = "pdf", units = "in", width = 12, height = 6)
#ggsave("kegg_ora.png", device = "png", dpi = 300, units = "in", font = "Helvetica", width = 15, height = 10)

```

```

library("cowplot")

options(repr.plot.width = 12, repr.plot.height = 8)

all2 <- ggdraw() +
draw_plot(p_kegg_gsea[[1]], x = 0, y = .725, width = 0.4, height = 0.275) +
draw_plot(p_kegg_gsea[[4]], x = 0.4, y = 0, width = 0.6, height = 1) +
draw_plot(p_kegg_gsea[[2]], x = 0, y = 0.275, width = 0.4, height = 0.45) +
draw_plot(p_kegg_gsea[[3]], x = 0, y = 0, width = 0.4, height = 0.275) +
draw_plot_label(label = c("a", "d", "b", "c"), size = 14, family = "Times New Roman", fontface = "plain", x = c(0, 0.4, 0, 0), y = c(1, 1, 0.725, 0.300))

all2

ggsave("kegg_gsea.pdf", device = "pdf", units = "in", width = 12, height = 8)
#ggsave("kegg_ora.png", device = "png", dpi = 300, units = "in", font = "Helvetica", width = 15, height = 10)

```

```

library("cowplot")

options(repr.plot.width = 12, repr.plot.height = 6)

all <- ggdraw() +
draw_plot(p_kegg_gsea[[1]], x = 0, y = 0.5, width = 0.5, height = 0.5) +
draw_plot(p_kegg_gsea[[2]], x = 0.5, y = 0.5, width = 0.5, height = 0.5) +
draw_plot(p_kegg_gsea[[3]], x = 0, y = 0, width = 0.4, height = 0.5) +

```

```

draw_plot(p_kegg_gsea[[4]], x = 0.4, y = 0, width = 0.6, height = 0.50) +
  draw_plot_label(label = c("a. PP.vs.FM", "b. BAC.vs.FM", "c. LY.vs.FM", "d. HY.vs.FM"), size = txt_size, family = "serif", x = c(0., 0.5, 0, 0.4), y = c(1, 1, 0.50, 0.5))

all

# ggsave("kegg_ora.pdf", device = "pdf", units = "in", font = "Helvetica", width = 16, height = 12)
ggsave("kegg_gsea.png", device = "png", dpi = 300, units = "in", font = "Helvetica", width = 20, height = 20)

contrast_index <- c("PP-FM", "BAC-FM", "LY-FM", "HY-FM")

ora_tframe <- c()

result_ora <- function(ora) {
  for (i in c(1, 2, 3, 4)) {
    ora_dframe <- c()
    ora_dframe <- ora[[i]] @ result[ora[[i]] @ result$p.adjust < 0.05,]
    count <- dim(ora_dframe)[1]
    ora_dframe <- cbind(rep(contrast_index[i], count), ora_dframe)
    ora_tframe <- rbind(ora_tframe, ora_dframe)
    row.names(ora_tframe) <- c()
    # go_ora[[2]] @ result[go_ora[[2]] @ result$p.adjust < 0.05, c(-1, -2)]
  }
  colnames(ora_tframe)[1] <- c("Contrast")
  return(ora_tframe)
}

#re_go_ora <- result_ora(go_ora)
re_kegg_ora <- result_ora(kegg_ora)
#re_go_gsea <- result_ora(go_gsea)

```

```

re_kegg_gsea<-result_ora(kegg_gsea)

library("xlsx")

#write.xlsx(re_go_ora,
#           file = './ora_gsea.xlsx',
#           sheetName = 'go_ora',append=TRUE)

write.xlsx(re_kegg_ora,
           file = './ora_gsea_211020.xlsx',
           sheetName = 'kegg_ora',append=TRUE)

#write.xlsx(re_go_gsea,
#           file = './ora_gsea.xlsx',
#           sheetName = 'go_gsea',append=TRUE)

write.xlsx(re_kegg_gsea,
           file = './ora_gsea_211020.xlsx',
           sheetName = 'kegg_gsea',append=TRUE)

length(unlist(strsplit(fugukegg@result$geneID,split = "/")))

length(unique(unlist(strsplit(fugukegg@result$geneID,split = "/"))))

sum(fugukegg@result$Count)

library("cowplot")

options(repr.plot.width=16, repr.plot.height=10)

all<- ggdraw() +
  draw_plot(p_go_ora[[1]], x = 0, y = 0, width = 0.60, height = 1) +

```

```

draw_plot(p_go_ora[[2]], x = 0.60, y = 0.50, width = 0.4, height = 0.50) +
draw_plot(p_go_ora[[3]], x = 0, y = 0, width = 0.4, height = 0.50) +
draw_plot(p_go_ora[[4]], x = 0, y = 0, width = 0.4, height = 0.50) +
draw_plot_label(label = c("a. BAC.vs.FM", "b. LY.vs.FM", "c. HY.vs.FM"), size = txt_size, family = "serif", x = c(0., 0.6, 0.6), y = c(1, 1, 0.50))

all

# ggsave("kegg_ora.pdf", device = "pdf", units = "in", font = "Helvetica", width = 16, height = 12)
ggsave("kegg_ora.png", device = "png", dpi = 300, units = "in", font = "Helvetica", width = 15, height = 10)

sum(fugukegg@result$Count)

```

fugukegg

```

library("cowplot")

options(repr.plot.width = 15, repr.plot.height = 10)

all <- ggdraw() +
draw_plot(P0, x = 0, y = .7, width = 0.4, height = 0.3) +
draw_plot(P1, x = 0, y = 0, width = 0.6, height = 0.7) +
draw_plot(P2, x = 0.6, y = 0.5, width = 0.4, height = 0.5) +
draw_plot(P3, x = 0.6, y = 0, width = 0.4, height = 0.5) +
draw_plot_label(label = c("a. PP.vs.FM", "b. BAC.vs.FM", "c. LY.vs.FM", "d. HY.vs.FM"), size =
= txt_size, family = "serif", x = c(0, 0, 0.6, 0.6), y = c(1, 0.7, 1, 0.5))

all

# ggsave("kegg_ora.pdf", device = "pdf", units = "in", font = "Helvetica", width = 16, height = 12)
# ggsave("kegg_ora.png", device = "png", dpi = 300, units = "in", font = "Helvetica", width = 15, height = 10)

```

```

library("cowplot")

options(repr.plot.width=20, repr.plot.height=15)

all<- ggdraw() +

  draw_plot(P0, x = 0, y = .7, width = 0.4, height = 0.3) +
  draw_plot(P1, x = 0.4, y = 0, width = 0.60, height = 1) +
  draw_plot(P2, x = 0, y = 0.35, width = 0.4, height = 0.35) +
  draw_plot(P3, x = 0, y = 0, width = 0.4, height = 0.35) +
  draw_plot_label(label = c("a. PP.vs.FM", "b. BAC.vs.FM", "c. LY.vs.FM", "d. HY.vs.FM"), size
  =txt_size,family="serif", x = c(0, 0.4,0, 0), y = c(1, 1, 0.7, 0.35))

all

# ggsave("kegg_ora.pdf",device="pdf",units="in",font="Helvetica",width=16, height=12)

#ggsave("kegg_ora.png",device="png",dpi = 300,units="in",font="Helvetica",width=15, height=10)

```

```
help(enricher)
```

```

gene_list<-row.names(PP.vs.FM[1,])

entrez_list<-gl_rd_entrez(gene_list)

em <- enricher(entrez_list$entrezgene_id, TERM2GENE=biomart_t2g)

em

```

```

gene_list<-row.names(PP.vs.FM[1,])

entrez_list<-gl_rd_entrez(gene_list)

em <- enricher(entrez_list$entrezgene_id, TERM2GENE=biomart_t2g)

p0<-dotplot(em,showCategory=20)+

  # scale_fill_manual(values=c("red", "blue", "limegreen", "purple", "yellow"),name="Diet") +
  theme(axis.text.y= element_text(size = txt_size),
        axis.text.x = element_text(size = txt_size)),

```

```

text=element_text(size=txt_size, family="serif"))

gene_list<-row.names(BAC.vs.FM[1:582,])

entrez_list<-gl_rd_entrez(gene_list)

em <- enricher(entrez_list$entrezgene_id, TERM2GENE=biomart_t2g)

p1<-dotplot(em,showCategory=20)+

# scale_fill_manual(values=c("red", "blue", "limegreen", "purple", "yellow"),name="Diet") +

theme(axis.text.y= element_text(size = txt_size),

      axis.text.x = element_text(size = txt_size),

      text=element_text(size=txt_size, family="serif"))

gene_list<-row.names(LY.vs.FM[1:8,])

entrez_list<-gl_rd_entrez(gene_list)

em <- enricher(entrez_list$entrezgene_id, TERM2GENE=biomart_t2g)

p2<-dotplot(em,showCategory=20)+

# scale_fill_manual(values=c("red", "blue", "limegreen", "purple", "yellow"),name="Diet") +

theme(axis.text.y= element_text(size = txt_size),

      axis.text.x = element_text(size = txt_size),

      text=element_text(size=txt_size, family="serif"))

gene_list<-row.names(HY.vs.FM[1:17,])

entrez_list<-gl_rd_entrez(gene_list)

em <- enricher(entrez_list$entrezgene_id, TERM2GENE=biomart_t2g)

p3<-dotplot(em,showCategory=20)+

# scale_fill_manual(values=c("red", "blue", "limegreen", "purple", "yellow"),name="Diet") +

theme(axis.text.y= element_text(size = txt_size),

      axis.text.x = element_text(size = txt_size),

      text=element_text(size=txt_size, family="serif"))

library("cowplot")

```

```

options(repr.plot.width=16, repr.plot.height=10)

all2<- ggdraw() +

  draw_plot(p1, x = 0, y = 0, width = 0.60, height = 1) +

  draw_plot(p2, x = 0.60, y = 0.50, width = 0.4, height = 0.50) +

  draw_plot(p3, x = 0.60, y = 0, width = 0.4, height = 0.50)+

  draw_plot_label(label = c("a. BAC.vs.FM", "b. LY.vs.FM", "c. HY.vs.FM"), size =txt_size,family="serif", x =
c(0., 0.6, 0.6), y = c(1, 1, 0.50))

all2

# ggsave("kegg_ora.pdf",device="pdf",units="in",font="Helvetica",width=16, height=12)

ggsave("go_ora.png",device="png",dpi = 300,units="in",font="Helvetica",width=15, height=10)

library("cowplot")

options(repr.plot.width=20, repr.plot.height=15)

all2<- ggdraw() +

  draw_plot(p0, x = 0, y = .7, width = 0.4, height = 0.3) +

  draw_plot(p1, x = 0.4, y = 0, width = 0.60, height = 1) +

  draw_plot(p2, x = 0, y = 0.35, width = 0.4, height = 0.35) +

  draw_plot(p3, x = 0, y = 0, width = 0.4, height = 0.35)+

  draw_plot_label(label = c("a. PP.vs.FM","b. BAC.vs.FM", "c. LY.vs.FM", "d. HY.vs.FM"), size
=txt_size,family="serif", x = c(0, 0.4,0, 0), y = c(1, 1, 0.7, 0.35))

all2

# ggsave("kegg_ora.pdf",device="pdf",units="in",font="Helvetica",width=16, height=12)

#ggsave("kegg_ora.png",device="png",dpi = 300,units="in",font="Helvetica",width=15, height=10)

test_gene_list<-PP.vs.FM

```

```

gene_list<-row.names(test_gene_list)

entrez_list<-gl_rd_entrez(gene_list)

rownames(test_gene_list)<-gsub(".2","",rownames(test_gene_list),fixed=TRUE)

rownames(test_gene_list)<-gsub(".1","",rownames(test_gene_list),fixed=TRUE)

geneList<-2^test_gene_list[entrez_list$ensembl_gene_id,]$logFC

names(geneList) <- as.character(entrez_list$entrezgene_id)

geneList <- sort(geneList, decreasing = TRUE)

kk1 <- gseKEGG(geneList = geneList,
                organism = 'tru',
                scoreType = "pos",
                pvalueCutoff = 0.05,
                verbose = FALSE, eps = 0)

p1 <- dotplot(kk1, showCategory=20) +
  # scale_fill_manual(values=c("red", "blue", "limegreen", "purple", "yellow"),name="Diet") +
  theme(axis.text.y= element_text(size = txt_size),
        axis.text.x = element_text(size = txt_size),
        text=element_text(size=txt_size, family="serif"))

test_gene_list<-BAC.vs.FM

gene_list<-row.names(test_gene_list)

entrez_list<-gl_rd_entrez(gene_list)

rownames(test_gene_list)<-gsub(".2","",rownames(test_gene_list),fixed=TRUE)

rownames(test_gene_list)<-gsub(".1","",rownames(test_gene_list),fixed=TRUE)

geneList<-2^test_gene_list[entrez_list$ensembl_gene_id,]$logFC

names(geneList) <- as.character(entrez_list$entrezgene_id)

geneList <- sort(geneList, decreasing = TRUE)

kk2 <- gseKEGG(geneList = geneList,

```

```

organism    = 'tru',
scoreType   = "pos",
pvalueCutoff = 0.05,
verbose     = FALSE, eps = 0)

p2 <- dotplot(kk2, showCategory=20) +
  theme(axis.text.y= element_text(size = txt_size),
        axis.text.x = element_text(size = txt_size),
        text=element_text(size=txt_size, family="serif"))

test_gene_list<-LY.vs.FM

gene_list<-row.names(test_gene_list)

entrez_list<-gl_rd_entrez(gene_list)

rownames(test_gene_list)<-gsub(".2","",rownames(test_gene_list),fixed=TRUE)

rownames(test_gene_list)<-gsub(".1","",rownames(test_gene_list),fixed=TRUE)

geneList<-2^test_gene_list[entrez_list$ensembl_gene_id,]$logFC

names(geneList) <- as.character(entrez_list$entrezgene_id)

geneList <- sort(geneList, decreasing = TRUE)

kk3 <- gseKEGG(geneList    = geneList,
                organism    = 'tru',
                scoreType   = "pos",
                pvalueCutoff = 0.05,
                verbose     = FALSE, eps = 0)

p3<- dotplot(kk3, showCategory=20) +
  theme(axis.text.y= element_text(size = txt_size),
        axis.text.x = element_text(size = txt_size),
        text=element_text(size=txt_size, family="serif"))

```

p3

```

test_gene_list<-HY.vs.FM

gene_list<-row.names(test_gene_list)

entrez_list<-gl_rd_entrez(gene_list)

rownames(test_gene_list)<-gsub(".2","",rownames(test_gene_list),fixed=TRUE)

rownames(test_gene_list)<-gsub(".1","",rownames(test_gene_list),fixed=TRUE)

geneList<-2^test_gene_list[entrez_list$ensembl_gene_id,]$logFC

names(geneList) <- as.character(entrez_list$entrezgene_id)

geneList <- sort(geneList, decreasing = TRUE)

kk4 <- gseKEGG(geneList    = geneList,
                organism   = 'tru',
                scoreType  = "pos",
                pvalueCutoff = 0.05,
                verbose    = FALSE, eps = 0)

p4<- dotplot(kk4, showCategory=20)+

theme(axis.text.y= element_text(size = txt_size),
      axis.text.x = element_text(size = txt_size),
      text=element_text(size=txt_size, family="serif"))

library("cowplot")

options(repr.plot.width=15, repr.plot.height=10)

all3<- ggdraw() +
  draw_plot(p1, x = 0, y = 0.5, width = 0.50, height = 0.5) +
  draw_plot(p2, x = 0.50, y = 0.50, width = 0.5, height = 0.50) +
  draw_plot(p3, x = 0, y = 0, width = 0.4, height = 0.50) +
  draw_plot(p4, x = 0.40, y = 0, width = 0.6, height = 0.50) +

```

```

draw_plot_label(label = c("a. PP.vs.FM", "b. BAC.vs.FM", "c. LY.vs.FM", "d. HY.vs.FM"), size
=txt_size,family="serif", x = c(0., 0.5,0, 0.4), y = c(1, 1, 0.5,0.5))

all3

# ggsave("kegg_ora.pdf",device="pdf",units="in",font="Helvetica",width=16, height=12)

#ggsave("kegg_gesa.png",device="png",dpi = 300,units="in",font="Helvetica",width=15, height=13)

emap1<-pairwise_termsim(kk1)

emap2<-pairwise_termsim(kk2)

emap3<-pairwise_termsim(kk3)

emap4<-pairwise_termsim(kk4)

p1 <- emapplot(emap1)+

theme(text=element_text(size=txt_size, family="serif"),

panel.border = element_rect(colour = "black", fill=NA, size=1))

p2 <- emapplot(emap2)+

theme(text=element_text(size=txt_size, family="serif"),

panel.border = element_rect(colour = "black", fill=NA, size=1))

p3 <- emapplot(emap3)+

theme(text=element_text(size=txt_size, family="serif"),

panel.border = element_rect(colour = "black", fill=NA, size=1))

p4 <- emapplot(emap4)+

theme(text=element_text(size=txt_size, family="serif"),

panel.border = element_rect(colour = "black", fill=NA, size=1))

library("cowplot")

options(repr.plot.width=20, repr.plot.height=20)

all4<- ggdraw() +

draw_plot(p1, x = 0, y = .5, width = 0.5, height = 0.5) +

```

```

draw_plot(p2, x = 0.5, y = .5, width = 0.5, height = 0.5) +
draw_plot(p3, x = 0, y = 0, width = 0.5, height = 0.5) +
draw_plot(p4, x = 0.5, y = 0, width = .5, height = .5) +
draw_plot_label(label = c("a. PP.vs.FM", "b. BAC.vs.FM", "c. LY.vs.FM", "d. HY.vs.FM"), size = txt_size, family = "serif", x = c(0., 0.5, 0, .5), y = c(1, 1, 0.5, 0.5))
all4

# ggsave("kegg_ora.pdf", device = "pdf", units = "in", font = "Helvetica", width = 16, height = 12)
ggsave("emapp_gesa.png", device = "png", dpi = 300, units = "in", font = "Helvetica", width = 20, height = 20)

test_gene_list<-PP.vs.FM
gene_list<-row.names(test_gene_list)
entrez_list<-gl_rd_entrez(gene_list)
rownames(test_gene_list)<-gsub(".2","",rownames(test_gene_list),fixed=TRUE)
rownames(test_gene_list)<-gsub(".1","",rownames(test_gene_list),fixed=TRUE)
geneList<-2^test_gene_list[entrez_list$ensembl_gene_id,]$logFC
names(geneList)<- as.character(entrez_list$entrezgene_id)
geneList <- sort(geneList, decreasing = TRUE)
em1 <- GSEA(geneList, TERM2GENE = biomart_t2g, scoreType = 'pos', nPermSimple = 10000, eps = 0)
p1 <- dotplot(em1, showCategory = 30) +
theme(axis.text.y = element_text(size = txt_size),
axis.text.x = element_text(size = txt_size),
text = element_text(size = txt_size, family = "serif"))

test_gene_list<-BAC.vs.FM
gene_list<-row.names(test_gene_list)
entrez_list<-gl_rd_entrez(gene_list)
rownames(test_gene_list)<-gsub(".2","",rownames(test_gene_list),fixed=TRUE)
rownames(test_gene_list)<-gsub(".1","",rownames(test_gene_list),fixed=TRUE)

```

```

geneList<-2^test_gene_list[entrez_list$ensembl_gene_id,]$logFC

names(geneList) <- as.character(entrez_list$entrezgene_id)

geneList <- sort(geneList, decreasing = TRUE)

em2 <- GSEA(geneList, TERM2GENE = biomart_t2g, scoreType = 'pos', nPermSimple = 10000, eps=0)

p2 <- dotplot(em2, showCategory=30) +

  theme(axis.text.y= element_text(size = txt_size),

        axis.text.x = element_text(size = txt_size),

        text=element_text(size=txt_size, family="serif"))

test_gene_list<-LY.vs.FM

gene_list<-row.names(test_gene_list)

entrez_list<-gl_rd_entrez(gene_list)

rownames(test_gene_list)<-gsub(".2","",rownames(test_gene_list),fixed=TRUE)

rownames(test_gene_list)<-gsub(".1","",rownames(test_gene_list),fixed=TRUE)

geneList<-2^test_gene_list[entrez_list$ensembl_gene_id,]$logFC

names(geneList) <- as.character(entrez_list$entrezgene_id)

geneList <- sort(geneList, decreasing = TRUE)

em3 <- GSEA(geneList, TERM2GENE = biomart_t2g, scoreType = 'pos', nPermSimple = 10000, eps=0)

p3 <- dotplot(em3, showCategory=30) +

  theme(axis.text.y= element_text(size = txt_size),

        axis.text.x = element_text(size = txt_size),

        text=element_text(size=txt_size, family="serif"))

test_gene_list<-HY.vs.FM

gene_list<-row.names(test_gene_list)

entrez_list<-gl_rd_entrez(gene_list)

rownames(test_gene_list)<-gsub(".2","",rownames(test_gene_list),fixed=TRUE)

rownames(test_gene_list)<-gsub(".1","",rownames(test_gene_list),fixed=TRUE)

geneList<-2^test_gene_list[entrez_list$ensembl_gene_id,]$logFC

```

```

names(geneList) <- as.character(entrez_list$entrezgene_id)

geneList <- sort(geneList, decreasing = TRUE)

em4 <- GSEA(geneList, TERM2GENE = biomart_t2g, scoreType = 'pos', nPermSimple = 10000, eps=0)

p4 <- dotplot(em4, showCategory=30) +
  theme(axis.text.y = element_text(size = txt_size),
        axis.text.x = element_text(size = txt_size),
        text=element_text(size=txt_size, family="serif"))

library("cowplot")

options(repr.plot.width=20, repr.plot.height=10)

all5<- ggdraw() +
  draw_plot(p1, x = 0, y = 0.5, width = 0.50, height = 0.5) +
  draw_plot(p2, x = 0.50, y = 0.50, width = 0.5, height = 0.50) +
  draw_plot(p3, x = 0, y = 0, width = 0.4, height = 0.50) +
  draw_plot(p4, x = 0.40, y = 0, width = 0.6, height = 0.50) +
  draw_plot_label(label = c("a. PP.vs.FM", "b. BAC.vs.FM", "c. LY.vs.FM", "d. HY.vs.FM"), size = txt_size, family="serif", x = c(0., 0.5, 0, 0.4), y = c(1, 1, 0.5, 0.5))

all5

# ggsave("kegg_ora.pdf",device="pdf",units="in",font="Helvetica",width=16, height=12)
ggsave("go_gesa.png",device="png",dpi = 300,units="in",font="Helvetica",width=20, height=15)

emap1<-pairwise_termsim(em1)

emap2<-pairwise_termsim(em2)

emap3<-pairwise_termsim(em3)

emap4<-pairwise_termsim(em4)

p1 <- emapplot(emap1) +
  theme(text=element_text(size=txt_size, family="serif"),

```

```

panel.border = element_rect(colour = "black", fill=NA, size=1))

p2 <- emapplot(emap2) +
  theme(text=element_text(size=txt_size, family="serif"),
        panel.border = element_rect(colour = "black", fill=NA, size=1))

p3 <- emapplot(emap3) +
  theme(text=element_text(size=txt_size, family="serif"),
        panel.border = element_rect(colour = "black", fill=NA, size=1))

p4 <- emapplot(emap4) +
  theme(text=element_text(size=txt_size, family="serif"),
        panel.border = element_rect(colour = "black", fill=NA, size=1))

library("cowplot")

options(repr.plot.width=15, repr.plot.height=15)

all6<- ggdraw() +
  draw_plot(p1, x = 0, y = 0.5, width = 0.50, height = 0.5) +
  draw_plot(p2, x = 0.50, y = 0.50, width = 0.5, height = 0.50) +
  draw_plot(p3, x = 0, y = 0, width = 0.5, height = 0.50) +
  draw_plot(p4, x = 0.50, y = 0, width = 0.5, height = 0.50) +
  draw_plot_label(label = c("a. PP.vs.FM","b. BAC.vs.FM", "c. LY.vs.FM", "d. HY.vs.FM"), size
=txt_size,family="serif", x = c(0., 0.5,0, 0.5), y = c(1, 1, 0.5,0.5))

all6

# ggsave("kegg_ora.pdf",device="pdf",units="in",font="Helvetica",width=16, height=12)

ggsave("emapp_go.png",device="png",dpi = 300,units="in",font="Helvetica",width=15, height=15)

options(repr.plot.width = 8, repr.plot.height = 8)

p2 <- dotplot(em3, showCategory=30) + ggtitle("dotplot for GSEA")

p2

```

```

emap<-pairwise_termsim(em3)

p1 <- emapplot(emap)

p1

library("gplots")

gene_names<-c("hmgera", "msmol1", "LOC101080196", "LOC101079966", "LOC101066347",
"LOC101077910", "PLA2G1B", "None", "LOC101080080", "AMDHD1","aspdh", "syt15")

heat_data<-data[data$sample_id != 125,]

heat_data["id"]<-1:399

reorder_ix<-(heat_data %>% arrange(diet,sample_id))$id

heatmap_data<-rbind(diet1.vs.diet4[1:5],diet1.vs.diet5[1:7,])

i <- which(row.names(lcpm_fn) %in% row.names(heatmap_data))

mycol <- colorpanel(399,"blue","white","red")

heatmap.2(lcpm_fn[i,reorder_ix], ColSideColors = as.character(heat_data[reorder_ix,]$diet),
key=T, srtRow=45, adjRow=c(0, 1),Rowv = FALSE, Colv=FALSE,scale = "row",
col=mycol,
labRow=(gene_names), labCol=x$samples$merged_id,
trace="none", density.info="none", cexRow=0.7,
dendrogram="none")

heatmap_data<-rbind(diet1.vs.diet4[1:5],diet1.vs.diet5[1:7,])

i <- which(row.names(lcpm_fn) %in% row.names(heatmap_data))

```

```

mycol <- colorpanel(399,"blue","white","red")

heatmap.2(lcpm_fn[i,reorder_ix], ColSideColors = as.character(heat_data[reorder_ix,]$diet),
          key=T, srtRow=45, adjRow=c(0, 1), Rowv = FALSE, scale = "row",
          col=mycol,
          labRow=(gene_names), labCol=x$samples$merged_id,
          trace="none", density.info="none", cexRow=0.7,
          dendrogram="column")

```

```
str(heat_data)
```

```
heat_data$library<-gsub("L1","1",heat_data$library)
```

```
heat_data$library<-gsub("L2","2",heat_data$library)
```

```
heat_data$library<-gsub("L3","3",heat_data$library)
```

```
heatmap_data<-rbind(diet1.vs.diet4[1:5], diet1.vs.diet5[1:7,])
```

```
i <- which(row.names(lcpm_fn) %in% row.names(heatmap_data))
```

```
mycol <- colorpanel(399,"blue","white","red")
```

```
heatmap.2(lcpm_fn[i,reorder_ix], ColSideColors = as.character(heat_data[reorder_ix,]$library),
```

```
key=T, srtRow=45, adjRow=c(0, 1), Rowv = FALSE, scale = "row",
```

```
col=mycol,
```

```
labRow=(gene_names), labCol=x$samples$merged_id,
```

```
trace="none", density.info="none", cexRow=0.7,
```

```
dendrogram="column")
```

```
heatmap_data<-rbind(diet1.vs.diet4[1:5], diet1.vs.diet5[1:7,])
```

```

i<- which(row.names(lcpm_fn) %in% row.names(heatmap_data))

mycol<- colorpanel(399,"blue","white","red")

heatmap.2(lcpm_fn[i,reorder_ix], ColSideColors = as.character(heat_data[reorder_ix]$library),
key=T, srtRow=45, adjRow=c(0, 1),Rowv = FALSE,scale = "row",
col=mycol,
labRow=(gene_names), labCol=x$samples$merged_id,
trace="none", density.info="none", cexRow=0.7,
dendrogram="column")

tank_individuals<-data[data$sample_id != 125,] %>% filter(replicates==1) %>% arrange(tank, sample_id)
head(tank_individuals)

summary(tank_individuals,maxsum=15)

#tank 每のサンプル数

tank_individuals2<-data[data$sample_id != 125,] %>% arrange(tank, sample_id)
head(tank_individuals)

summary(tank_individuals2,maxsum=15)

blood_190515<- read.csv("190515_afterbloodtest_reform.csv")
pheno_190517<- read.csv("190517jst_data_reform.csv")

head(blood_190515)

head(pheno_190517)

```

```

#x$samples$sample_id

blood_result<-c()

for (i in x$samples$sample_id){

blood_result<-
rbind(blood_result,cbind(blood_190515[,c(1,8,9,10,13,15,16,17,18)]%>%filter(sample_id==c(i)),

pheno_190517[,c(5:17)]%>%filter(sample_id0515==c(i))))}

}

blood_result<-blood_result[,c(-1,-12,-15,-18,-19)] 

blood_result["SL0515mSL0214"]<- blood_result$SL0515-blood_result$SL0214

blood_result["BW0515mBW0214"]<- blood_result$BW0515-blood_result$BW0214

str(blood_result)

blood_result %>% group_by

DE_gene_list<-c(row.names(head(diet1.vs.diet4, n=5)),row.names(head(diet1.vs.diet5, n=7)))

DE_g_cor<-c()

for (i in DE_gene_list){

DE_g_cor = rbind(DE_g_cor, cor(d_n$counts[row.names(d_n$counts)==i,],blood_result))

}

row.names(DE_g_cor)<-DE_gene_list

DE_g_cor

```

```

DE_g_cor_dietvs<-c()

colname_list<-c()

DE_gene_list1r<-c(DE_gene_list[1],DE_gene_list)

Diet_vs<-c(2,4,4,4,4,5,5,5,5,5,5)

for (i in 1:length(DE_gene_list1r)) {

  g_name<-DE_gene_list1r[i]

  Diet<-Diet_vs[i]

  colname_list<-c(colname_list,paste(DE_gene_list1r[i],"&diet1.vs.",Diet_vs[i],sep=""))

  DE_g_cor_dietvs = rbind(DE_g_cor_dietvs,

    cor(d_n$counts[row.names(d_n$counts)==g_name,d_n$samples$group==1 |
d_n$samples$group==Diet],

      blood_result[d_n$samples$group==1 | d_n$samples$group==Diet,]))

}

row.names(DE_g_cor_dietvs)<-colname_list


DE_g_cor_dietvs

# Gender may have effect on the result

```

Section 4.1

4.1.1 R script for phenotypes (Pop-C)

```
R.Version()$version.string  
library("ggplot2")  
library("ggpubr")  
library("extrafont")  
  
print(paste("ggplot2 version",packageVersion("ggplot2")))  
print(paste("ggpubr version",packageVersion("ggpubr")))  
print(paste("extrafont version",packageVersion("extrafont")))  
# remotes::install_version("Rttf2pt1", version = "1.3.8")  
# ttf_import()  
# fonts()  
# loadfonts(device = "pdf")  
# extrafont::loadfonts(device = "postscript")  
  
phenoRaw <- read.csv("./200401H31_modified.csv")  
# attach(pheno)  
# n<-length(pheno$BW)  
head(phenoRaw)  
str(phenoRaw)  
  
# Filtration  
# pheno_BCW_cleaned<-pheno[!is.na(pheno$BCW),]  
pheno_BCW_cleaned1<-phenoRaw[!is.na(phenoRaw$BCW),]  
pheno_BCW_cleaned<-pheno_BCW_cleaned1[pheno_BCW_cleaned1$memo!="Mistake",]  
str(pheno_BCW_cleaned)  
  
pheno<-data.frame(SL1=pheno_BCW_cleaned$X190930_SL,  
                    BW1=pheno_BCW_cleaned$X190930_BW,  
                    tank_f=factor(pheno_BCW_cleaned$X190930_tank),  
                    SL2=pheno_BCW_cleaned$X191108SL,  
                    BW2=pheno_BCW_cleaned$X191108BW,  
                    bcw=pheno_BCW_cleaned$BCW,  
                    HD=pheno_BCW_cleaned$BCW/(pheno_BCW_cleaned$X191108SL)^2*100,  
                    egg=factor(pheno_BCW_cleaned$egg),  
                    human = factor(pheno_BCW_cleaned$Conducter),  
                    deltaSL=(pheno_BCW_cleaned$X191108SL-
```

```

pheno_BCW_cleaned$X190930_SL)/pheno_BCW_cleaned$X190930_SL*100,
deltaBW=(pheno_BCW_cleaned$X191108BW-
pheno_BCW_cleaned$X190930_BW)/pheno_BCW_cleaned$X190930_BW*100)
attach(pheno)

str(pheno)

mean_sd_norm_round<-function(v){
  cbind(round(mean(v),3),round(sd(v),3),
round(shapiro.test(v)[2]$p.value,3),round(max(v),3),round(min(v),3))
}

re_sum<-c()
for (i in c(1,2,4,5,6,7,10,11)){
  re_sum<-rbind(re_sum,mean_sd_norm_round(pheno[,i]))
}
row.names(re_sum)<-colnames(pheno)[c(1,2,4,5,6,7,10,11)]
colnames(re_sum)<-c("Ave","sd","p_norm","max","min")
re_sum

# Pearson's r
cor(pheno[,c(1,2,4,5,6,7,10,11)])]

library(caret)
bctbcw<-BoxCoxTrans(bcw+1)
bctbcw
predictbcw<-predict(bctbcw, bcw+1) # ((bcw+1)^0.6-1)/0.6
# round(predictbcw,3)===round(((bcw+1)^0.6-1)/0.6,3)
pheno["predictbcw"]<-predictbcw

cor.test(pheno$predictbcw,pheno$SL1)

binw<-round(((1+1)^0.6-1)/0.6,3)

# Transformed H. okamotoi count
line_th_unit<-0.6 ##size unit, mm
scaleFUN <- function(x) sprintf("%.2f", x)
options(repr.plot.width=8, repr.plot.height=4)
p3<-ggplot(pheno, aes(x=predictbcw))+
```

```

ggpubr::theme_pubr(base_family="Times New Roman",base_size = 12)+  

  geom_histogram(aes(y = ..density..),color="black",fill="grey" ,binwidth = binw,size=line_th_unit)+  

  geom_density(size=line_th_unit)+  

  labs(x="Transformed HC", y="Density")  

p3$theme$line$size<-line_th_unit  

p3$theme$line$size<-line_th_unit  

p3$theme$axis.ticks$size<-line_th_unit  

p3$theme$axis.line$size<-line_th_unit  

p3$theme$panel.grid$colour<-"black"  

p3$theme$axis.ticks$colour<-"black"  

p3  

ggsave("Fig. 4. tbcw.pdf",device="pdf",units="in",width =6, height =3 )

```

```

# H. okamotoi count  

line_th_unit<-0.6 ##size unit, mm  

scaleFUN <- function(x) sprintf("%,.2f", x)  

options(repr.plot.width=8, repr.plot.height=4)  

p1<-ggplot(pheno, aes(x=bew))+  

  ggpibr::theme_pubr(base_family="Times New Roman",base_size = 12)+  

  geom_histogram(aes(y = ..density..),color="black",fill="grey" ,binwidth=1,size=line_th_unit)+  

  geom_density(size=line_th_unit)+  

  labs(x="HC", y="Density")  

p1$theme$line$size<-line_th_unit  

p1$theme$line$size<-line_th_unit  

p1$theme$axis.ticks$size<-line_th_unit  

p1$theme$axis.line$size<-line_th_unit  

p1$theme$panel.grid$colour<-"black"  

p1$theme$axis.ticks$colour<-"black"  

  

# Standard length  

p2<-ggplot(pheno, aes(x=SL1))+  

  ggpibr::theme_pubr(base_family="Times New Roman",base_size = 12)+  

  geom_histogram(aes(y = ..density..),color="black",fill="grey", binwidth=0.1,size=line_th_unit)+  

  labs(x="SL", y="Density")  

  geom_density(size=line_th_unit)+  

  labs(x="SL", y="Density")  

p2$theme$line$size<-line_th_unit

```

```

p2$theme$line$size<-line_th_unit
p2$theme$axis.ticks$size<-line_th_unit
p2$theme$axis.line$size<-line_th_unit
p2$theme$panel.grid $ colour<-"black"
p2$theme$axis.ticks$colour<-"black"
# p2

options(repr.plot.width=6, repr.plot.height=6)
figure <- ggpubr::ggarrange(p1, p2,
  labels = c("a", "b"),
  ncol = 1,
  nrow = 2,
  font.label = list(size = 14,
    color = "black",
    face = "plain",
    family = "Times New Roman"))
figure

ggsave("Fig. 4.1.pdf",device="pdf",units="in",width =6, height =6 )
# ggsave("Fig1.eps",device="eps",units="in",width =7.2)
# ggsave("Fig. 1.1.tiff", width=7.2, units="in", res=300)

```

4.1.2 Bash script for genotyping (Pop-C)

```
#!/bin/bash

#sh TRIM.sh 1>TRIM.log 2>TRIM.err

TRIMMO=/home/lin/GS2018_SNPCalling/soft/Trimmomatic-0.39/trimmomatic-0.39.jar

DIR=/media/lin/HD-SHU3/GS_FMshort2021_NGS

# 4 times Fastq files mv to "Fastq" file
cd ${DIR}
mkdir Fastq
FASTQ=${DIR}/Fastq
mv ${DIR}/JP*/Fastq/* ${FASTQ}
# Use Fastq_re rather than poor quality Fastq.
# A
rm ${FASTQ}/JST291_R*
rm ${FASTQ}/JST537_R*
rm ${FASTQ}/JST69_R*
# B
rm ${FASTQ}/JST1054_R*
rm ${FASTQ}/JST165_R*
# D
rm ${FASTQ}/JST1079_R*
rm ${FASTQ}/JST1275_R*

mv ${DIR}/JP*/Fastq_re/* ${FASTQ}

# Looping for trimming
mkdir ${DIR}/NGSscript
ls ${FASTQ} | sed -e "s/_/\t/g" | cut -f1 | uniq -d | cut -f2 | sort -n > ${DIR}/NGSscript/sample.list
mkdir ${DIR}/TRIM

for i in `cat ${DIR}/NGSscript/sample.list`
do
mkdir ${DIR}/TRIM/${i}
TRIM=${DIR}/TRIM/${i}

java -jar ${TRIMMO} \
```

```

PE -threads 32 \
${FASTQ}/${i}_R1.fastq.gz ${FASTQ}/${i}_R2.fastq.gz \
${TRIM}/${i}"_paired_R1.fq.gz" ${TRIM}/${i}"_unpaired_R1.fq.gz" \
${TRIM}/${i}"_paired_R2.fq.gz" ${TRIM}/${i}"_unpaired_R2.fq.gz" \
ILLUMINACLIP:/home/lin/GS2018_SNPalignment/soft/Trimmomatic-0.39/adapters/NexeraPE-PE.fa:2:30:10
SLIDINGWINDOW:30:20 AVGQUAL:20

```

```

# rm ${FASTQ}/${i}_R1.fastq.gz
# rm ${FASTQ}/${i}_R2.fastq.gz
done

```

```

echo "JST291" > ${DIR}/NGSscript/sample_re.list
echo "JST537" >> ${DIR}/NGSscript/sample_re.list
echo "JST69" >> ${DIR}/NGSscript/sample_re.list
echo "JST1054" >> ${DIR}/NGSscript/sample_re.list
echo "JST165" >> ${DIR}/NGSscript/sample_re.list
echo "JST1079" >> ${DIR}/NGSscript/sample_re.list
echo "JST1275" >> ${DIR}/NGSscript/sample_re.list

```

```

for i in `cat ${DIR}/NGSscript/sample_re.list`
do
mkdir ${DIR}/TRIM/${i}
TRIM=${DIR}/TRIM/${i}

```

```

java -jar ${TRIMMO} \
PE -threads 32 \
${FASTQ}/${i}_re_R1.fastq.gz ${FASTQ}/${i}_re_R2.fastq.gz \
${TRIM}/${i}"_paired_R1.fq.gz" ${TRIM}/${i}"_unpaired_R1.fq.gz" \
${TRIM}/${i}"_paired_R2.fq.gz" ${TRIM}/${i}"_unpaired_R2.fq.gz" \
ILLUMINACLIP:/home/lin/GS2018_SNPalignment/soft/Trimmomatic-0.39/adapters/NexeraPE-PE.fa:2:30:10
SLIDINGWINDOW:30:20 AVGQUAL:20

```

```

# rm ${FASTQ}/${i}_R1.fastq.gz
# rm ${FASTQ}/${i}_R2.fastq.gz
done

```

```

#!/bin/bash

#sh BWA.sh 1>BWA.log 2>BWA.err


DIR=/media/lin/HD-SHU3/GS_FMshort2021_NGS
REF=/home/lin/GS2018_SNPcalling/reference/fr3.fa

#bwa index ${REF}
#samtools faidx ${REF}

cd ${DIR}
mkdir ${DIR}/BWA/

for i in `cat ${DIR}/NGSscript/sample.list`
do
mkdir ${DIR}/BWA/${i}
BWA=${DIR}/BWA/${i}
TRIM=${DIR}/TRIM/${i}

bwa mem -t 32 -M ${REF} ${TRIM}/${i}"_paired_R1.fq.gz" ${TRIM}/${i}"_paired_R2.fq.gz" -R
"@RG\tID:\"${i}\\"\tSM:\"${i}\\"\tPL:Illumina" | samtools view -b -F 2308 -q 10 -@ 32 -o ${BWA}/${i}.bam"
samtools sort -n -@ 32 -o ${BWA}/${i}_namesort.bam ${BWA}/${i}.bam
samtools fixmate -@ 32 -m ${BWA}/${i}_namesort.bam ${BWA}/${i}_fixmate.bam
samtools sort -@ 32 -o ${BWA}/${i}_sorted.bam ${BWA}/${i}_fixmate.bam
samtools index ${BWA}/${i}_sorted.bam

#rm ${BWA}/${i}.sam
#rm ${BWA}/${i}.bam
#rm ${BWA}/${i}_namesort.bam
#rm ${BWA}/${i}_fixmate.bam

done

for i in `cat ${DIR}/NGSscript/sample_re.list`
do
mkdir ${DIR}/BWA/${i}
BWA=${DIR}/BWA/${i}
TRIM=${DIR}/TRIM/${i}

```

```

bwa mem -t 32 -M ${REF} ${TRIM}/${i}"_paired_R1.fq.gz" ${TRIM}/${i}"_paired_R2.fq.gz" -R
"@RG\tID:"${i}"\tSM:"${i}"\tPL:Illumina" | samtools view -b -F 2308 -q 10 -@ 32 -o ${BWA}/${i}.bam"
samtools sort -n -@ 32 -o ${BWA}/${i}_namesort.bam ${BWA}/${i}.bam
samtools fixmate -@ 32 -m ${BWA}/${i}_namesort.bam ${BWA}/${i}_fixmate.bam
samtools sort -@ 32 -o ${BWA}/${i}_sorted.bam ${BWA}/${i}_fixmate.bam
samtools index ${BWA}/${i}_sorted.bam

```

```

#rm ${BWA}/${i}.sam
#rm ${BWA}/${i}.bam
#rm ${BWA}/${i}_namesort.bam
#rm ${BWA}/${i}_fixmate.bam

```

done

```

#!/bin/bash
#sh GATK_GRAS_parallel.sh 1>>GATK_GRAS_parallel.log 2>>GATK_GRAS_parallel.err

```

```

#PBS -N gatk
#PBS -l select=1:ncpus=8:mem=15gb,walltime=2:00:00
#PBS -j oe

```

```
echo "START -----"
```

```

#module add java/12.0.2
#module load parallel
#module load gatk

```

```

DIR=/media/lin/HD-SHU3/GS_FMshort2021_NGS
REF=/home/lin/GS2018_SNPcalling/reference/fr3.fa
gatk=/home/lin/gatk-4.1.6.0/gatk

```

```

cd ${DIR}
mkdir VCF

```

```

### Use gnu-parallel to use multiple cores
### within one script

```

```

cat ${DIR}/NGSscript/sample.list | parallel --verbose -j 6 "${gatk} HaplotypeCaller --output-mode
EMIT_ALL_CONFIDENT_SITES -stand-call-conf 30 -R ${REF} -I ${DIR}/BWA/{}_{}/{}_sorted.bam -O
${DIR}/VCF/{}_pe_variants_gatk.g.vcf.gz -ERC GVCF"

cat ${DIR}/NGSscript/sample_re.list | parallel --verbose -j 6 "${gatk} HaplotypeCaller --output-mode
EMIT_ALL_CONFIDENT_SITES -stand-call-conf 30 -R ${REF} -I ${DIR}/BWA/{}_{}/{}_sorted.bam -O
${DIR}/VCF/{}_pe_variants_gatk.g.vcf.gz -ERC GVCF"
echo "FINISH -----"

## Bash

## Call variants per-sample (GVCF mode)

## Variables
DIR=/media/lin/HD-SHU3/GS_FMshort2021_NGS
DIR2=/media/lin/HD-SHU3/GS_FMshort2021_NGS/NGSscript
REF=/home/lin/GS2018_SNPcalling/reference/fr3.fa
gatk=/home/lin/gatk-4.1.6.0/gatk

#rm ${DIR}/BWA/*/*_pe_variants_gatk.g.*
#cp ${DIR}/BWA/*/*_pe_variants_gatk.g.* ${DIR}/VCF
##sudo chmod +x *

echo "${gatk} CombineGVCFs --java-options "-Xmx100G"" > ${DIR2}/CombineGVCFs.sh
for i in `cat ${DIR2}/sample.list`
do
echo "-V ${DIR}/VCF/${i}_pe_variants_gatk.g.vcf.gz" >> ${DIR2}/CombineGVCFs.sh
done
echo "-R ${REF}" >> ${DIR2}/CombineGVCFs.sh
echo "-O ${DIR}/VCF/cohort.g.vcf.gz" >> ${DIR2}/CombineGVCFs.sh

cat ${DIR2}/CombineGVCFs.sh | tr "\n" " " > ${DIR2}/Spa_CombineGVCFs.sh

bash ${DIR2}/Spa_CombineGVCFs.sh

# tabix -p vcf ${DIR}/VCF/cohort.g.vcf.gz

${gatk} GenotypeGVCFs -R ${REF} -V ${DIR}/VCF/cohort.g.vcf.gz -O

```

```

${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --tmp-dir=${DIR}/VCF/temp

#bgzip -d ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz
#grep "##contig" ${DIR}/VCF/Output_jonit_call_cohort.vcf | sed -e "s/= /g" -e "s/,/ /g" | awk '{print $3}' >
${DIR}/VCF/pos.list
#cat -n ${DIR}/VCF/pos.list | awk '{print $2"\t"$1}'> ${DIR}/VCF/Chom.map
#bgzip ${DIR}/VCF/Output_jonit_call_cohort.vcf
tabix -p vcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz
#vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --min-meanDP 6 --max-meanDP 500 --max-
missing 0.3 --remove-indels --recode --stdout > ${DIR}/VCF/Hetero_realigned_cov10_filtered.vcf

```

```

## Line plot to visualize.

## --min-meanDP 6 to 10, and --max-missing 0-1
for i in $(seq 0 0.1 1)
do
echo $i >> ${DIR}/NGSscript/filter_parameter2.txt
done

#single --min-meanDP-5
(cat ${DIR}/NGSscript/filter_parameter2.txt | parallel --verbose --joblog parallel.log "vcftools --gzvcf
${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --chr chr1 --chr chr2 --chr chr3 --chr chr4 --chr chr5 --chr chr6 --
chr chr7 --chr chr8 --chr chr9 --chr chr10 --chr chr11 --chr chr12 --chr chr13 --chr chr14 --chr chr15 --chr chr16 -
--chr chr17 --chr chr18 --chr chr19 --chr chr20 --chr chr21 --chr chr22 --remove-indels --min-alleles 2 --max-
alleles 2 --minQ 10 --min-meanDP 5 --max-meanDP 500 --maf 0.01 --hwe 0.05 --max-missing {} " 2>&1) |tee
${DIR}/NGSscript/log.txt
cat ${DIR}/NGSscript/log.txt |grep possible|sed s/"After filtering, kept"/g |sed s/"out of a possible 2476352
Sites"/g |sort -n

```

23798 out of a possible 2466708 Sites
36306 out of a possible 2466708 Sites
36566 out of a possible 2466708 Sites
36697 out of a possible 2466708 Sites
36767 out of a possible 2466708 Sites
36829 out of a possible 2466708 Sites

```
36870 out of a possible 2466708 Sites
36929 out of a possible 2466708 Sites
36972 out of a possible 2466708 Sites
37027 out of a possible 2466708 Sites
37380 out of a possible 2466708 Sites
```

```
#single --min-meanDP-10
(cat ${DIR}/NGSscript/filter_parameter2.txt | parallel --verbose --joblog parallel.log "vcftools --gzvcf
${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --chr chr1 --chr chr2 --chr chr3 --chr chr4 --chr chr5 --chr chr6 --
chr chr7 --chr chr8 --chr chr9 --chr chr10 --chr chr11 --chr chr12 --chr chr13 --chr chr14 --chr chr15 --chr chr16 -
--chr chr17 --chr chr18 --chr chr19 --chr chr20 --chr chr21 --chr chr22 --remove-indels --min-alleles 2 --max-
alleles 2 --minQ 10 --min-meanDP 10 --max-meanDP 500 --maf 0.01 --hwe 0.05 --max-missing {} " 2>&1) |tee
${DIR}/NGSscript/log2.txt
cat ${DIR}/NGSscript/log2.txt |grep possible|sed s/"After filtering, kept"/g |sed s/"out of a possible 2476352
Sites"/g |sort -n
```

```
23504 out of a possible 2466708 Sites
31417 out of a possible 2466708 Sites
31567 out of a possible 2466708 Sites
31640 out of a possible 2466708 Sites
31686 out of a possible 2466708 Sites
31726 out of a possible 2466708 Sites
31745 out of a possible 2466708 Sites
31773 out of a possible 2466708 Sites
31801 out of a possible 2466708 Sites
31830 out of a possible 2466708 Sites
31980 out of a possible 2466708 Sites
```

```
## SNP filtering and matrix, not down
mkdir ${DIR}/Post_VCF_processing/
filter=${DIR}/Post_VCF_processing

task(){
mkdir ${filter}/${i};
vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --chr chr1 --chr chr2 --chr chr3 --chr chr4 --chr
```

```

chr5 --chr chr6 --chr chr7 --chr chr8 --chr chr9 --chr chr10 --chr chr11 --chr chr12 --chr chr13 --chr chr14 --chr
chr15 --chr chr16 --chr chr17 --chr chr18 --chr chr19 --chr chr20 --chr chr21 --chr chr22 --remove-indels --min-
alleles 2 --max-alleles 2 --minQ 10 --min-meanDP 10 --max-meanDP 500 --maf 0.01 --hwe 0.05 --max-missing
${i} --recode --stdout > ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf;
zgrep -v "#" ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf | awk '{print $1"\t"$2}' >
${filter}/${i}/position.list;
cat -n ${filter}/${i}/position.list | awk '{print $2"\t"$1}' > ${filter}/${i}/Chom.map;
bgzip ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf;
tabix -p vcf ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf.gz;
vcftools --gzvcf ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf.gz --plink --chrom-map
${filter}/${i}/Chom.map --out ${filter}/${i}/Hetero_realigned_cov10_filtered;
plink1 --ped ${filter}/${i}/Hetero_realigned_cov10_filtered.ped --map
${filter}/${i}/Hetero_realigned_cov10_filtered.map --recodeA --out
${filter}/${i}/Hetero_realigned_cov10_filtered --noweb;
echo "empty" > ${filter}/${i}/IDlist;
zgrep "#CHROM" ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf.gz | cut -f10- | sed -e "s/\t/\n/g" >>
${filter}/${i}/IDlist;
sed -e "s/empty//g" ${filter}/${i}/IDlist > ${filter}/${i}/IDs.txt
cat ${filter}/${i}/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- >
${filter}/${i}/Hetero_realigned_cov10_filtered2.raw;
paste -d " " ${filter}/${i}/IDs.txt ${filter}/${i}/Hetero_realigned_cov10_filtered2.raw >
${filter}/${i}/Hetero_realigned_cov10_filtered3.raw;
sed -n "1p" ${filter}/${i}/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- | sed -e "s/ \n/g" >
${filter}/${i}/Hetero_realigned_cov10_filtered.pos;
less ${filter}/${i}/Hetero_realigned_cov10_filtered.pos|sed -e "s/_/\t/g"|sed -e "s/:/\t/g"|sed -e "s/-/\t/g"|sed
"s/"chr"/g"| awk '{print $1, $2}'>${filter}/${i}/lin.map
}
# for i in $(seq 0 0.1 1);do task "$i" &done
i=0.7
task "$i"

```

```

Im_soft=/home/lin/GS2018_SNPalignment/VCF_imputation/20200114_linkimpute
mkdir ${DIR}/VCF_imputation
imputation=${DIR}/VCF_imputation
## Imputation
taskIM(){
mkdir ${imputation}/${i}
java -jar ${Im_soft}/LinkImpute.jar ${filter}/${i}/Hetero_realigned_cov10_filtered.raw

```

```

${imputation}/${i}/Imputed.raw;
cat ${imputation}/${i}/Imputed.raw | cut -d " " -f7- >
${imputation}/${i}/Hetero_realigned_cov10_filtered2.raw;
cp ${filter}/${i}/IDs.txt ${imputation}/${i}/IDs.txt
cp ${filter}/${i}/lin.map ${imputation}/${i}/lin.map
paste -d " " ${imputation}/${i}/IDs.txt ${imputation}/${i}/Hetero_realigned_cov10_filtered2.raw >
${imputation}/${i}/Hetero_realigned_cov10_filtered3.raw;
}
taskIM "$i"

```

```

# Post_VCF_2wSNPs210922
mkdir ${DIR}/Post_VCF_2wSNPs211021
filter=${DIR}/Post_VCF_2wSNPs211021

task(){
mkdir ${filter}/${i};
vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --chr chr1 --chr chr2 --chr chr3 --chr chr4 --chr
chr5 --chr chr6 --chr chr7 --chr chr8 --chr chr9 --chr chr10 --chr chr11 --chr chr12 --chr chr13 --chr chr14 --chr
chr15 --chr chr16 --chr chr17 --chr chr18 --chr chr19 --chr chr20 --chr chr21 --chr chr22 --remove-indels --min-
alleles 2 --max-alleles 2 --minQ 20 --min-meanDP 20 --minDP 10 --max-meanDP 300 --maf 0.01 --hwe 0.05 --
max-missing ${i} --recode --stdout > ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf;
zgrep -v "#" ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf | awk '{print $1"\t"$2}' >
${filter}/${i}/position.list;
cat -n ${filter}/${i}/position.list | awk '{print $2"\t"$1}' > ${filter}/${i}/Chom.map;
bgzip ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf;
tabix -p vcf ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf.gz;
vcftools --gzvcf ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf.gz --plink --chrom-map
${filter}/${i}/Chom.map --out ${filter}/${i}/Hetero_realigned_cov10_filtered;
plink1 --ped ${filter}/${i}/Hetero_realigned_cov10_filtered.ped --map
${filter}/${i}/Hetero_realigned_cov10_filtered.map --recodeA --out
${filter}/${i}/Hetero_realigned_cov10_filtered --noweb;
echo "empty" > ${filter}/${i}/IDlist;
zgrep "#CHROM" ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf.gz | cut -f10- | sed -e "s/\t/\n/g" >>
${filter}/${i}/IDlist;
sed -e "s/empty//g" ${filter}/${i}/IDlist > ${filter}/${i}/IDs.txt
cat ${filter}/${i}/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- >
${filter}/${i}/Hetero_realigned_cov10_filtered2.raw;

```

```

paste -d" " ${filter}/${i}/IDs.txt ${filter}/${i}/Hetero_realigned_cov10_filtered2.raw >
${filter}/${i}/Hetero_realigned_cov10_filtered3.raw;
sed -n "1p" ${filter}/${i}/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- | sed -e "s/ \n/g" >
${filter}/${i}/Hetero_realigned_cov10_filtered.pos;
less ${filter}/${i}/Hetero_realigned_cov10_filtered.pos|sed -e "s/_t/g"|sed -e "s:/t/g"|sed -e "s/-t/g"|sed
"s/"chr"/g"| awk '{print $1, $2}'>${filter}/${i}/lin.map
}

# for i in $(seq 0 0.1 1);do task "$i" &done
i=0.7
task "$i"

```

kept 19227 out of a possible 2466708 Sites

```

# imputaion
Im_soft=/home/lin/GS2018_SNPcalling/VCF_imputation/20200114_linkimpute
mkdir ${DIR}/VCF_imputation_211021
imputation=${DIR}/VCF_imputation_211021
taskIM(){
mkdir ${imputation}/${i}
java -jar ${Im_soft}/LinkImpute.jar ${filter}/${i}/Hetero_realigned_cov10_filtered.raw
${imputation}/${i}/Imputed.raw;
cat ${imputation}/${i}/Imputed.raw | cut -d " " -f7- >
${imputation}/${i}/Hetero_realigned_cov10_filtered2.raw;
cp ${filter}/${i}/IDs.txt ${imputation}/${i}/IDs.txt
cp ${filter}/${i}/lin.map ${imputation}/${i}/lin.map
paste -d" " ${imputation}/${i}/IDs.txt ${imputation}/${i}/Hetero_realigned_cov10_filtered2.raw >
${imputation}/${i}/Hetero_realigned_cov10_filtered3.raw;
}

```

taskIM "\$i"

```

####
# Ne = 26.3
# vcf2genepop https://github.com/z0on/2bRAD_denovo/blob/master/vcf2genepop.pl

gunzip ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf.gz
/home/lin/GS2018_SNPcalling/Haplo_20210102/vcf2genepop.pl
vcf=${filter}/${i}/Hetero_realigned_cov10_filtered.vcf> ${filter}/filtered_Ne.gen

```

```

bgzip ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf

awk '/chr/ { print; print "pop"; next }' ${filter}/filtered_Ne.gen >> ${filter}/filtered_Ne_pop.gen

##not done

###

vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --chr chr1 --chr chr2 --chr chr3 --chr chr4 --chr
chr5 --chr chr6 --chr chr7 --chr chr8 --chr chr9 --chr chr10 --chr chr11 --chr chr12 --chr chr13 --chr chr14 --chr
chr15 --chr chr16 --chr chr17 --chr chr18 --chr chr19 --chr chr20 --chr chr21 --chr chr22 --remove-indels --min-
alleles 2 --max-alleles 2 --minQ 10 --min-meanDP 10 --max-meanDP 500 --maf 0.01 --hwe 0.05 --max-missing
1 --recode --stdout > ${DIR}/VCF_imputation/1/Hetero_realigned_cov10_filtered.vcf

zgrep -v "##" ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.vcf | awk '{print $1"\t"$2}' >
${DIR}/VCF_imputation/${i}/position.list
cat -n ${DIR}/VCF_imputation/${i}/position.list | awk '{print $2"\t"$1}' >
${DIR}/VCF_imputation/${i}/Chom.map
bgzip ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.vcf
tabix -p vcf ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.vcf.gz
vcftools --gzvcf ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.vcf.gz --plink --chrom-map
${DIR}/VCF_imputation/${i}/Chom.map --out ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered
plink1 --ped ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.ped --map
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.map --recodeA --out
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered --noweb
echo "empty" > ${DIR}/VCF_imputation/${i}/IDlist
zgrep "#CHROM" ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.vcf.gz | cut -f10- | sed -e
"s/\t/\n/g" >> ${DIR}/VCF_imputation/${i}/IDlist
sed -e "s/empty//g" ${DIR}/VCF_imputation/${i}/IDlist > ${DIR}/VCF_imputation/${i}/IDs.txt
cat ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- >
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered2.raw
paste -d " " ${DIR}/VCF_imputation/${i}/IDs.txt
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered2.raw >
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered3.raw
sed -n "1p" ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- | sed -e "s/\n/g"
> ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.pos

```

```
less ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.pos|sed -e "s/_/\t/g"|sed -e "s:/\t/g"/sed -e "s/-/\t/g"|awk '{print $2,$3+$5}'>${DIR}/VCF_imputation/${i}/lin.map
```

```
##not done
```

```
## from plot, decide the filtering parameters and imputation?
```

```
## how many training individuals are necessary?
```

```
#multiple
```

```
cd ${DIR}/VCF_filtering
```

```
for i in {6..15}
```

```
do
```

```
(cat ${DIR}/VCF_filtering/filter_parameter2.txt | parallel --verbose --joblog parallel.log "vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --chr chr1 --chr chr2 --chr chr3 --chr chr4 --chr chr5 --chr chr6 --chr chr7 --chr chr8 --chr chr9 --chr chr10 --chr chr11 --chr chr12 --chr chr13 --chr chr14 --chr chr15 --chr chr16 --chr chr17 --chr chr18 --chr chr19 --chr chr20 --chr chr21 --chr chr22 --remove-indels --min-alleles 2 --max-alleles 2 --minQ 10 --min-meanDP ${i} --max-meanDP 500 --maf 0.01 --hwe 0.05 --max-missing {}" 2>&1) | tee ${DIR}/VCF_filtering/"log"${i}.txt"
```

```
done
```

```
parallel --verbose "cat log{}.txt |grep possible|sed s/'After filtering, kept'//g |sed s/'out of a possible 2196091
```

```
Sites'//g |sort >> SNP_count_min-meanDP_{}.txt" :: {5..15}
```

```
paste $(ls|grep SNP_count_min-meanDP_|sort -t " " -k4 -g) > merge.txt
```

```
DIR=/media/lin/HD-SHU3/GS_FMshort2021_NGS
```

```
fastqfile=${DIR}/Fastq
```

```
mkdir ${DIR}/seqstats_result
```

```
statsdir=${DIR}/seqstats_result
```

```
TRIMfile=${DIR}/TRIM
```

```
BWAfile=${DIR}/BWA
```

```
#raw
```

```
ls ${fastqfile} | cat |grep _R1.fastq.gz | parallel --verbose -j 6 "/home/lin/seqstats/seqstats ${fastqfile}/{} |sed -n 1p|cut -f 2 >> ${statsdir}/raw_num_read_1.txt"
```

```
/home/lin/seqstats/seqstats ${fastqfile}/{} |sed -n 2p|cut -f 2|cut -d' ' -f1 >> ${statsdir}/raw_bp_1.txt"
```

```
ls ${fastqfile} | cat |grep _R2.fastq.gz | parallel --verbose -j 6 "/home/lin/seqstats/seqstats ${fastqfile}/{} |sed -n
```

```

1p|cut -f 2 >> ${statsdir}/raw_num_read_2.txt|
/home/lin/seqstats/seqstats ${fastqfile}{} |sed -n 2p|cut -f 2|cut -d' ' -f1 >> ${statsdir}/raw_bp_2.txt"
#trim
ls ${TRIMfile} | cat | parallel --verbose -j 10 "/home/lin/seqstats/seqstats ${TRIMfile}{}{}_paired_R1.fq.gz
|sed -n 1p|cut -f 2 >> ${statsdir}/trimed_num_read_1.txt|
/home/lin/seqstats/seqstats ${TRIMfile}{}{}_paired_R1.fq.gz |sed -n 2p|cut -f 2|cut -d' ' -f1 >>
${statsdir}/trimed_bp_1.txt"
ls ${TRIMfile} | cat | parallel --verbose -j 10 "/home/lin/seqstats/seqstats ${TRIMfile}{}{}_paired_R2.fq.gz
|sed -n 1p|cut -f 2 >> ${statsdir}/trimed_num_read_2.txt|
/home/lin/seqstats/seqstats ${TRIMfile}{}{}_paired_R2.fq.gz |sed -n 2p|cut -f 2|cut -d' ' -f1 >>
${statsdir}/trimed_bp_2.txt"
#map
ls ${BWAfile} | cat | parallel --verbose -j 10 "samtools flagstat ${BWAfile}{}{}_sorted.bam |sed -n 1p |sed -e
's/ ^t/g' |cut -f 1 >> ${statsdir}/mapped_num_read.txt"

cp ${DIR}/seqstats_result/* ~/GS_FMshort2021_NGS/seqstats_result

```

4.1.3 R script for genotyping (Pop-C)

```
rb1<-read.table("raw_bp_1.txt")
rb2<-read.table("raw_bp_2.txt")
rn1<-read.table("raw_num_read_1.txt")
rn2<-read.table("raw_num_read_2.txt")
tb1<-read.table("trimed_bp_1.txt")
tb2<-read.table("trimed_bp_2.txt")
tn1<-read.table("trimed_num_read_1.txt")
tn2<-read.table("trimed_num_read_2.txt")

nm<-read.table("mapped_num_read.txt")

(sum(rb1)+sum(rb2))
(sum(tb1)+sum(tb2))
(sum(tb1)+sum(tb2))/(sum(rb1)+sum(rb2))

mean(unlist(rn1+rn2))
sd(unlist(rn1+rn2))

mean(unlist(tn1+tn2))
sd(unlist(tn1+tn2))

mean(unlist(tn1+tn2))/mean(unlist(rn1+rn2))

(sum(rn1)+sum(rn2))
(sum(tn1)+sum(tn2))
(sum(tn1)+sum(tn2))/(sum(rn1)+sum(rn2))

(sum(rb1)+sum(rb2))/(sum(rn1)+sum(rn2))
(sum(tb1)+sum(tb2))/(sum(tn1)+sum(tn2))
```

4.1.4 Python script for population structure (Pop-C)

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm
import seaborn as sns; sns.set()
import matplotlib as mpl

# Loading data
pheno = pd.read_csv("./pheno_BCW_cleaned_ngs.csv")
#I = pd.read_table("./Hetero_realigned_cov10_filtered3.raw",sep="\s+")
I = pd.read_table("./reorderG.txt",sep="\s+")-1
bcw=pheno["BCW"]

from sklearn.manifold import TSNE
model = TSNE(random_state=0,perplexity=20)
tsne5 = model.fit_transform(I)
lin_wide=1.28
font_size=12
%matplotlib inline
%config InlineBackend.figure_format = 'svg'
plt.style.use('seaborn-white')
plt.style.use('seaborn-paper')
plt.rc('font',size=font_size)
plt.rc('lines',lw=lin_wide)
plt.rcParams['font.family'] = 'serif'
plt.rcParams['font.sans-serif'] = 'Times New Roman'
#plt.rcParams['axes.formatter.use_mathtext'] = True
#plt.rcParams['mathtext.bf'] = 'serif:normal'
#plt.rcParams['text.hinting'] = 'none'
#plt.rcParams['text.usetex'] = True
#plt.rcParams['axes.labelweight'] = 'normal'

plt.rcParams['axes.linewidth'] = lin_wide
plt.rcParams['xtick.major.width'] = lin_wide
plt.rcParams['xtick.minor.width'] = lin_wide
plt.rcParams['xtick.labelsize'] = font_size
plt.rcParams['ytick.major.width'] = lin_wide
```

```
plt.rcParams['ytick.minor.width'] = lin_width
plt.rcParams['ytick.labelsize'] = font_size
plt.rcParams['axes.spines.bottom'] = True
# plt.style.use('bmh')
plt.scatter(tsne5[:, 0], tsne5[:, 1], edgecolor='k', c=bcw, cmap=cm.Reds, linewidth=lin_width, s=s_value)
plt.xlabel("t-SNE coordinate 1", size=font_size)
plt.ylabel("t-SNE coordinate 2", size=font_size)
plt.subplots_adjust(bottom=0.1, right=0.8, top=0.9)
cax = plt.axes([0.85, 0.1, 0.075, 0.8])
plt.colorbar(cax=cax)
plt.gcf().set_size_inches(7, 6)
plt.savefig("C5.3_tSNE_HC.pdf", format="pdf")
plt.show()
# plt.savefig("PCA_diets.png", dpi=300)
```

4.1.5 R script for heritability and genetic correlation (Pop-C)

```
reorderG<-read.table("GS2019/reorderG.txt")
x<-as.matrix(reorderG)-1
pheno<-read.csv("GS2019/pheno_BCW_cleaned_ngs.csv")
n<-dim(x)[1]
# library(caret)
# bctbcw<-BoxCoxTrans(bcw+1)
# bctbcw
# predictbcw<-predict(bctbcw, bcw+1) # ((bcw+1)^0.6-1)/0.6
predictbcw<-((pheno$BCW+1)^0.6-1)/0.6

library("sommer")
packageVersion("sommer")
A<- A.mat(x,n.core=8)
row.names(A)=1:n;colnames(A)=1:n
data <- data.frame(tbcw=predictbcw,length=X190930_SL,id=factor(1:n))
attach(data)
ans.m <- mmer(cbind(tbcw,length)~1,random=~ vs(id, Gu=A, Gtc=unsm(2)),
               rcov=~ vs(units, Gtc=unsm(2)),
               data=data)
(sp_cor<-cov2cor(ans.m$sigma$`u:id`))
pin(ans.m, h2~ V1/(V1+V4))
pin(ans.m, h2~ V3/(V3+V6))
```

4.1.6 R script for GWAS (Pop-C)

```
R.Version()$version.string  
library("ggplot2")  
library("ggpubr")  
library("extrafont")  
print(paste("ggplot2 version",packageVersion("ggplot2")))  
print(paste("ggpubr version",packageVersion("ggpubr")))  
print(paste("extrafont version",packageVersion("extrafont")))  
# remotes::install_version("Rttf2pt1", version = "1.3.8")  
# ttf_import()  
# fonts()  
# loadfonts(device = "pdf")  
  
phenoRaw <- read.csv("./200401H31_modified.csv")  
# attach(pheno)  
# n<-length(pheno$BW)  
head(phenoRaw)  
str(phenoRaw)  
  
R.Version()$version.string  
library("ggplot2")  
library("ggpubr")  
library("extrafont")  
print(paste("ggplot2 version",packageVersion("ggplot2")))  
print(paste("ggpubr version",packageVersion("ggpubr")))  
print(paste("extrafont version",packageVersion("extrafont")))  
# remotes::install_version("Rttf2pt1", version = "1.3.8")  
# ttf_import()  
# fonts()  
# loadfonts(device = "pdf")  
  
phenoRaw <- read.csv("./200401H31_modified.csv")  
# attach(pheno)  
# n<-length(pheno$BW)  
head(phenoRaw)  
str(phenoRaw)  
  
# Filration
```

```

# pheno_BCW_cleaned<-pheno[!is.na(pheno$BCW),]
pheno_BCW_cleaned1<-phenoRaw[!is.na(phenoRaw$BCW),]
pheno_BCW_cleaned<-pheno_BCW_cleaned1[pheno_BCW_cleaned1$memo!="Mistake",]
str(pheno_BCW_cleaned)

pheno<-data.frame(SL1=pheno_BCW_cleaned$X190930_SL,
                   BW1=pheno_BCW_cleaned$X190930_BW,
                   tank_f=factor(pheno_BCW_cleaned$X190930_tank),
                   SL2=pheno_BCW_cleaned$X191108SL,
                   BW2=pheno_BCW_cleaned$X191108BW,
                   bcw=pheno_BCW_cleaned$BCW,
                   HD=pheno_BCW_cleaned$BCW/(pheno_BCW_cleaned$X191108SL)^2*100,
                   egg=factor(pheno_BCW_cleaned$egg),
                   human = factor(pheno_BCW_cleaned$Conducter),
                   deltaSL=(pheno_BCW_cleaned$X191108SL-
                   pheno_BCW_cleaned$X190930_SL)/pheno_BCW_cleaned$X190930_SL*100,
                   deltaBW=(pheno_BCW_cleaned$X191108BW-
                   pheno_BCW_cleaned$X190930_BW)/pheno_BCW_cleaned$X190930_BW*100)
n<-length(pheno$bcw)
attach(pheno)
str(pheno)

cor(SL1,deltaSL)

# Loading raw G matrix
geno<-read.table("./Imputation/Hetero_realigned_cov10_filtered3.raw", header=T, row.names=1)
dim(geno)

# reorder
reorderIndex<-c()
for (i in pheno_BCW_cleaned$JSTNGSID){
  reorderIndex<-c(reorderIndex,match(i,row.names(geno)))
}
reorderG<-geno[reorderIndex,]
row.names(reorderG)=1:dim(reorderG)[1]
x <- as.matrix(reorderedG)-1
head(reorderedG)
str(x)

```

```

write.table(reorderG,"reorderG.txt")
write.csv(pheno_BCW_cleaned,"pheno_BCW_cleaned_ngs.csv")

# GBLUP
library(rrBLUP)
print(paste("rrBLUP version",packageVersion("rrBLUP")))

# genomic relationship matrix
A <- A.mat(x)
row.names(A)=1:n;colnames(A)=1:n

library(caret)
bctbcw<-BoxCoxTrans(bcw+1)
bctbcw
predictbcw<-predict(bctbcw, bcw+1) # ((bcw+1)^0.6-1)/0.6

# data
data <- data.frame(tbcw=predictbcw,length=pheno$SL1,SL2=pheno$SL2,
                     bcw_d_length2=pheno$HD,BW=pheno$BW1,BW2=pheno$BW2,
                     deltaSL=pheno$deltaSL, deltaBW=pheno$deltaBW,gid=1:n,x=x)
row.names(A)=1:n;colnames(A)=1:n

lin_map<-read.table("./Imputation/lin.map")
g <- data.frame(rownames(lin_map),lin_map$V1, lin_map$V2, t(x))
rownames(g) <-1:nrow(g)
colnames(g) <-c("marker", "chrom", "pos", rownames(x))
# Bonferroni-corrected significance threshold
(thred <- round(-log10(0.05/nrow(g)),3))

head(g)

table(unlist(g[(g$chrom==3)&(g$pos==5531819),][c(-1,-2,-3)]))

mean(data$tbcw)
mean(data[unlist(g[(g$chrom==3)&(g$pos==5531819),][c(-1,-2,-3)])==-1,]$tbcw)
mean(data[unlist(g[(g$chrom==3)&(g$pos==5531819),][c(-1,-2,-3)])==0,]$tbcw)
mean(data[unlist(g[(g$chrom==3)&(g$pos==5531819),][c(-1,-2,-3)])==1,]$tbcw)

```

```

data["bcw"]<-pheno$bcw
mean(data$bcw)
mean(data[unlist(g[(g$chrom==3)&(g$pos==5531819),][c(-1,-2,-3)])==-1,]$bcw)
mean(data[unlist(g[(g$chrom==3)&(g$pos==5531819),][c(-1,-2,-3)])==0,]$bcw)
mean(data[unlist(g[(g$chrom==3)&(g$pos==5531819),][c(-1,-2,-3)])==1,]$bcw)
sd(data[unlist(g[(g$chrom==3)&(g$pos==5531819),][c(-1,-2,-3)])==1,]$bcw)

sd(data[unlist(g[(g$chrom==3)&(g$pos==5531819),][c(-1,-2,-3)])==-1,]$bcw)

bcwframe <-data.frame(1:n, data$tbcw)
colnames(bcwframe)<-c("gid", "tbcw")
# Performing GWAS
GWAS_bcw<-GWAS(bcwframe, g,K=A, n.PC = 10)
# SNP markers with lowest P-values
GWAS_bcw$chrom[head(order(GWAS_bcw$tbcw,decreasing = TRUE),3)]
# Corresponding position (base pair)
GWAS_bcw$pos[head(order(GWAS_bcw$tbcw,decreasing = TRUE),3)]

str(GWAS_bcw)

# top100
top100<-cbind(rep("GS2019",100),GWAS_bcw$chrom[head(order(GWAS_bcw$tbcw,decreasing = TRUE),100)],GWAS_bcw$pos[head(order(GWAS_bcw$tbcw,decreasing = TRUE),100)])
colname<-c("Exp","chr","pos")
write.csv(top100,"GS2019_GAWS_HC100.csv")

# Manhattan plot
line_th_unit<-0.6
Chrom<-GWAS_bcw$chrom
GWAS_bcw$chrom<-factor(Chrom,levels=c(as.character(1:22)))
options(repr.plot.width=8, repr.plot.height=4)
my_y_title <- expression(paste("-", "log"[10], "(", italic("p"), ") for HC"))
p1<-ggplot(GWAS_bcw, aes(x=1:dim(x)[2], y=tbcw, color=chrom))+
  ggpibr::theme_pubr(base_family="Times New Roman",base_size = 12,border = TRUE)+
  geom_point()+
  scale_color_manual(values = rep(c("black", "skyblue"), 24 ))+

```

```

labs(x="Physical order of SNPs", y=my_y_title)+
theme(legend.position="none",
axis.text.y=element_text(size=12),
axis.text.x=element_text(size=12),
text=element_text(size=12,
family="Times New Roman"))+
geom_hline(yintercept=thred, linetype="dashed", color = "red")+
geom_text(aes(0,thred,label = thred, vjust = 1.2),color = "black",family="Times New Roman")

p1$theme$line$size<-line_th_unit
p1$theme$line$size<-line_th_unit
p1$theme$axis.ticks$size<-line_th_unit
p1$theme$axis.line$size<-line_th_unit
p1$theme$panel.grid$colour<-"black"
p1$theme$axis.ticks$colour<-"black"

p1

```

```

lengthframe <-data.frame(1:n, SL1)
colnames(lengthframe) <-c("gid", "length")
# Performing GWAS
GWAS_length <-GWAS(lengthframe, g,K=A, n.PC = 10)
# SNP markers with lowest P-values
GWAS_length$chrom[head(order(GWAS_length$length,decreasing = TRUE),3)]
# Corresponding position (base pair)
GWAS_length$pos[head(order(GWAS_length$length,decreasing = TRUE),3)]

# top100
top100<-cbind(rep("GS2019",100),GWAS_length$chrom[head(order(GWAS_length$length,decreasing = TRUE),100)],
GWAS_length$pos[head(order(GWAS_length$length,decreasing = TRUE),100)])
colname<-c("Exp","chr","pos")
write.csv(top100,"GS2019_GAWS_SL100.csv")

```

```

Chrom<-GWAS_length$chrom
GWAS_length$chrom<-factor(Chrom,levels=c(as.character(1:22)))
options(repr.plot.width=8, repr.plot.height=4)
my_y_title <- expression(paste("-", "log"[10], "(", italic("p"), ") for SL"))
p2<-ggplot(GWAS_length, aes(x=1:dim(x)[2], y=length, color=chrom)) +
ggnpubr::theme_pubr(base_family="Times New Roman",base_size = 12,border = TRUE)+
```

```

geom_point()+
  scale_color_manual(values = rep(c("black", "skyblue"), 24 ))+
  labs(x="Physical order of SNPs", y=my_y_title)+
  theme(legend.position="none",
        axis.text.y=element_text(size=12),
        axis.text.x=element_text(size=12),
        text=element_text(size=12,
                          family="Times New Roman"))+
  geom_hline(yintercept=thred, linetype="dashed", color = "red")+
  geom_text(aes(0,thred,label = thred, vjust = 1.2),color = "black",family="Times New Roman")

p2$theme$line$size<-line_th_unit
p2$theme$line$size<-line_th_unit
p2$theme$axis.ticks$size<-line_th_unit
p2$theme$axis.line$size<-line_th_unit
p2$theme$panel.grid$colour<-"black"
p2$theme$axis.ticks$colour<-"black"

p2

options(repr.plot.width=5, repr.plot.height=5)
figure <- ggpubr::ggarrange(p1, p2,
                            labels = c("a", "b"),
                            ncol = 1,
                            nrow = 2,
                            font.label = list(size = 14,
                                              color = "black",
                                              face = "plain",
                                              family = "Times New Roman"))

# figure

ggsave("Fig. 4.2.pdf",device="pdf",units="in",width =6, height = 6 )
# ggsave("Fig2.eps",device="eps",units="in",width =7.2)
# ggsave("Fig. 1.2.tiff", width=7.2, units="in", res=300)

## GWAS for deltaSL

lengthframe <-data.frame(1:n, deltaSL)
colnames(lengthframe) <-c("gid", "length")
# Performing GWAS

```

```

GWAS_length <-GWAS(lengthframe, g,K=A, n.PC = 10)
# SNP markers with lowest P-values
GWAS_length$chrom[head(order(GWAS_length$length,decreasing = TRUE),3)]
# Corresponding position (base pair)
GWAS_length$pos[head(order(GWAS_length$length,decreasing = TRUE),3)]]

Chrom<-GWAS_length$chrom
GWAS_length$chrom<-factor(Chrom,levels=c(as.character(1:22)))
options(repr.plot.width=8, repr.plot.height=4)
my_y_title <- expression(paste("-", "log"[10], "(", italic("p"), ") for deltaSL"))
p2<-ggplot(GWAS_length, aes(x=1:dim(x)[2], y=length, color=chrom)) +
  ggpibr::theme_pubr(base_family="Times New Roman",base_size = 12,border = TRUE) +
  geom_point() +
  scale_color_manual(values = rep(c("black", "skyblue"), 24 )) +
  labs(x="Physical order of SNPs", y=my_y_title) +
  theme(legend.position="none",
        axis.text.y=element_text(size=12),
        axis.text.x=element_text(size=12),
        text=element_text(size=12,
                          family="Times New Roman")) +
  geom_hline(yintercept=thred, linetype="dashed", color = "red") +
  geom_text(aes(0,thred,label = thred, vjust = 1.2),color = "black",family="Times New Roman")
p2$theme$line$size<-line_th_unit
p2$theme$line$size<-line_th_unit
p2$theme$axis.ticks$size<-line_th_unit
p2$theme$axis.line$size<-line_th_unit
p2$theme$panel.grid$colour<-"black"
p2$theme$axis.ticks$colour<-"black"
p2

## GWAS for deltaSL

lengthframe <-data.frame(1:n, deltaBW)
colnames(lengthframe) <-c("gid", "length")
# Performing GWAS
GWAS_length <-GWAS(lengthframe, g,K=A, n.PC = 10)
# SNP markers with lowest P-values
GWAS_length$chrom[head(order(GWAS_length$length,decreasing = TRUE),3)]

```

```

# Corresponding position (base pair)
GWAS_length$pos[head(order(GWAS_length$length,decreasing = TRUE),3)]


Chrom<-GWAS_length$chrom
GWAS_length$chrom<-factor(Chrom,levels=c(as.character(1:22)))
options(repr.plot.width=8, repr.plot.height=4)
my_y_title <- expression(paste("-", "log"[10], "(", italic("p"), ") for deltaBW"))
p2<-ggplot(GWAS_length, aes(x=1:dim(x)[2], y=length, color=chrom)) +
  ggpibr::theme_pubr(base_family="Times New Roman",base_size = 12,border = TRUE) +
  geom_point() +
  scale_color_manual(values = rep(c("black", "skyblue"), 24 )) +
  labs(x="Physical order of SNPs", y=my_y_title) +
  theme(legend.position="none",
        axis.text.y=element_text(size=12),
        axis.text.x=element_text(size=12),
        text=element_text(size=12,
                          family="Times New Roman")) +
  geom_hline(yintercept=thred, linetype="dashed", color = "red") +
  geom_textt(aes(0,thred,label = thred, vjust = 1.2),color = "black",family="Times New Roman")

p2$theme$line$size<-line_th_unit
p2$theme$line$size<-line_th_unit
p2$theme$axis.ticks$size<-line_th_unit
p2$theme$axis.line$size<-line_th_unit
p2$theme$panel.grid$colour<-"black"
p2$theme$axis.ticks$colour<-"black"

p2

```

4.1.7 R script for phenotypes (Pop-CW)

```
R.Version()$version.string  
library("ggplot2")  
library("ggpubr")  
library("extrafont")  
  
print(paste("ggplot2 version",packageVersion("ggplot2")))  
print(paste("ggpubr version",packageVersion("ggpubr")))  
print(paste("extrafont version",packageVersion("extrafont")))  
# remotes::install_version("Rttf2pt1", version = "1.3.8")  
# ttf_import()  
# fonts()  
# loadfonts(device = "pdf")  
# extrafont::loadfonts(device = "postscript")  
  
pheno <- read.csv("./210301JST_reform.csv")  
# attach(pheno)  
# n<-length(pheno$BW)  
head(pheno)  
str(pheno)  
  
# Filration  
pheno_BCW_cleaned<-pheno[!is.na(pheno$BCW),]  
str(pheno_BCW_cleaned)  
  
data_p<-data.frame(SL1=pheno_BCW_cleaned$X201020_SL,  
                    BW1=pheno_BCW_cleaned$X201020_BW,  
                    tank_f=factor(pheno_BCW_cleaned$X201020_TANK),  
                    SL2=pheno_BCW_cleaned$X201201_SL,  
                    BW2=pheno_BCW_cleaned$X201201_BW,  
                    bcw=pheno_BCW_cleaned$BCW,  
                    HD=pheno_BCW_cleaned$BCW/(pheno_BCW_cleaned$X201201_SL)^2*100,  
                    egg=factor(pheno_BCW_cleaned$egg),  
                    human = factor(pheno_BCW_cleaned$Human),  
                    tank_I = factor(pheno_BCW_cleaned$X201020_TANK),  
                    deltaSL= 100*(pheno_BCW_cleaned$X201201_SL-  
                    pheno_BCW_cleaned$X201020_SL)/pheno_BCW_cleaned$X201020_SL,  
                    deltaBW= 100*(pheno_BCW_cleaned$X201201_BW-
```

```

pheno_BCW_cleaned$X201020_BW)/pheno_BCW_cleaned$X201020_BW)
attach(data_p)
str(data_p)

mean_sd_norm_round<-function(v){
  cbind(round(mean(v),3),
        round(shapiro.test(v)[2]$p.value,3),round(max(v),3),round(min(v),3))
}
re_sum<-c()
for (i in c(1,2,4,5,6,7,11,12)){
  re_sum<-rbind(re_sum,mean_sd_norm_round(data_p[i]))
}
row.names(re_sum)<-colnames(data_p)[c(1,2,4,5,6,7,11,12)]
colnames(re_sum)<-c("Ave","sd","p_norm","max","min")
re_sum

# Pearson's r
cor(data_p[,c(1,2,4,5,6,7,11,12)])

library(caret)
bctbcw<-BoxCoxTrans(bcw+1)
bctbcw
predictbcw<-predict(bctbcw, bcw+1) # ((bcw+1)^0.6-1)/0.6
# round(predictbcw,3)==round(((bcw+1)^0.6-1)/0.6,3)
data_p["predictbcw"]<-predictbcw

cor.test(data_p$predictbcw,data_p$SL1)

binw<-round(((1+1)^0.6-1)/0.6,3)
# Transformed H. okamotoi count
line_th_unit<-0.6 ##size unit, mm
scaleFUN <- function(x) sprintf("%.2f", x)
options(repr.plot.width=8, repr.plot.height=4)
p3<-ggplot(data_p, aes(x=predictbcw))+
  ggpibr::theme_pubr(base_family="Times New Roman",base_size = 12)+
  geom_histogram(aes(y = ..density..),color="black",fill="grey" ,binwidth = binw,size=line_th_unit)+
  geom_density(size=line_th_unit)+
  labs(x="Transformed HC", y="Density")

```

```

p3$theme$line$size<-line_th_unit
p3$theme$line$size<-line_th_unit
p3$theme$axis.ticks$size<-line_th_unit
p3$theme$axis.line$size<-line_th_unit
p3$theme$panel.grid$colour<-"black"
p3$theme$axis.ticks$colour<-"black"
p3
ggsave("Fig. 5. tbcw.pdf",device="pdf",units="in",width =6, height =3 )

# H. okamotoi count
line_th_unit<-0.6 ##size unit, mm
scaleFUN <- function(x) sprintf("% .2f", x)
options(repr.plot.width=8, repr.plot.height=4)
p1<-ggplot(data_p, aes(x=bew))+
  ggpibr::theme_pubr(base_family="Times New Roman",base_size = 12)+ 
  geom_histogram(aes(y = ..density..),color="black",fill="grey", binwidth=1,size=line_th_unit)+ 
  geom_density(size=line_th_unit)+ 
  labs(x="HC", y="Density")
p1$theme$line$size<-line_th_unit
p1$theme$line$size<-line_th_unit
p1$theme$axis.ticks$size<-line_th_unit
p1$theme$axis.line$size<-line_th_unit
p1$theme$panel.grid$colour<-"black"
p1$theme$axis.ticks$colour<-"black"

# Standard length
p2<-ggplot(data_p, aes(x=SL1))+ 
  ggpibr::theme_pubr(base_family="Times New Roman",base_size = 12)+ 
  geom_histogram(aes(y = ..density..),color="black",fill="grey", binwidth=0.1,size=line_th_unit)+ 
  labs(x="SL", y="Density")+
  geom_density(size=line_th_unit)+ 
  labs(x="SL", y="Density")
p2$theme$line$size<-line_th_unit
p2$theme$line$size<-line_th_unit
p2$theme$axis.ticks$size<-line_th_unit
p2$theme$axis.line$size<-line_th_unit
p2$theme$panel.grid $ colour<-"black"

```

```
p2$theme$axis.ticks$colour<- "black"
# p2

options(repr.plot.width=6, repr.plot.height=6)
figure <- ggpubr::ggarrange(p1, p2,
  labels = c("a", "b"),
  ncol = 1,
  nrow = 2,
  font.label = list(size = 14,
    color = "black",
    face = "plain",
    family = "Times New Roman"))

figure

ggsave("Fig. 5.1.pdf", device="pdf", units="in", width =6, height =6 )
# ggsave("Fig1.eps", device="eps", units="in", width =7.2)
# ggsave("Fig. 1.1.tiff", width=7.2, units="in", res=300)
```

4.1.8 Bash script for genotyping (Pop-CW)

```
#!/bin/bash

#sh TRIM.sh 1>TRIM.log 2>TRIM.err

TRIMMO=/home/lin/GS2018_SNPcalling/soft/Trimmomatic-0.39/trimmomatic-0.39.jar

DIR=/home/lin/GS2021_NGS/
FASTQ=${DIR}/Fastq
ls ${FASTQ} | sed -e "s/_\t/g" | cut -f1 | uniq -d | cut -f2 | sort -n > ${DIR}/NGSscript/sample.list
mkdir ${DIR}/TRIM

for i in `cat ${DIR}/NGSscript/sample.list`
do
mkdir ${DIR}/TRIM/${i}
TRIM=${DIR}/TRIM/${i}

java -jar ${TRIMMO} \
PE -threads 32 \
${FASTQ}/${i}_R1.fastq.gz ${FASTQ}/${i}_R2.fastq.gz \
${TRIM}/${i}"_paired_R1.fq.gz" ${TRIM}/${i}"_unpaired_R1.fq.gz" \
${TRIM}/${i}"_paired_R2.fq.gz" ${TRIM}/${i}"_unpaired_R2.fq.gz" \
ILLUMINACLIP:/home/lin/GS2018_SNPcalling/soft/Trimmomatic-0.39/adapters/NexteraPE-PE.fa:2:30:10
SLIDINGWINDOW:30:20 AVGQUAL:20

rm ${FASTQ}/${i}_R1.fastq.gz
rm ${FASTQ}/${i}_R2.fastq.gz
done

#!/bin/bash

#sh BWA.sh 1>BWA.log 2>BWA.err

DIR=/home/lin/GS2021_NGS
REF=/home/lin/GS2018_SNPcalling/reference/fr3.fa

#bwa index ${REF}
#samtools faidx ${REF}
```

```

cd ${DIR}
mkdir ${DIR}/BWA/

for i in `cat ${DIR}/NGSscript/sample.list`
do
mkdir ${DIR}/BWA/${i}
BWA=${DIR}/BWA/${i}
TRIM=${DIR}/TRIM/${i}

bwa mem -t 32 -M ${REF} ${TRIM}/${i}"_paired_R1.fq.gz" ${TRIM}/${i}"_paired_R2.fq.gz" -R
"@RG\tID:\"${i}\\"\tSM:\"${i}\\"\tPL:Illumina" | samtools view -b -F 2308 -q 10 -@ 32 -o ${BWA}/${i}.bam"
samtools sort -n -@ 32 -o ${BWA}/${i}_namesort.bam ${BWA}/${i}.bam
samtools fixmate -@ 32 -m ${BWA}/${i}_namesort.bam ${BWA}/${i}_fixmate.bam
samtools sort -@ 32 -o ${BWA}/${i}_sorted.bam ${BWA}/${i}_fixmate.bam
samtools index ${BWA}/${i}_sorted.bam

rm ${BWA}/${i}.sam
rm ${BWA}/${i}.bam
rm ${BWA}/${i}_namesort.bam
rm ${BWA}/${i}_fixmate.bam

done

#!/bin/bash
#sh GATK_GRAS_parallel.sh 1>>GATK_GRAS_parallel.log 2>>GATK_GRAS_parallel.err

#PBS -N gatk
#PBS -l select=1:ncpus=8:mem=15gb,walltime=2:00:00
#PBS -j oe

echo "START -----"

#module add java/12.0.2
#module load parallel
#module load gatk

```

```

DIR=/home/lin/GS2021_NGS
REF=/home/lin/GS2018_SNPeCalling/reference/fr3.fa
gatk=/home/lin/gatk-4.1.6.0/gatk

cd ${DIR}
mkdir VCF

### Use gnu-parallel to use multiple cores
### within one script
cat ${DIR}/NGSscript/sample.list | parallel --verbose -j 6 "${gatk}" HaplotypeCaller --output-mode
EMIT_ALL_CONFIDENT_SITES -stand-call-conf 30 -R ${REF} -I ${DIR}/BWA/{}_{}/{}_sorted.bam -O
${DIR}/VCF/{}_pe_variants_gatk.g.vcf.gz -ERC GVCF"

echo "FINISH -----"

## Bash

## Call variants per-sample (GVCF mode)

DIR=/home/lin/GS2021_NGS/
DIR2=/home/lin/GS2021_NGS/NGSscript
REF=/home/lin/GS2018_SNPeCalling/reference/fr3.fa
gatk=/home/lin/gatk-4.1.6.0/gatk

#rm ${DIR}/BWA/*/*_pe_variants_gatk.g.*
#cp ${DIR}/BWA/*/*_pe_variants_gatk.g.* ${DIR}/VCF
##sudo chmod +x *

echo "${gatk} CombineGVCFs --java-options "-Xmx100G"" > ${DIR2}/CombineGVCFs.sh
for i in `cat ${DIR2}/sample.list`
do
echo "-V ${DIR}/VCF/${i}_pe_variants_gatk.g.vcf.gz" >> ${DIR2}/CombineGVCFs.sh
done
echo "-R ${REF}" >> ${DIR2}/CombineGVCFs.sh
echo "-O ${DIR}/VCF/cohort.g.vcf.gz" >> ${DIR2}/CombineGVCFs.sh

cat ${DIR2}/CombineGVCFs.sh | tr "\n" " " > ${DIR2}/Spa_CombineGVCFs.sh

```

```

.${DIR2}/Spa_CombineGVCFs.sh

# tabix -p vcf ${DIR}/VCF/cohort.g.vcf.gz

${gatk}      GenotypeGVCFs      -R      ${REF}      -V      ${DIR}/VCF/cohort.g.vcf.gz      -O
${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --tmp-dir=${DIR}/VCF/temp

#bgzip -d ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz
#grep "##contig" ${DIR}/VCF/Output_jonit_call_cohort.vcf | sed -e "s/= /g" -e "s/,/ /g" | awk '{print $3}' >
${DIR}/VCF/pos.list
#cat -n ${DIR}/VCF/pos.list | awk '{print $2"\t"$1}' > ${DIR}/VCF/Chom.map
#bgzip ${DIR}/VCF/Output_jonit_call_cohort.vcf
tabix -p vcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz
#vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --min-meanDP 6 --max-meanDP 500 --max-
missing 0.3 --remove-indels --recode --stdout > ${DIR}/VCF/Hetero_realigned_cov10_filtered.vcf

## Line plot to visualize.
## --min-meanDP 6 to 10, and --max-missing 0-1
for i in $(seq 0 0.1 1)
do
echo $i >> ${DIR}/NGSscript/filter_parameter2.txt
done

#single --min-meanDP-5
(cat ${DIR}/NGSscript/filter_parameter2.txt | parallel --verbose --joblog parallel.log "vcftools --gzvcf
${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --chr chr1 --chr chr2 --chr chr3 --chr chr4 --chr chr5 --chr chr6 --
chr chr7 --chr chr8 --chr chr9 --chr chr10 --chr chr11 --chr chr12 --chr chr13 --chr chr14 --chr chr15 --chr chr16 -
--chr chr17 --chr chr18 --chr chr19 --chr chr20 --chr chr21 --chr chr22 --remove-indels --min-alleles 2 --max-
alleles 2 --minQ 10 --min-meanDP 5 --max-meanDP 500 --maf 0.01 --hwe 0.05 --max-missing {} " 2>&1) |tee
${DIR}/NGSscript/log.txt
cat ${DIR}/NGSscript/log.txt |grep possible|sed s/"After filtering, kept"/g |sed s/"out of a possible 2476352
Sites"/g |sort -n

```

9065

37331

37584

37731

37825

37898

37951

37997

38045

38123

38501

```
#single --min-meanDP-10
(cat ${DIR}/NGSscript/filter_parameter2.txt | parallel --verbose --joblog parallel.log "vcftools --gzvcf
${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --chr chr1 --chr chr2 --chr chr3 --chr chr4 --chr chr5 --chr chr6 --
chr chr7 --chr chr8 --chr chr9 --chr chr10 --chr chr11 --chr chr12 --chr chr13 --chr chr14 --chr chr15 --chr chr16 --
-chr chr17 --chr chr18 --chr chr19 --chr chr20 --chr chr21 --chr chr22 --remove-indels --min-alleles 2 --max-
alleles 2 --minQ 10 --min-meanDP 10 --max-meanDP 500 --maf 0.01 --hwe 0.05 --max-missing {} " 2>&1) |tee
${DIR}/NGSscript/log2.txt
cat ${DIR}/NGSscript/log.txt2 |grep possible|sed s/"After filtering, kept"/g |sed s/"out of a possible 2476352
Sites"/g |sort -n
```

9044

33076

33240

33315

33374

33412

33434

33453

33475

33513

33698

```
## SNP filtering and matrix
mkdir ${DIR}/Post_VCF_processing/
filter=${DIR}/Post_VCF_processing
```

```

task(){
mkdir ${filter}/$i;
vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --chr chr1 --chr chr2 --chr chr3 --chr chr4 --chr chr5 --chr chr6 --chr chr7 --chr chr8 --chr chr9 --chr chr10 --chr chr11 --chr chr12 --chr chr13 --chr chr14 --chr chr15 --chr chr16 --chr chr17 --chr chr18 --chr chr19 --chr chr20 --chr chr21 --chr chr22 --remove-indels --min-alleles 2 --max-alleles 2 --minQ 10 --min-meanDP 10 --max-meanDP 500 --maf 0.01 --hwe 0.05 --max-missing ${i} --recode --stdout > ${filter}/$i/Hetero_realigned_cov10_filtered.vcf;
zgrep -v "#" ${filter}/$i/Hetero_realigned_cov10_filtered.vcf | awk '{print $1"\t"$2}' > ${filter}/$i/position.list;
cat -n ${filter}/$i/position.list | awk '{print $2"\t"$1}' > ${filter}/$i/Chom.map;
bgzip ${filter}/$i/Hetero_realigned_cov10_filtered.vcf;
tabix -p vcf ${filter}/$i/Hetero_realigned_cov10_filtered.vcf.gz;
vcftools --gzvcf ${filter}/$i/Hetero_realigned_cov10_filtered.vcf.gz --plink --chrom-map ${filter}/$i/Chom.map --out ${filter}/$i/Hetero_realigned_cov10_filtered;
plink1 --ped ${filter}/$i/Hetero_realigned_cov10_filtered.ped --map ${filter}/$i/Hetero_realigned_cov10_filtered.map --recodeA --out ${filter}/$i/Hetero_realigned_cov10_filtered --noweb;
echo "empty" > ${filter}/$i/IDlist;
zgrep "#CHROM" ${filter}/$i/Hetero_realigned_cov10_filtered.vcf.gz | cut -f10- | sed -e "s/\t/\n/g" >> ${filter}/$i/IDlist;
sed -e "s/empty//g" ${filter}/$i/IDlist > ${filter}/$i/IDs.txt
cat ${filter}/$i/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- > ${filter}/$i/Hetero_realigned_cov10_filtered2.raw;
paste -d " " ${filter}/$i/IDs.txt ${filter}/$i/Hetero_realigned_cov10_filtered2.raw > ${filter}/$i/Hetero_realigned_cov10_filtered3.raw;
sed -n "1p" ${filter}/$i/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- | sed -e "s/ /\n/g" > ${filter}/$i/Hetero_realigned_cov10_filtered.pos;
less ${filter}/$i/Hetero_realigned_cov10_filtered.pos|sed -e "s/_/\t/g"|sed -e "s/:/\t/g"|sed -e "s/-/\t/g"|sed "s/"chr"/g"| awk '{print $1, $2}'>${filter}/$i/lin.map
}

# for i in $(seq 0 0.1 1);do task "$i" &done
i=0.7
task "$i"

```

```

Im_soft=/home/lin/GS2018_SNPealling/VCF_imputation/20200114_linkimpute
mkdir ${DIR}/VCF_imputation
imputation=${DIR}/VCF_imputation
## Imputation

```

```

taskIM(){
mkdir ${imputation}/${i}
java -jar ${Im_soft}/LinkImpute.jar ${filter}/${i}/Hetero_realigned_cov10_filtered.raw
${imputation}/${i}/Imputed.raw;
cat ${imputation}/${i}/Imputed.raw | cut -d " " -f7- >
${imputation}/${i}/Hetero_realigned_cov10_filtered2.raw;
cp ${filter}/${i}/IDs.txt ${imputation}/${i}/IDs.txt
cp ${filter}/${i}/lin.map ${imputation}/${i}/lin.map
paste -d " " ${imputation}/${i}/IDs.txt ${imputation}/${i}/Hetero_realigned_cov10_filtered2.raw >
${imputation}/${i}/Hetero_realigned_cov10_filtered3.raw;
}

taskIM "$i"

```

```

# Post_VCF_2wSNPs210922
mkdir ${DIR}/Post_VCF_2wSNPs210922
filter=${DIR}/Post_VCF_2wSNPs210922

task(){
mkdir ${filter}/${i};
vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --chr chr1 --chr chr2 --chr chr3 --chr chr4 --chr
chr5 --chr chr6 --chr chr7 --chr chr8 --chr chr9 --chr chr10 --chr chr11 --chr chr12 --chr chr13 --chr chr14 --chr
chr15 --chr chr16 --chr chr17 --chr chr18 --chr chr19 --chr chr20 --chr chr21 --chr chr22 --remove-indels --min-
alleles 2 --max-alleles 2 --minQ 20 --min-meanDP 20 --minDP 10 --max-meanDP 300 --maf 0.01 --hwe 0.05 --
max-missing ${i} --recode --stdout > ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf;
zgrep -v "#" ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf | awk '{print $1"\t"$2}' >
${filter}/${i}/position.list;
cat -n ${filter}/${i}/position.list | awk '{print $2"\t"$1}' > ${filter}/${i}/Chom.map;
bgzip ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf;
tabix -p vcf ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf.gz;
vcftools --gzvcf ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf.gz --plink --chrom-map
${filter}/${i}/Chom.map --out ${filter}/${i}/Hetero_realigned_cov10_filtered;
plink1 --ped ${filter}/${i}/Hetero_realigned_cov10_filtered.ped --map
${filter}/${i}/Hetero_realigned_cov10_filtered.map --recodeA --out
${filter}/${i}/Hetero_realigned_cov10_filtered --noweb;
echo "empty" > ${filter}/${i}/IDlist;
zgrep "#CHROM" ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf.gz | cut -f10- | sed -e "s/\t/\n/g" >>
${filter}/${i}/IDlist;

```

```

sed -e "s/empty//g" ${filter}/${i}/IDlist > ${filter}/${i}/IDs.txt
cat ${filter}/${i}/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- >
${filter}/${i}/Hetero_realigned_cov10_filtered2.raw;
paste -d " " ${filter}/${i}/IDs.txt ${filter}/${i}/Hetero_realigned_cov10_filtered2.raw >
${filter}/${i}/Hetero_realigned_cov10_filtered3.raw;
sed -n "1p" ${filter}/${i}/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- | sed -e "s/ \n/g" >
${filter}/${i}/Hetero_realigned_cov10_filtered.pos;
less ${filter}/${i}/Hetero_realigned_cov10_filtered.pos|sed -e "s/_/\t/g"|sed -e "s/:/\t/g"|sed -e "s/-/\t/g"|sed
"s/"chr"/\g"| awk '{print $1, $2}'>${filter}/${i}/lin.map
}
# for i in $(seq 0 0.1 1);do task "$i" &done
i=0.7
task "$i"

```

kept 21272 out of a possible 2476352 Sites

```

# imputaion
Im_soft=/home/lin/GS2018_SNPCalling/VCF_imputation/20200114_linkimpute
mkdir ${DIR}/VCF_imputation_210922
imputation=${DIR}/VCF_imputation_210922
taskIM(){
mkdir ${imputation}/${i}
java -jar ${Im_soft}/LinkImpute.jar ${filter}/${i}/Hetero_realigned_cov10_filtered.raw
${imputation}/${i}/Imputed.raw;
cat ${imputation}/${i}/Imputed.raw | cut -d " " -f7- >
${imputation}/${i}/Hetero_realigned_cov10_filtered2.raw;
cp ${filter}/${i}/IDs.txt ${imputation}/${i}/IDs.txt
cp ${filter}/${i}/lin.map ${imputation}/${i}/lin.map
paste -d " " ${imputation}/${i}/IDs.txt ${imputation}/${i}/Hetero_realigned_cov10_filtered2.raw >
${imputation}/${i}/Hetero_realigned_cov10_filtered3.raw;
}
taskIM "$i"

```

```

####
# Ne = 50.7
# vcf2genepop https://github.com/z0on/2bRAD_denovo/blob/master/vcf2genepop.pl

```

```

gunzip ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf.gz
/home/lin/GS2018_SNPcalling/Haplo_20210102/vcf2genepop.pl
vcf=${filter}/${i}/Hetero_realigned_cov10_filtered.vcf > ${filter}/filtered_Ne.gen
bgzip ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf

awk '/chr/ { print; print "pop"; next }1' ${filter}/filtered_Ne.gen >> ${filter}/filtered_Ne_pop.gen

fastqfile=~/GS2021_NGS/Fastq
statsdir=~/GS2021_NGS/seqstats_result

ls ${fastqfile} | cat |grep _R1.fastq.gz | parallel --verbose -j 6 "/home/lin/seqstats/seqstats ${fastqfile}{} | sed -n
1p|cut -f 2 >> ${statsdir}/raw_num_read_1.txt"

/home/lin/seqstats/seqstats ${fastqfile}{} |sed -n 2p|cut -f 2|cut -d' ' -f1 >> ${statsdir}/raw_bp_1.txt"

ls ${fastqfile} | cat |grep _R2.fastq.gz | parallel --verbose -j 6 "/home/lin/seqstats/seqstats ${fastqfile}{} |sed -n
1p|cut -f 2 >> ${statsdir}/raw_num_read_2.txt"

/home/lin/seqstats/seqstats ${fastqfile}{} |sed -n 2p|cut -f 2|cut -d' ' -f1 >> ${statsdir}/raw_bp_2.txt"

TRIMfile=~/GS2021_NGS/TRIM

ls ${TRIMfile} | cat | parallel --verbose -j 10 "/home/lin/seqstats/seqstats ${TRIMfile}{}/_paired_R1.fq.gz
|sed -n 1p|cut -f 2 >> ${statsdir}/trimed_num_read_1.txt"

/home/lin/seqstats/seqstats ${TRIMfile}{}/_paired_R1.fq.gz |sed -n 2p|cut -f 2|cut -d' ' -f1 >>
${statsdir}/trimed_bp_1.txt"

ls ${TRIMfile} | cat | parallel --verbose -j 10 "/home/lin/seqstats/seqstats ${TRIMfile}{}/_paired_R2.fq.gz
|sed -n 1p|cut -f 2 >> ${statsdir}/trimed_num_read_2.txt"

/home/lin/seqstats/seqstats ${TRIMfile}{}/_paired_R2.fq.gz |sed -n 2p|cut -f 2|cut -d' ' -f1 >>
${statsdir}/trimed_bp_2.txt"

```

BWAfile=~/GS2021_NGS/BWA

```

ls ${BWAfile} | cat | parallel --verbose -j 10 "samtools flagstat ${BWAfile}{}/_sorted.bam |sed -n 1p |sed -e
's/^\t/g' |cut -f 1 >> ${statsdir}/mapped_num_read.txt"

```

4.1.9 R script for genotyping (Pop-CW)

```
rb1<-read.table("raw_bp_1.txt")
rb2<-read.table("raw_bp_2.txt")
rn1<-read.table("raw_num_read_1.txt")
rn2<-read.table("raw_num_read_1.txt")

tb1<-read.table("trimed_bp_1.txt")
tb2<-read.table("trimed_bp_2.txt")
tn1<-read.table("trimed_num_read_1.txt")
tn2<-read.table("trimed_num_read_2.txt")

mn<-read.table("mapped_num_read.txt")

mean(unlist(rb1+rb2))
sd(unlist(rb1+rb2))

mean(unlist(rn1+rn2))
sd(unlist(rn1+rn2))

mean(unlist(tn1+tn2))
sd(unlist(tn1+tn2))

mean(unlist(tb1+tb2))
sd(unlist(tb1+tb2))

mean(unlist(mn))
sd(unlist(mn))

(sum(rb1)+sum(rb2))
(sum(tb1)+sum(tb2))
(sum(tb1)+sum(tb2))/(sum(rb1)+sum(rb2))

sum(tn1)+sum(tn2)
sum(tn2)

(sum(rn1)+sum(rn2))
(sum(tn1))*2
(sum(tn1))*2/(sum(rn1)+sum(rn2))
```

```
(sum(mn)/((sum(tn1))*2))  
mean(unlist(tn1+tn2))  
sd(unlist(tn1+tn2))  
  
mean(unlist(tn1+tn2))/mean(unlist(rn1+rn2))  
  
(sum(rn1)+sum(rn2))  
(sum(tn1)+sum(tn2))  
(sum(tn1)+sum(tn2))/(sum(rn1)+sum(rn2))  
  
(sum(rb1)+sum(rb2))/(sum(rn1)+sum(rn2))  
(sum(tb1)+sum(tb2))/(sum(tn1)+sum(tn2))
```

4.1.10 Python script for population structure (Pop-CW)

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm
import seaborn as sns; sns.set()
import matplotlib as mpl

# Loading data
pheno = pd.read_csv("./pheno_BCW_cleaned_ngs.csv")
#I = pd.read_table("./Hetero_realigned_cov10_filtered3.raw",sep="\s+")
I = pd.read_table("./reorderG.txt",sep="\s+")-1
bcw=pheno["BCW"]

from sklearn.manifold import TSNE
model = TSNE(random_state=0,perplexity=20)
tsne5 = model.fit_transform(I)

lin_wide=1.28
font_size=12
%matplotlib inline
%config InlineBackend.figure_format = 'svg'
plt.style.use('seaborn-white')
plt.style.use('seaborn-paper')
plt.rc('font',size=font_size)
plt.rc('lines',lw=lin_wide)
plt.rcParams['font.family'] = 'serif'
plt.rcParams['font.sans-serif'] = 'Times New Roman'
#plt.rcParams['axes.formatter.use_mathtext'] = True
#plt.rcParams['mathtext.bf'] = 'serif:normal'
#plt.rcParams['text.hinting'] = 'none'
#plt.rcParams['text.usetex'] = True
#plt.rcParams['axes.labelweight'] = 'normal'

plt.rcParams['axes.linewidth'] = lin_wide
plt.rcParams['xtick.major.width'] = lin_wide
plt.rcParams['xtick.minor.width'] = lin_wide
plt.rcParams['xtick.labelsize'] = font_size
```

```
plt.rcParams['ytick.major.width'] = lin_wide
plt.rcParams['ytick.minor.width'] = lin_wide
plt.rcParams['ytick.labelsize'] = font_size
plt.rcParams['axes.spines.bottom'] = True
# plt.style.use('bmh')
plt.scatter(tsne5[:, 0], tsne5[:, 1], edgecolor='k', c=bcw, cmap=cm.Reds, linewidth=lin_wide, s=s_value)
plt.xlabel("t-SNE coordinate 1", size=font_size)
plt.ylabel("t-SNE coordinate 2", size=font_size)
plt.subplots_adjust(bottom=0.1, right=0.8, top=0.9)
cax = plt.axes([0.85, 0.1, 0.075, 0.8])
plt.colorbar(cax=cax)
plt.gcf().set_size_inches(7, 6)
plt.savefig("C5.3_tSNE_HC.pdf", format="pdf")
plt.show()
# plt.savefig("PCA_diets.png", dpi=300)
```

4.1.11 R script for heritability and genetic correlation (Pop-CW)

```
reorderG<-read.table("GS2020/reorderG.txt")
x<-as.matrix(reorderG)-1
pheno<-read.csv("GS2020/pheno_BCW_cleaned_ngs.csv")
n<-dim(x)[1]
predictbcw<-((pheno$BCW+1)^0.6-1)/0.6

library("sommer")
packageVersion("sommer")
A<- A.mat(x,n.core=8)
row.names(A)=1:n;colnames(A)=1:n
data <- data.frame(tbcw=predictbcw,length=X201020_SL,id=factor(1:n))
attach(data)
ans.m <- mmer(cbind(tbcw,length)~1,random=~ vs(id, Gu=A, Gtc=unsm(2)),
               rcov=~ vs(units, Gtc=unsm(2)),
               data=data)
(sp_cor<-cov2cor(ans.m$sigma$`u:id`))
pin(ans.m, h2~ V1/(V1+V4))
pin(ans.m, h2~ V3/(V3+V6))
```

4.1.12 R script for GWAS (Pop-CW)

```
R.Version()$version.string  
library("ggplot2")  
library("ggpubr")  
library("extrafont")  
print(paste("ggplot2 version",packageVersion("ggplot2")))  
print(paste("ggpubr version",packageVersion("ggpubr")))  
print(paste("extrafont version",packageVersion("extrafont")))  
# remotes::install_version("Rttf2pt1", version = "1.3.8")  
# ttf_import()  
# fonts()  
# loadfonts(device = "pdf")  
  
# phenotype  
phenoRaw <- read.csv("./210301JST_reform.csv")  
# attach(pheno)  
# n<-length(pheno$BW)  
head(phenoRaw)  
str(phenoRaw)  
  
# Filtration  
pheno_BCW_cleaned<-phenoRaw[!is.na(phenoRaw$BCW),]  
str(pheno_BCW_cleaned)  
  
pheno<-data.frame(SL1=pheno_BCW_cleaned$X201020_SL,  
                   BW1=pheno_BCW_cleaned$X201020_BW,  
                   tank_f=factor(pheno_BCW_cleaned$X201020_TANK),  
                   SL2=pheno_BCW_cleaned$X201201_SL,  
                   BW2=pheno_BCW_cleaned$X201201_BW,  
                   bcw=pheno_BCW_cleaned$BCW,  
                   HD=pheno_BCW_cleaned$BCW/(pheno_BCW_cleaned$X201201_SL)^2*100,  
                   egg=factor(pheno_BCW_cleaned$egg),  
                   human = factor(pheno_BCW_cleaned$Human),  
                   tank_I = factor(pheno_BCW_cleaned$X201020_TANK),  
                   deltaSL= 100*(pheno_BCW_cleaned$X201201_SL-  
                   pheno_BCW_cleaned$X201020_SL)/pheno_BCW_cleaned$X201020_SL,  
                   deltaBW= 100*(pheno_BCW_cleaned$X201201_BW-  
                   pheno_BCW_cleaned$X201020_BW)/pheno_BCW_cleaned$X201020_BW)
```

```

n<-length(pheno$bcw)
attach(pheno)
str(pheno)

# Loading raw G matrix
geno<-read.table("./Imputation//Hetero_realigned_cov10_filtered3.raw", header=T, row.names=1)
dim(geno)

# reorder
id_NGS <- read.csv("./201204NGSid.csv",header=T )
pheno_BCW_cleaned_ngs<-cbind(pheno_BCW_cleaned,id_NGS$NGSID)
colnames(pheno_BCW_cleaned_ngs)[19]<-"NGSID"
# head(pheno_BCW_cleaned_ngs)
id_g <- read.csv("./Imputation//IDs.txt",header=F )
reorderIndex<-c()
for (i in pheno_BCW_cleaned_ngs$NGSID){
  reorderIndex<-c(reorderIndex,match(i,id_g$V1))
}
reorderG<-geno[reorderIndex,]
row.names(reorderG)=1:dim(geno)[1]
x <- as.matrix(reorderG)-1
head(reorderG)
str(x)

write.csv(pheno_BCW_cleaned_ngs,"pheno_BCW_cleaned_ngs.csv")
write.table(reorderG,"reorderG.txt")

# GBLUP
library(rrBLUP)
print(paste("rrBLUP version",packageVersion("rrBLUP")))

# genomic relationship matrix
A <- A.mat(x)
row.names(A)=1:n;colnames(A)=1:n

library(caret)
bctbcw<-BoxCoxTrans(bcw+1)

```

```

bctbcw

predictbcw<-predict(bctbcw, bcw+1) # ((bcw+1)^0.6-1)/0.6

# data
data <- data.frame(tbcw=predictbcw,length=pheno$SL1,SL2=pheno$SL2,
                     bcw_d_length2=pheno$HD,BW=pheno$BW1,BW2=pheno$BW2,
                     deltaSL=pheno$deltaSL, deltaBW=pheno$deltaBW,gid=1:n,x=x)
row.names(A)=1:n;colnames(A)=1:n

lin_map<-read.table("./Imputation/lin.map")
g <- data.frame(rownames(lin_map),lin_map$V1, lin_map$V2, t(x))
rownames(g) <-1:nrow(g)
colnames(g) <-c("marker", "chrom", "pos", rownames(x))
# Bonferroni-corrected significance threshold
(thred <- round(-log10(0.05/nrow(g)),3))

bcwframe <-data.frame(1:n, data$tbcw)
colnames(bcwframe) <-c("gid", "tbcw")
# Performing GWAS
GWAS_bcw <-GWAS(bcwframe, g,K=A, n.PC = 10)
# SNP markers with lowest P-values
GWAS_bcw$chrom[head(order(GWAS_bcw$tbcw,decreasing = TRUE),3)]
# Corresponding position (base pair)
GWAS_bcw$pos[head(order(GWAS_bcw$tbcw,decreasing = TRUE),3)]

g[(g$chrom==3)&(g$pos<(5531819+1000))&(g$pos>(5531819-1000)),]

table(unlist(g[(g$chrom==3)&(g$pos==5531819),][c(-1,-2,-3)]))

str(GWAS_bcw)

data["bcw"]<-pheno$bcw
mean(data$bcw)
mean(data[unlist(g[(g$chrom==3)&(g$pos==5531819),][c(-1,-2,-3)])==-1,]$bcw)
mean(data[unlist(g[(g$chrom==3)&(g$pos==5531819),][c(-1,-2,-3)])==0,]$bcw)
mean(data[unlist(g[(g$chrom==3)&(g$pos==5531819),][c(-1,-2,-3)])==1,]$bcw)
sd(data[unlist(g[(g$chrom==3)&(g$pos==5531819),][c(-1,-2,-3)])==1,]$bcw)

```

```

# top100
top100<-cbind(rep("GS2020",100),GWAS_bcw$chrom[head(order(GWAS_bcw$tbcw,decreasing = TRUE),100)],GWAS_bcw$pos[head(order(GWAS_bcw$tbcw,decreasing = TRUE),100)])
colname<-c("Exp","chr","pos")
write.csv(top100,"GS2020_GAWS_HC100.csv")

# Manhattan plot
line_th_unit<-0.6
Chrom<-GWAS_bcw$chrom
GWAS_bcw$chrom<-factor(Chrom,levels=c(as.character(1:22)))
options(repr.plot.width=8, repr.plot.height=4)
my_y_title<- expression(paste("-", "log"[10], "(", italic("p"), ") for HC"))
p1<-ggplot(GWAS_bcw, aes(x=1:dim(x)[2], y=tbcw, color=chrom))+
  ggpubr::theme_pubr(base_family="Times New Roman",base_size = 12,border = TRUE)+  

  geom_point()+
  scale_color_manual(values = rep(c("black", "skyblue"), 24 ))+
  labs(x="Physical order of SNPs", y=my_y_title)+  

  theme(legend.position="none",
        axis.text.y=element_text(size=12),
        axis.text.x=element_text(size=12),
        text=element_text(size=12,
                          family="Times New Roman"))+
  geom_hline(yintercept=thred, linetype="dashed", color = "red")+
  geom_text(aes(0,thred,label = thred, vjust = 1.2),color = "black",family="Times New Roman")
p1$theme$line$size<-line_th_unit
p1$theme$line$size<-line_th_unit
p1$theme$axis.ticks$size<-line_th_unit
p1$theme$axis.line$size<-line_th_unit
p1$theme$panel.grid$colour<-"black"
p1$theme$axis.ticks$colour<-"black"
p1

lengthframe <-data.frame(1:n, SL1)
colnames(lengthframe) <-c("gid", "length")
# Performing GWAS
GWAS_length <-GWAS(lengthframe, g,K=A, n.PC = 10)
# SNP markers with lowest P-values
GWAS_length$chrom[head(order(GWAS_length$length,decreasing = TRUE),3)]
```

```

# Corresponding position (base pair)
GWAS_length$pos[head(order(GWAS_length$length,decreasing = TRUE),3)]


# top100
top100<-cbind(rep("GS2020",100),GWAS_length$chrom[head(order(GWAS_length$length,decreasing = TRUE),100)],

GWAS_length$pos[head(order(GWAS_length$length,decreasing = TRUE),100)])

colname<-c("Exp","chr","pos")
write.csv(top100,"GS2020_GAWS_SL100.csv")

Chrom<-GWAS_length$chrom
GWAS_length$chrom<-factor(Chrom,levels=c(as.character(1:22)))
options(repr.plot.width=8, repr.plot.height=4)
my_y_title <- expression(paste("-", "log"[10], (" ", italic("p"), ") for SL"))
p2<-ggplot(GWAS_length, aes(x=1:dim(x)[2], y=length, color=chrom)) +
  ggpubr::theme_pubr(base_family="Times New Roman",base_size = 12,border = TRUE) +
  geom_point() +
  scale_color_manual(values = rep(c("black", "skyblue"), 24 )) +
  labs(x="Physical order of SNPs", y=my_y_title) +
  theme(legend.position="none",
        axis.text.y=element_text(size=12),
        axis.text.x=element_text(size=12),
        text=element_text(size=12,
                          family="Times New Roman")) +
  geom_hline(yintercept=thred, linetype="dashed", color = "red") +
  geom_text(aes(0,thred,label = thred, vjust = 1.2),color = "black",family="Times New Roman")
p2$theme$line$size<-line_th_unit
p2$theme$line$size<-line_th_unit
p2$theme$axis.ticks$size<-line_th_unit
p2$theme$axis.line$size<-line_th_unit
p2$theme$panel.grid$colour<-"black"
p2$theme$axis.ticks$colour<-"black"
p2

options(repr.plot.width=5, repr.plot.height=5)
figure <- ggpubr::ggarrange(p1, p2,
                            labels = c("a", "b"),
                            ncol = 1,

```

```

nrow = 2,
font.label = list(size = 14,
                  color = "black",
                  face = "plain",
                  family = "Times New Roman"))

# figure

ggsave("Fig. 5.2.pdf",device="pdf",units="in",width =6, height = 6 )
# ggsave("Fig2.eps",device="eps",units="in",width =7.2)
# ggsave("Fig. 1.2.tiff", width=7.2, units="in", res=300)

## GWAS for deltaSL

lengthframe <-data.frame(1:n, deltaSL)
colnames(lengthframe) <-c("gid", "length")
# Performing GWAS
GWAS_length <-GWAS(lengthframe, g,K=A, n.PC = 10)
# SNP markers with lowest P-values
GWAS_length$chrom[head(order(GWAS_length$length,decreasing = TRUE),3)]
# Corresponding position (base pair)
GWAS_length$pos[head(order(GWAS_length$length,decreasing = TRUE),3)]

Chrom<-GWAS_length$chrom
GWAS_length$chrom<-factor(Chrom,levels=c(as.character(1:22)))
options(repr.plot.width=8, repr.plot.height=4)
my_y_title <- expression(paste("-", "log"[10], "(", italic("p"), ") for deltaSL"))
p2<-ggplot(GWAS_length, aes(x=1:dim(x)[2], y=length, color=chrom)) +
  ggpubr::theme_pubr(base_family="Times New Roman",base_size = 12,border = TRUE) +
  geom_point() +
  scale_color_manual(values = rep(c("black", "skyblue"), 24 )) +
  labs(x="Physical order of SNPs", y=my_y_title) +
  theme(legend.position="none",
        axis.text.y=element_text(size=12),
        axis.text.x=element_text(size=12),
        text=element_text(size=12,
                          family="Times New Roman"))+

```

```

geom_hline(yintercept=thred, linetype="dashed", color = "red")+
  geom_text(aes(0,thred,label = thred, vjust = 1.2),color = "black",family="Times New Roman")
p2$theme$line$size<-line_th_unit
p2$theme$line$size<-line_th_unit
p2$theme$axis.ticks$size<-line_th_unit
p2$theme$axis.line$size<-line_th_unit
p2$theme$panel.grid$colour<-"black"
p2$theme$axis.ticks$colour<-"black"
p2

```

p2

GWAS for deltaSL

```

lengthframe <-data.frame(1:n, deltaBW)
colnames(lengthframe) <-c("gid", "length")
# Performing GWAS
GWAS_length <-GWAS(lengthframe, g,K=A, n.PC = 10)
# SNP markers with lowest P-values
GWAS_length$chrom[head(order(GWAS_length$length,decreasing = TRUE),3)]
# Corresponding position (base pair)
GWAS_length$pos[head(order(GWAS_length$length,decreasing = TRUE),3)]

```

```

Chrom<-GWAS_length$chrom
GWAS_length$chrom<-factor(Chrom,levels=c(as.character(1:22)))
options(repr.plot.width=8, repr.plot.height=4)
my_y_title <- expression(paste("-", "log"[10], "(", italic("p"), ") for deltaBW"))
p2<-ggplot(GWAS_length, aes(x=1:dim(x)[2], y=length, color=chrom)) +
  ggpibr::theme_pubr(base_family="Times New Roman", base_size = 12, border = TRUE) +
  geom_point() +
  scale_color_manual(values = rep(c("black", "skyblue"), 24 )) +
  labs(x="Physical order of SNPs", y=my_y_title) +
  theme(legend.position="none",
        axis.text.y=element_text(size=12),
        axis.text.x=element_text(size=12),
        text=element_text(size=12,
                          family="Times New Roman")) +
  geom_hline(yintercept=thred, linetype="dashed", color = "red")+

```

```
geom_text(aes(0,thred,label = thred, vjust = 1.2),color = "black",family="Times New Roman")
p2$theme$line$size<-line_th_unit
p2$theme$line$size<-line_th_unit
p2$theme$axis.ticks$size<-line_th_unit
p2$theme$axis.line$size<-line_th_unit
p2$theme$panel.grid$colour<-"black"
p2$theme$axis.ticks$colour<-"black"
p2
```

Section 4.2

4.2.1 R script for predictive abilities of genomic prediction (Pop-C)

```
# The same codes were used for loading genotypes, phenotypes and genomic relationship matrix in Section 4.1  
#BLUP  
# Parallel computation  
library(doParallel)  
library(foreach)  
cl<-makeCluster(16)  
  
# Parameters for cross validation  
repeats <- 10  
n.fold <- 5  
n.sample <- length(pheno$bcw)  
  
registerDoParallel(cl)  
system.time({  
  GBLUP<-foreach(j=1:repeats,.combine = "rbind") %do% {  
    set.seed(100+3*j+1)  
    id <- sample(1:n.sample %% n.fold) + 1  
    foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {  
      bcw_test <- data  
      bcw_test$tbcw[id == i] <- NA  
      res <- kin.blup(bcw_test, K=A, geno="gid", pheno="tbcw",covariate = "length")  
      cor(data$tbcw[id==i],res$pred[id==i])  
    }  
  }  
})  
stopImplicitCluster()  
  
registerDoParallel(cl)  
system.time({  
  GBLUP1<-foreach(j=1:repeats,.combine = "rbind") %do% {  
    set.seed(100+3*j+1)  
    id <- sample(1:n.sample %% n.fold) + 1  
    foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {  
      bcw_test <- data  
      bcw_test$tbcw[id == i] <- NA  
      res <- kin.blup(bcw_test, K=A, geno="gid", pheno="tbcw",fixed = "length")  
    }  
  }  
})
```

```

cor(data$tbcw[id==i],res$pred[id==i])
}
}
})
stopImplicitCluster()

mean(unlist(GLBLUP1))

registerDoParallel(cl)
system.time({
GLBLUP2<-foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {
    bcbw_test <- data
    bcbw_test$length[id == i] <- NA
    res <- kin.blup(bcbw_test, K=A, geno="gid", pheno="length")
    cor(data$length[id==i],res$pred[id==i])
  }
}
})
stopImplicitCluster()

registerDoParallel(cl)
system.time({
GLBLUP3<-foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {
    bcbw_test <- data
    bcbw_test$tbcw[id == i] <- NA
    res <- kin.blup(bcbw_test, K=A, geno="gid", pheno="tbcw")
    cor(data$tbcw[id==i],res$pred[id==i])
  }
}
})
stopImplicitCluster()

```

```

mean(unlist(GLBLUP3))

library("BGLR")
packageVersion("BGLR")

registerDoParallel(cl)
system.time({
BC1 <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$tbcw[id == i] <- NA
    fmBC=BGLR(y=bew_test$tbcw,
    ETA=list(
      mrk = list(X=x,model='BayesC'),
      fixed=list(~length,data=bew_test,model='FIXED')),
      nIter=10000,burnIn=2000,verbose = FALSE)
    #fmBC=BGLR(y=bew_test$tbcw,ETA=list(list(X=x,model='BayesC')),nIter=10000,burnIn=2000,verbose
    = FALSE)
    cor(data$tbcw[id == i],fmBC$yHat[id == i])
  }
}
})
stopImplicitCluster()

registerDoParallel(cl)
system.time({
BC2 <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$length[id == i] <- NA
    fmBC=BGLR(y=bew_test$length,ETA=list(list(X=x,model='BayesC')),nIter=10000,burnIn=2000,verbose
    = FALSE)
    cor(data$length[id == i],fmBC$yHat[id == i])
  }
}
})

```

```

        }
    })
stopImplicitCluster()

registerDoParallel(cl)
system.time({
BC3 <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$tbcw[id == i] <- NA
    fmBC=BGLR(y=bcw_test$tbcw,ETA=list(list(X=x,model='BayesC')),nIter=10000,burnIn=2000,verbose =
    FALSE)
    cor(data$tbcw[id == i],fmBC$yHat[id == i])
  }
}
})
stopImplicitCluster()

# Reproducing kernel
p<-ncol(x)
D<-(as.matrix(dist(x,method='euclidean'))^2)/p
h<-0.5;K<-exp(-h*D)
ETA<-list(list(K=K,model='RKHS'))

registerDoParallel(cl)
system.time({
RKHS1 <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$tbcw[id == i] <- NA
    fmRKHS=BGLR(y=bcw_test$tbcw,ETA=ETA,nIter=10000,burnIn=2000,verbose = FALSE)
    cor(data$tbcw[id == i],fmRKHS$yHat[id == i])
  }
}
})

```

```

  })
stopImplicitCluster()

registerDoParallel(cl)
system.time({
RKHS2 <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$length[id == i] <- NA
    fmRKHS=BGLR(y=bcw_test$length,ETA=ETA,nIter=10000,burnIn=2000,verbose = FALSE)
    cor(data$length[id == i],fmRKHS$yHat[id == i])
  }
}
})
stopImplicitCluster()

```

```

# Reproducing kernel
p<-ncol(x)
D<-(as.matrix(dist(x,method='euclidean'))^2)/p
h<-0.5;K<-exp(-h*D)

```

```

registerDoParallel(cl)
system.time({
RKHS3 <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$tbcw[id == i] <- NA
    fmRKHS=BGLR(y=bcw_test$tbcw,
      ETA=list(mrk=list(K=K,model='RKHS'),
      fixed=list(~length,data=bcw_test,model='FIXED')),
      nIter=10000,burnIn=2000,verbose = FALSE)
    cor(data$tbcw[id == i],fmRKHS$yHat[id == i])
  }
}
})

```

```

        }
    })
stopImplicitCluster()

# data
Acc<-data.frame(unlist(GLBLUP),unlist(BC1),unlist(RKHS3))
colnames(Acc)<-c("GLBLUP_HC","BC_HC","RKHS3_HC")
# Predictive ability
(summary<-data.frame(PA_mean=sapply(Acc,function(x) round(mean(x),digits = 3)),
PA_SE=sapply(Acc,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3)))))

# data
Acc<-data.frame(unlist(GLBLUP),unlist(BC1),unlist(RKHS1),unlist(GLBLUP2),unlist(BC2),unlist(RKHS2))
colnames(Acc)<-c("GLBLUP_HC","BayesC_HC","RKHS_HC", "GLBLUP_SL","BayesC_SL","RKHS_SL")
# Predictive ability
(summary<-data.frame(PA_mean=sapply(Acc,function(x) round(mean(x),digits = 3)),
PA_SE=sapply(Acc,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3)))))

# data
Acc<-data.frame(unlist(GLBLUP3),unlist(BC3),unlist(RKHS1),unlist(GLBLUP2),unlist(BC2),unlist(RKHS2))
colnames(Acc)<-c("GLBLUP_HC","BayesC_HC","RKHS_HC", "GLBLUP_SL","BayesC_SL","RKHS_SL")
# Predictive ability
(summary<-data.frame(PA_mean=sapply(Acc,function(x) round(mean(x),digits = 3)),
PA_SE=sapply(Acc,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3)))))

# Accuracy
h2_HC = 0.472
Acc1<-data.frame(GLBLUP_HC=Acc[,c(1,2,3)])
(summary<-data.frame(Acc_mean=sapply(Acc1,function(x) round(mean(x/sqrt(h2_HC)),digits = 3)),
Acc_SE=sapply(Acc1,function(x) round(sd(x/sqrt(h2_HC))/sqrt(repeats*n.fold),digits = 3))))
h2_SL = 0.620
Acc2<-data.frame(GLBLUP_SL=Acc[,c(4,5,6)])
(summary<-data.frame(Acc_mean=sapply(Acc2,function(x) round(mean(x/sqrt(h2_SL)),digits = 3)),
Acc_SE=sapply(Acc2,function(x) round(sd(x/sqrt(h2_SL))/sqrt(repeats*n.fold),digits = 3))))
```

4.2.2 R script for predictive abilities of genomic prediction (Pop-CW)

```
# The same codes were used for loading genotypes, phenotypes and genomic relationship matrix in Section 4.1  
#BLUP  
# Parallel computation  
library(doParallel)  
library(foreach)  
cl<-makeCluster(16)  
  
# Parameters for cross validation  
repeats <- 10  
n.fold <- 5  
n.sample <- length(pheno$bcw)  
  
registerDoParallel(cl)  
system.time({  
  GBLUP<-foreach(j=1:repeats,.combine = "rbind") %do% {  
    set.seed(100+3*j+1)  
    id <- sample(1:n.sample %% n.fold) + 1  
    foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {  
      bcw_test <- data  
      bcw_test$tbcw[id == i] <- NA  
      res <- kin.blup(bcw_test, K=A, geno="gid", pheno="tbcw",covariate = "length")  
      cor(data$tbcw[id==i],res$pred[id==i])  
    }  
  }  
})  
stopImplicitCluster()  
  
registerDoParallel(cl)  
system.time({  
  GBLUP1<-foreach(j=1:repeats,.combine = "rbind") %do% {  
    set.seed(100+3*j+1)  
    id <- sample(1:n.sample %% n.fold) + 1  
    foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {  
      bcw_test <- data  
      bcw_test$tbcw[id == i] <- NA  
      res <- kin.blup(bcw_test, K=A, geno="gid", pheno="tbcw",fixed = "length")  
      cor(data$tbcw[id==i],res$pred[id==i])  
    }  
  }  
})
```

```

        }
    }
})

stopImplicitCluster()

registerDoParallel(cl)
system.time({
  GBLUP2<-foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {
      bcw_test <- data
      bcw_test$length[id == i] <- NA
      res <- kin.blup(bcw_test, K=A, geno="gid", pheno="length")
      cor(data$length[id==i],res$pred[id==i])
    }
  }
})
stopImplicitCluster()

registerDoParallel(cl)
system.time({
  GBLUP3<-foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {
      bcw_test <- data
      bcw_test$tbcw[id == i] <- NA
      res <- kin.blup(bcw_test, K=A, geno="gid", pheno="tbcw")
      cor(data$tbcw[id==i],res$pred[id==i])
    }
  }
})
stopImplicitCluster()

mean(unlist(GBLUP3))
sd(unlist(GBLUP3))

```

```

library("BGLR")
packageVersion("BGLR")

registerDoParallel(cl)
system.time({
BC1 <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$tbcw[id == i] <- NA
    fmBC=BGLR(y=bcw_test$tbcw,
    ETA=list(
      mrk = list(X=x,model='BayesC'),
      fixed=list(~length,data=bcw_test,model='FIXED')),
      nIter=10000,burnIn=2000,verbose = FALSE)
    #fmBC=BGLR(y=bcw_test$tbcw,ETA=list(list(X=x,model='BayesC')),nIter=10000,burnIn=2000,verbose
    = FALSE)
    cor(data$tbcw[id == i],fmBC$yHat[id == i])
  }
}
})
stopImplicitCluster()

registerDoParallel(cl)
system.time({
BC2 <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$length[id == i] <- NA
    fmBC=BGLR(y=bcw_test$length,ETA=list(list(X=x,model='BayesC')),nIter=10000,burnIn=2000,verbose
    = FALSE)
    cor(data$length[id == i],fmBC$yHat[id == i])
  }
}
})

```

```

stopImplicitCluster()

registerDoParallel(cl)
system.time({
  BC3 <- foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="BGLR") %dopar% {
      bcw_test <- data
      bcw_test$tbcw[id == i] <- NA
      fmBC=BGLR(y=bcw_test$tbcw,ETA=list(list(X=x,model='BayesC')),nIter=10000,burnIn=2000,verbose =
      FALSE)
      cor(data$tbcw[id == i],fmBC$yHat[id == i])
    }
  }
})
stopImplicitCluster()

# Reproducing kernel
p<-ncol(x)
D<-(as.matrix(dist(x,method='euclidean'))^2)/p
h<-0.5;K<-exp(-h*D)
ETA<-list(list(K=K,model='RKHS'))

registerDoParallel(cl)
system.time({
  RKHS1 <- foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="BGLR") %dopar% {
      bcw_test <- data
      bcw_test$tbcw[id == i] <- NA
      fmRKHS=BGLR(y=bcw_test$tbcw,ETA=ETA,nIter=10000,burnIn=2000,verbose = FALSE)
      cor(data$tbcw[id == i],fmRKHS$yHat[id == i])
    }
  }
})
stopImplicitCluster()

```

```

registerDoParallel(cl)
system.time({
RKHS2 <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$length[id == i] <- NA
    fmRKHS=BGLR(y=bcw_test$length,ETA=ETA,nIter=10000,burnIn=2000,verbose = FALSE)
    cor(data$length[id == i],fmRKHS$yHat[id == i])
  }
}
})
stopImplicitCluster()

# Reproducing kernel
p<-ncol(x)
D<-(as.matrix(dist(x,method='euclidean'))^2)/p
h<-0.5;K<-exp(-h*D)

registerDoParallel(cl)
system.time({
RKHS3 <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$tbcw[id == i] <- NA
    fmRKHS=BGLR(y=bcw_test$tbcw,
      ETA=list(mrk=list(K=K,model='RKHS'),
      fixed=list(~length,data=bcw_test,model='FIXED')),
      nIter=10000,burnIn=2000,verbose = FALSE)
    cor(data$tbcw[id == i],fmRKHS$yHat[id == i])
  }
}
})

```

```

stopImplicitCluster()

# data
Acc<-data.frame(unlist(GLBLUP),unlist(BC1),unlist(RKHS3))
colnames(Acc)<-c("GLBLUP_HC","BC_HC","RKHS3_HC")
# Predictive ability
(summary<-data.frame(PA_mean=sapply(Acc,function(x) round(mean(x),digits = 3)),
PA_SE=sapply(Acc,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3)))))

# data
Acc<-data.frame(unlist(GLBLUP),unlist(BC1),unlist(RKHS1),unlist(GLBLUP2),unlist(BC2),unlist(RKHS2))
colnames(Acc)<-c("GLBLUP_HC","BayesC_HC","RKHS_HC", "GLBLUP_SL","BayesC_SL","RKHS_SL")
# Predictive ability
(summary<-data.frame(PA_mean=sapply(Acc,function(x) round(mean(x),digits = 3)),
PA_SE=sapply(Acc,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3)))))

# data
Acc<-data.frame(unlist(GLBLUP3),unlist(BC3),unlist(RKHS1),unlist(GLBLUP2),unlist(BC2),unlist(RKHS2))
colnames(Acc)<-c("GLBLUP_HC","BayesC_HC","RKHS_HC", "GLBLUP_SL","BayesC_SL","RKHS_SL")
# Predictive ability
(summary<-data.frame(PA_mean=sapply(Acc,function(x) round(mean(x),digits = 3)),
PA_SE=sapply(Acc,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3)))))

# Accuracy
h2_HC = 0.569
Acc1<-data.frame(GLBLUP_HC=Acc[,c(1,2,3)])
(summary<-data.frame(Acc_mean=sapply(Acc1,function(x) round(mean(x/sqrt(h2_HC)),digits = 3)),
Acc_SE=sapply(Acc1,function(x) round(sd(x/sqrt(h2_HC))/sqrt(repeats*n.fold),digits = 3))))
h2_SL = 0.669
Acc2<-data.frame(GLBLUP_SL=Acc[,c(4,5,6)])
(summary<-data.frame(Acc_mean=sapply(Acc2,function(x) round(mean(x/sqrt(h2_SL)),digits = 3)),
Acc_SE=sapply(Acc2,function(x) round(sd(x/sqrt(h2_SL))/sqrt(repeats*n.fold),digits = 3))))

```

Section 5.1

5.1.1 R script for tested population and low fish meal diet treatment

```
R.Version()$version.string
```

```
library("ggplot2")
```

```
library("ggpubr")
```

```
library("extrafont")
```

```
print(paste("ggplot2 version",packageVersion("ggplot2")))
```

```
print(paste("ggpubr version",packageVersion("ggpubr")))
```

```
print(paste("extrafont version",packageVersion("extrafont")))
```

```
# remotes::install_version("Rttf2pt1", version = "1.3.8")
```

```
# ttf_import()
```

```
# fonts()
```

```
# loadfonts(device = "pdf")
```

```
# extrafont::loadfonts(device = "postscript")
```

```
phenoRaw <- read.csv("./200818JST210113.csv")
```

```
# attach(pheno)
```

```
str(phenoRaw)
```

```
# Filtration
```

```
pheno_cleaned<-phenoRaw[-740,]
```

```
str(pheno_cleaned)
```

```
pheno<-data.frame(
```

```
  SL1 = pheno_cleaned$X200107_SL,
```

```
  SL2 = pheno_cleaned$X2005SL,
```

```
  SL3 = pheno_cleaned$X200818_SL,
```

```
  SL4 = as.numeric(pheno_cleaned$X210113_SL),
```

```
  gender = as.character(pheno_cleaned$X210113_sex),
```

```
  original_tank = as.character(pheno_cleaned$Original_tank),
```

```
  feeding_tank = as.character(pheno_cleaned$Feeding_tank)
```

```
)
```

```
n<-length(pheno$SL1)
```

```
attach(pheno)
```

```
str(pheno)
```

```
# General statistics
```

```
mean_sd_norm_round<-function(v){
```

```

cbind(round(mean(v),2),round(sd(v),2),
round(shapiro.test(v)[2]$p.value,3),round(max(v),3),round(min(v),3))
}

re_sum<-c()
for (i in c(1,2,3,4)){
  re_sum<-rbind(re_sum,mean_sd_norm_round(pheno[,i]))
}
row.names(re_sum)<-colnames(pheno)[c(1,2,3,4)]
colnames(re_sum)<-c("Ave","sd","p_norm","max","min")
re_sum

# Pearson's r
cor(pheno[,c(1,2,3,4)])

cor_p<-function(v1,v2){
  return(cor.test(v1,v2)$p.value)
}

cor_p(SL1,SL2)
cor_p(SL1,SL3)
cor_p(SL1,SL4)
cor_p(SL2,SL3)
cor_p(SL2,SL4)
cor_p(SL3,SL4)

# Histogram
# SL1
# binw<-round(((1+1)^0.6-1)/0.6,3)
line_th_unit<-0.6 ##size unit, mm
scaleFUN <- function(x) sprintf("% .2f", x)
options(repr.plot.width=8, repr.plot.height=4)
p1<-ggplot(pheno, aes(x=SL1))+
  ggpibr::theme_pubr(base_family="Times New Roman",base_size = 12)+
  geom_histogram(aes(y = ..density..),color="black",fill="grey" ,binwidth=0.1,size=line_th_unit)+
  geom_density(size=line_th_unit)+xlim(18,42)+
  labs(x="SL1", y="Density")
p1$theme$line$size<-line_th_unit
p1$theme$line$size<-line_th_unit

```

```

p1$theme$axis.ticks$size<-line_th_unit
p1$theme$axis.line$size<-line_th_unit
p1$theme$panel.grid$colour<-"black"
p1$theme$axis.ticks$colour<-"black"

# SL2

p2<-ggplot(pheno, aes(x=SL2))+
  ggpubr::theme_pubr(base_family="Times New Roman",base_size = 12)+
  geom_histogram(aes(y = ..density..),color="black",fill="grey", binwidth=0.1,size=line_th_unit)+
  geom_density(size=line_th_unit)+xlim(18,42)+
  labs(x="SL2", y="Density")

p2$theme$line$size<-line_th_unit
p2$theme$line$size<-line_th_unit
p2$theme$axis.ticks$size<-line_th_unit
p2$theme$axis.line$size<-line_th_unit
p2$theme$panel.grid $ colour<-"black"
p2$theme$axis.ticks$colour<-"black"

# p2

# SL3

line_th_unit<-0.6 ##size unit, mm
scaleFUN <- function(x) sprintf("%.2f", x)
options(repr.plot.width=8, repr.plot.height=4)

p3<-ggplot(pheno, aes(x=SL3))+
  ggpubr::theme_pubr(base_family="Times New Roman",base_size = 12)+
  geom_histogram(aes(y = ..density..),color="black",fill="grey" ,binwidth = 0.1,size=line_th_unit)+
  geom_density(size=line_th_unit)+xlim(18,42)+
  labs(x="SL3", y="Density")

p3$theme$line$size<-line_th_unit
p3$theme$line$size<-line_th_unit
p3$theme$axis.ticks$size<-line_th_unit
p3$theme$axis.line$size<-line_th_unit
p3$theme$panel.grid$colour<-"black"
p3$theme$axis.ticks$colour<-"black"

# SL4

line_th_unit<-0.6 ##size unit, mm
scaleFUN <- function(x) sprintf("%.2f", x)

```

```

options(repr.plot.width=8, repr.plot.height=4)
p4<-ggplot(pheno, aes(x=SL4))+  

  ggpubr::theme_pubr(base_family="Times New Roman",base_size = 12)+  

  geom_histogram(aes(y = ..density..),color="black",fill="grey" ,binwidth = 0.1,size=line_th_unit)+  

  geom_density(size=line_th_unit)+xlim(18,42)+  

  labs(x="SL4", y="Density")  

p4$theme$line$size<-line_th_unit  

p4$theme$line$size<-line_th_unit  

p4$theme$axis.ticks$size<-line_th_unit  

p4$theme$axis.line$size<-line_th_unit  

p4$theme$panel.grid$colour<-"black"  

p4$theme$axis.ticks$colour<-"black"

options(repr.plot.width=6, repr.plot.height=8)
figure <- ggpubr::ggarrange(p1, p2, p3, p4,
  labels = c("a", "b", "c", "d"),
  ncol = 1,
  nrow = 4,
  font.label = list(size = 14,
    color = "black",
    face = "plain",
    family = "Times New Roman"))

figure
ggsave("Fig.pdf",device="pdf",units="in",width =8, height =12 )

```

5.1.2 Bash script for genotyping

```
#!/bin/bash

#sh TRIM.sh 1>TRIM.log 2>TRIM.err

TRIMMO=/home/lin/GS2018_SNPcalling/soft/Trimmomatic-0.39/trimmomatic-0.39.jar

DIR=/home/lin/2020AutophenotypingNGS
FASTQ=${DIR}/JP-MUS-20200612-01A/Fastq
ls ${FASTQ} | sed -e "s/_\t/g" | cut -f3 | uniq -d > ${DIR}/script/sample1.list

for i in `cat ${DIR}/script/sample1.list`
do
mkdir ${DIR}/TRIM/${i}
TRIM=${DIR}/TRIM/${i}

trim_galore \
--fastqc --gzip --length 100 -o ${TRIM} --paired --stringency 10 --cores 12 \
${FASTQ}/JST_GT_${i}_R1.fastq.gz ${FASTQ}/JST_GT_${i}_R2.fastq.gz
done

# hint for adapter sequence
# https://support.illumina.com/bulletins/2016/12/what-sequences-do-i-use-for-adapter-trimming.html

for i in `cat ${DIR}/script/sample1.list`
do
mkdir ${DIR}/TRIM/${i}
TRIM=${DIR}/TRIM/${i}

trim_galore \
--fastqc --gzip --length 100 -o ${TRIM} --paired --2colour 20 --stringency 10 --cores 12 \
-a CTGTCTTATACACATCT \
${FASTQ}/JST_GT_${i}_R1.fastq.gz ${FASTQ}/JST_GT_${i}_R2.fastq.gz

done

##
```

```

for i in `cat ${DIR}/script/sample1.list`
do
mkdir ${DIR}/TRIM/${i}
TRIM=${DIR}/TRIM/${i}
java -jar ${TRIMMO} \
PE -threads 24 \
${FASTQ}/*${i}_R1.fastq.gz ${FASTQ}/*${i}_R2.fastq.gz \
${TRIM}/${i}"_paired_R1.fq.gz" ${TRIM}/${i}"_unpaired_R1.fq.gz" \
${TRIM}/${i}"_paired_R2.fq.gz" ${TRIM}/${i}"_unpaired_R2.fq.gz" \
ILLUMINACLIP:/home/lin/GS2018_SNPCalling/soft/Trimmomatic-0.39/adapters/novaseq6000-SE.fa:2:30:10
SLIDINGWINDOW:30:20 AVGQUAL:20 MINLEN:100
done

```

```

#adapter sequence 'CTGTCTCTTATACACATCT' in novaseq6000.fa file.
## Truseq3-PE.fa, NexteraPE-PE.fa, TruSeq2-PE.fa, TruSeq3-PE-2.fa TruSeq3-SE.fa fail

```

```

#!/bin/bash
#sh TRIM.sh 1>TRIM.log 2>TRIM.err

```

```
TRIMMO=/home/lin/GS2018_SNPCalling/soft/Trimmomatic-0.39/trimmomatic-0.39.jar
```

```

DIR=/home/lin/2020AutophenotypingNGS
FASTQ=${DIR}/JP-MUS-20200612-01B/Fastq
ls ${FASTQ} | sed -e "s/_\t/g" | cut -f3 | uniq -d > ${DIR}/script/sample2.list

```

```

for i in `cat ${DIR}/script/sample2.list`
do
mkdir ${DIR}/TRIM/${i}
TRIM=${DIR}/TRIM/${i}

trim_galore \
--fastqc --gzip --length 100 -o ${TRIM} --paired --stringency 10 --cores 12 \
${FASTQ}/JST_GT_${i}_R1.fastq.gz ${FASTQ}/JST_GT_${i}_R2.fastq.gz

# hint for adapter sequencce
# https://support.illumina.com/bulletins/2016/12/what-sequences-do-i-use-for-adapter-trimming.html

```

```
done
```

```
for i in `cat ${DIR}/script/sample1.list`  
do  
mkdir ${DIR}/TRIM/${i}  
TRIM=${DIR}/TRIM/${i}  
  
trim_galore \  
--fastqc --gzip --length 100 -o ${TRIM} --paired --2colour 20 --stringency 10 --cores 12 \  
-a CTGTCTCTTATACACATCT \  
${FASTQ}/JST_GT_${i}_R1.fastq.gz ${FASTQ}/JST_GT_${i}_R2.fastq.gz
```

```
done
```

```
##  
for i in `cat ${DIR}/script/sample1.list`  
do  
mkdir ${DIR}/TRIM/${i}  
TRIM=${DIR}/TRIM/${i}  
java -jar ${TRIMMO} \  
PE -threads 24 \  
${FASTQ}/*${i}_R1.fastq.gz ${FASTQ}/*${i}_R2.fastq.gz \  
${TRIM}/${i}"_paired_R1.fq.gz" ${TRIM}/${i}"_unpaired_R1.fq.gz" \  
${TRIM}/${i}"_paired_R2.fq.gz" ${TRIM}/${i}"_unpaired_R2.fq.gz" \  
ILLUMINACLIP:/home/lin/GS2018_SNPalignment/soft/Trimmomatic-0.39/adapters/novaseq6000-SE.fa:2:30:10  
SLIDINGWINDOW:30:20 AVGQUAL:20 MINLEN:100  
done
```

```
#adapter sequence 'CTGTCTCTTATACACATCT' in novaseq6000.fa file.  
## Truseq3-PE.fa, NexteraPE-PE.fa, TruSeq2-PE.fa, TruSeq3-PE-2.fa TruSeq3-SE.fa fail
```

```
#!/bin/bash  
#sh TRIM.sh 1>>TRIM.log 2>>TRIM.err
```

```
TRIMMO=/home/lin/GS2018_SNPalignment/soft/Trimmomatic-0.39/trimmomatic-0.39.jar
```

```
DIR=/home/lin/2020AutophenotypingNGS  
FASTQ=${DIR}/JP-MUS-20200612-01C/Fastq
```

```
ls ${FASTQ} | sed -e "s/_\t/g" | cut -f3 | uniq -d > ${DIR}/script/sample3.list
```

```
for i in `cat ${DIR}/script/sample3.list`  
do  
mkdir ${DIR}/TRIM/${i}  
TRIM=${DIR}/TRIM/${i}
```

```
trim_galore \  
--fastqc --gzip --length 100 -o ${TRIM} --paired --stringency 10 --cores 12 \  
${FASTQ}/JST_GT_${i}_R1.fastq.gz ${FASTQ}/JST_GT_${i}_R2.fastq.gz
```

```
# hint for adapter sequcence  
# https://support.illumina.com/bulletins/2016/12/what-sequences-do-i-use-for-adapter-trimming.html  
done
```

```
for i in `cat ${DIR}/script/sample1.list`  
do  
mkdir ${DIR}/TRIM/${i}  
TRIM=${DIR}/TRIM/${i}
```

```
trim_galore \  
--fastqc --gzip --length 100 -o ${TRIM} --paired --2colour 20 --stringency 10 --cores 12 \  
-a CTGTCTCTTATACACATCT \  
${FASTQ}/JST_GT_${i}_R1.fastq.gz ${FASTQ}/JST_GT_${i}_R2.fastq.gz
```

```
done
```

```
##  
for i in `cat ${DIR}/script/sample1.list`  
do  
mkdir ${DIR}/TRIM/${i}  
TRIM=${DIR}/TRIM/${i}  
java -jar ${TRIMMO} \  
PE -threads 24 \  
${FASTQ}/*${i}_R1.fastq.gz ${FASTQ}/*${i}_R2.fastq.gz \  
${TRIM}/${i}"_paired_R1.fq.gz" ${TRIM}/${i}"_unpaired_R1.fq.gz" \  
${TRIM}/${i}"_paired_R2.fq.gz" ${TRIM}/${i}"_unpaired_R2.fq.gz"
```

```
 ${TRIM}"/${i}"_paired_R2.fq.gz" ${TRIM}"/${i}"_unpaired_R2.fq.gz" \
ILLUMINACLIP:/home/lin/GS2018_SNPCalling/soft/Trimmomatic-0.39/adapters/novaseq6000-SE.fa:2:30:10
SLIDINGWINDOW:30:20 AVGQUAL:20 MINLEN:100
done
```

```
#adapter sequence 'CTGTCTCTTATACACATCT' in novaseq6000.fa file.
## Truseq3-PE.fa, NexteraPE-PE.fa, TruSeq2-PE.fa, TruSeq3-PE-2.fa TruSeq3-SE.fa fail
```

```
#!/bin/bash
#sh TRIM.sh 1>>TRIM.log 2>>TRIM.err
```

```
TRIMMO=/home/lin/GS2018_SNPCalling/soft/Trimmomatic-0.39/trimmomatic-0.39.jar
```

```
DIR=/home/lin/2020AutophenotypingNGS
FASTQ=${DIR}/JP-MUS-20200612-01D/Fastq
ls ${FASTQ} | sed -e "s/_\t/g" | cut -f3 | uniq -d > ${DIR}/script/sample4.list
```

```
for i in `cat ${DIR}/script/sample4.list`
do
mkdir ${DIR}/TRIM/${i}
TRIM=${DIR}/TRIM/${i}

trim_galore \
--fastqc --gzip --length 100 -o ${TRIM} --paired --stringency 10 --cores 12 \
${FASTQ}/JST_GT_${i}_R1.fastq.gz ${FASTQ}/JST_GT_${i}_R2.fastq.gz
```

```
# hint for adapter sequcence
# https://support.illumina.com/bulletins/2016/12/what-sequences-do-i-use-for-adapter-trimming.html
done
```

```
for i in `cat ${DIR}/script/sample1.list`
do
mkdir ${DIR}/TRIM/${i}
TRIM=${DIR}/TRIM/${i}

trim_galore \
```

```
--fastqc --gzip --length 100 -o ${TRIM} --paired --2colour 20 --stringency 10 --cores 12 \
-a CTGTCTCTTATACACATCT \
${FASTQ}/JST_GT_${i}_R1.fastq.gz ${FASTQ}/JST_GT_${i}_R2.fastq.gz
```

done

```
##  
for i in `cat ${DIR}/script/sample1.list`  
do  
mkdir ${DIR}/TRIM/${i}  
TRIM=${DIR}/TRIM/${i}  
java -jar ${TRIMMO} \  
PE -threads 24 \  
${FASTQ}/*${i}_R1.fastq.gz ${FASTQ}/*${i}_R2.fastq.gz \  
${TRIM}/${i}"_paired_R1.fq.gz" ${TRIM}/${i}"_unpaired_R1.fq.gz" \  
${TRIM}/${i}"_paired_R2.fq.gz" ${TRIM}/${i}"_unpaired_R2.fq.gz" \  
ILLUMINACLIP:/home/lin/GS2018_SNPCalling/soft/Trimmomatic-0.39/adapters/novaseq6000-SE.fa:2:30:10  
SLIDINGWINDOW:30:20 AVGQUAL:20 MINLEN:100  
done
```

```
#adapter sequence 'CTGTCTCTTATACACATCT' in novaseq6000.fa file.  
## Truseq3-PE.fa, NexteraPE-PE.fa, TruSeq2-PE.fa, TruSeq3-PE-2.fa TruSeq3-SE.fa fail
```

```
#!/bin/bash  
#sh TRIM.sh 1>TRIM.log 2>TRIM.err
```

```
TRIMMO=/home/lin/GS2018_SNPCalling/soft/Trimmomatic-0.39/trimmomatic-0.39.jar
```

```
DIR=/home/lin/2020AutophenotypingNGS  
FASTQ=${DIR}/JP-MUS-20200612-01D/Fastq  
ls ${FASTQ} |grep F0| sed -e "s/_\t/g" | cut -f2 | uniq -d > ${DIR}/script/sample5_F0.list
```

```
for i in `cat ${DIR}/script/sample5_F0.list`  
do  
mkdir ${DIR}/TRIM/${i}  
TRIM=${DIR}/TRIM/${i}
```

```

trim_galore \
--fastqc --gzip --length 100 -o ${TRIM} --paired --stringency 10 --cores 12 \
${FASTQ}/F0_${i}_R1.fastq.gz ${FASTQ}/F0_${i}_R2.fastq.gz

# hint for adapter sequence
# https://support.illumina.com/bulletins/2016/12/what-sequences-do-i-use-for-adapter-trimming.html
done

for i in `cat ${DIR}/script/sample1.list`
do
mkdir ${DIR}/TRIM/${i}
TRIM=${DIR}/TRIM/${i}

trim_galore \
--fastqc --gzip --length 100 -o ${TRIM} --paired --2colour 20 --stringency 10 --cores 12 \
-a CTGTCTCTTATACACATCT \
${FASTQ}/JST_GT_${i}_R1.fastq.gz ${FASTQ}/JST_GT_${i}_R2.fastq.gz

done

## 
for i in `cat ${DIR}/script/sample1.list`
do
mkdir ${DIR}/TRIM/${i}
TRIM=${DIR}/TRIM/${i}
java -jar ${TRIMMO} \
PE -threads 24 \
${FASTQ}/*${i}_R1.fastq.gz ${FASTQ}/*${i}_R2.fastq.gz \
${TRIM}/${i}"_paired_R1.fq.gz" ${TRIM}/${i}"_unpaired_R1.fq.gz" \
${TRIM}/${i}"_paired_R2.fq.gz" ${TRIM}/${i}"_unpaired_R2.fq.gz" \
ILLUMINACLIP:/home/lin/GS2018_SNPcalling/soft/Trimmomatic-0.39/adapters/novaseq6000-SE.fa:2:30:10
SLIDINGWINDOW:30:20 AVGQUAL:20 MINLEN:100
done

#adapter sequence 'CTGTCTCTTATACACATCT' in novaseq6000.fa file.
## Truseq3-PE.fa, NexteraPE-PE.fa, TruSeq2-PE.fa, TruSeq3-PE-2.fa TruSeq3-SE.fa fail

```

```

#!/bin/bash
#sh BWA.sh 1>BWA.log 2>BWA.err

DIR=/home/lin/2020AutophenotypingNGS
REF=/home/lin/GS2018_SNPcalling/reference/fr3.fa

#bwa index ${REF}
#samtools faidx ${REF}

mkdir ${DIR}/BWA/

for i in `cat ${DIR}/script/sample1234.list`
do
mkdir ${DIR}/BWA/${i}
BWA=${DIR}/BWA/${i}
TRIM=${DIR}/TRIM/${i}

bwa mem -t 32 -M ${REF} ${TRIM}"/JST_GT_"${i}"_R1_val_1.fq.gz"
${TRIM}"/JST_GT_"${i}"_R2_val_2.fq.gz" -R "@RG\tID:"${i}"\tSM:"${i}"\tPL:Illumina" | samtools view -b -F 2308 -q 10 -@ 32 -o ${BWA}/${i}.bam
samtools sort -n -@ 32 -o ${BWA}/${i}_namesort.bam ${BWA}/${i}.bam
samtools fixmate -@ 32 -m ${BWA}/${i}_namesort.bam ${BWA}/${i}_fixmate.bam
samtools sort -@ 32 -o ${BWA}/${i}_sorted.bam ${BWA}/${i}_fixmate.bam
samtools index ${BWA}/${i}_sorted.bam

rm ${BWA}/${i}.sam
rm ${BWA}/${i}.bam
rm ${BWA}/${i}_namesort.bam
rm ${BWA}/${i}_fixmate.bam

done

#!/bin/bash
#sh BWA.sh 1>BWA.log 2>BWA.err

```

DIR=/home/lin/2020AutophenotypingNGS

```
REF=/home/lin/GS2018_SNPcalling/reference/fr3.fa
```

```
#bwa index ${REF}
```

```
#samtools faidx ${REF}
```

```
mkdir ${DIR}/BWA/
```

```
for i in `cat ${DIR}/script/sample5_F0.list`
```

```
do
```

```
mkdir ${DIR}/BWA/${i}
```

```
BWA=${DIR}/BWA/${i}
```

```
TRIM=${DIR}/TRIM/${i}
```

```
bwa mem -t 32 -M ${REF} ${TRIM}"/F0_"${i}"_R1_val_1.fq.gz" ${TRIM}"/F0_"${i}"_R2_val_2.fq.gz" -R  
"@RG\tID:"${i}"\tSM:"${i}"\tPL:Illumina"\|samtools view -b -F 2308 -q 10 -@ 32 -o ${BWA}/${i}.bam"  
samtools sort -n -@ 32 -o ${BWA}/${i}_namesort.bam ${BWA}/${i}.bam  
samtools fixmate -@ 32 -m ${BWA}/${i}_namesort.bam ${BWA}/${i}_fixmate.bam  
samtools sort -@ 32 -o ${BWA}/${i}_sorted.bam ${BWA}/${i}_fixmate.bam  
samtools index ${BWA}/${i}_sorted.bam
```

```
rm ${BWA}/${i}.sam
```

```
rm ${BWA}/${i}.bam
```

```
rm ${BWA}/${i}_namesort.bam
```

```
rm ${BWA}/${i}_fixmate.bam
```

```
done
```

```
#!/bin/bash
```

```
#sh GATK_GRAS_parallel.sh 1>>GATK_GRAS_parallel.log 2>>GATK_GRAS_parallel.err
```

```
#PBS -N gatk
```

```
#PBS -l select=1:ncpus=8:mem=15gb,walltime=2:00:00
```

```
#PBS -j oe
```

```
echo "START -----"
```

```
#module add java/12.0.2
```

```

#module load parallel
#module load gatk

statsdir=/home/lin/2020AutophenotypingNGS
list_dir=/home/lin/2020AutophenotypingNGS/script/sample12345_F0.list
REF=/home/lin/GS2018_SNPcalling/reference/fr3.fa
gatk=/home/lin/gatk-4.1.6.0/gatk

mkdir ${statsdir}/VCF/
sudo chmod -R 777 ./VCF/
### Use gnu-parallel to use multiple cores
### within one script
cat ${list_dir} | parallel --verbose -j 12 "${gatk}" HaplotypeCaller --output-mode
EMIT_ALL_CONFIDENT_SITES -stand-call-conf 30 -R ${REF} -I ${statsdir}/BWA/{}_{}/{}_sorted.bam -O
${statsdir}/VCF/{}_pe_variants_gatk.g.vcf.gz -ERC GVCF"

echo "FINISH -----"

## Bash

## Call variants per-sample (GVCF mode)

#_____
```

DIR=/home/lin/2020AutophenotypingNGS

list_dir=/home/lin/2020AutophenotypingNGS/script/sample12345_F0.list

REF=/home/lin/GS2018_SNPcalling/reference/fr3.fa

gatk=/home/lin/gatk-4.1.6.0/gatk

```

#rm ${DIR}/BWA/*/*_pe_variants_gatk.g.*
#cp ${DIR}/BWA/*/*_pe_variants_gatk.g.* ${DIR}/VCF
##sudo chmod +x *
```

#_____

```

#rm ${DIR}/BWA/*/*_pe_variants_gatk.g.*
#cp ${DIR}/BWA/*/*_pe_variants_gatk.g.* ${DIR}/VCF
##sudo chmod +x *

echo "${gatk} GenomicsDBImport" > ${DIR}/script/GenomicsDBImport.sh
for i in `cat ${list_dir}`
do
echo "-V ${DIR}/VCF/${i}_pe_variants_gatk.g.vcf.gz" >> ${DIR}/script/GenomicsDBImport.sh
done
echo "-R ${REF}" >> ${DIR}/script/GenomicsDBImport.sh
echo "-O ${DIR}/VCF/cohort.g.vcf.gz" >> ${DIR}/script/GenomicsDBImport.sh
echo "--genomicsdb-workspace-path my_database"
echo                                         "--intervals
chr1,chr2,chr3,chr4,chr5,chr6,chr7,chr8,chr9,chr10,chr11,chr12,chr13,chr14,chr15,chr16,chr17,chr18,chr19,chr
20,chr21,chr22"

cat ${DIR}/script/GenomicsDBImport.sh | tr "\n" " " > ${DIR}/script/Spa_GenomicsDBImport.sh

```

#_____

#running

```
tabix -p vcf ${DIR}/VCF/cohort.g.vcf.gz
```

```

${gatk}      GenotypeGVCFs      -R      ${REF}      -V      ${DIR}/VCF/cohort.g.vcf.gz      -O
${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --tmp-dir=${DIR}/VCF/temp

```

```

#bgzip -d ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz
#grep "##contig" ${DIR}/VCF/Output_jonit_call_cohort.vcf | sed -e "s/= /g" -e "s/,/ /g" | awk '{print $3}' >
${DIR}/VCF/pos.list
#cat -n ${DIR}/VCF/pos.list | awk '{print $2"\t"$1}' > ${DIR}/VCF/Chom.map
#bgzip ${DIR}/VCF/Output_jonit_call_cohort.vcf
tabix -p vcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz
#vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --min-meanDP 6 --max-meanDP 500 --max-
missing 0.3 --remove-indels --recode --stdout > ${DIR}/VCF/Hetero_realigned_cov10_filtered.vcf

```

```

cat ${DIR}/VCF_filtering/filter_parameter1.txt | parallel --verbose "vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --chr chr1 --chr chr2 --chr chr3 --chr chr4 --chr chr5 --chr chr6 --chr chr7 --chr chr8 --chr chr9 --chr chr10 --chr chr11 --chr chr12 --chr chr13 --chr chr14 --chr chr15 --chr chr16 --chr chr17 --chr chr18 --chr chr19 --chr chr20 --chr chr21 --chr chr22 --remove-indels --min-alleles 2 --max-alleles 2 --minQ 10 --min-meanDP 6 --max-meanDP 500 --maf 0.01 --hwe 0.05 --max-missing {}"

## Line plot to visualize.

## --min-meanDP 6 to 10, and --max-missing 0-1
for i in $(seq 0 0.1 1)
do
echo $i >> ${DIR}/VCF_filtering/filter_parameter2.txt
done

#single
(cat ${DIR}/VCF_filtering/filter_parameter2.txt | parallel --verbose --joblog parallel.log "vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --chr chr1 --chr chr2 --chr chr3 --chr chr4 --chr chr5 --chr chr6 --chr chr7 --chr chr8 --chr chr9 --chr chr10 --chr chr11 --chr chr12 --chr chr13 --chr chr14 --chr chr15 --chr chr16 --chr chr17 --chr chr18 --chr chr19 --chr chr20 --chr chr21 --chr chr22 --remove-indels --min-alleles 2 --max-alleles 2 --minQ 10 --min-meanDP 5 --max-meanDP 500 --maf 0.01 --hwe 0.05 --max-missing {}" 2>&1) | tee ${DIR}/VCF_filtering/log5.txt
cat log6.txt |grep possible|sed s/"After filtering, kept"//g |sed s/"out of a possible 2196091 Sites"//g

## from plot, decide the fitering parameters and imputation?
## how many trainning individuals are nessisary?
#multiple

cd ${DIR}/VCF_filtering
for i in {6..15}
do
(cat ${DIR}/VCF_filtering/filter_parameter2.txt | parallel --verbose --joblog parallel.log "vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --chr chr1 --chr chr2 --chr chr3 --chr chr4 --chr chr5 --chr chr6 --chr chr7 --chr chr8 --chr chr9 --chr chr10 --chr chr11 --chr chr12 --chr chr13 --chr chr14 --chr chr15 --chr chr16 --chr chr17 --chr chr18 --chr chr19 --chr chr20 --chr chr21 --chr chr22 --remove-indels --min-alleles 2 --max-alleles 2 --minQ 10 --min-meanDP ${i} --max-meanDP 500 --minDP 10 --maf 0.01 --hwe 0.05 --max-missing {}" 2>&1) | tee ${DIR}/VCF_filtering/"log"${i}.txt"
done
parallel --verbose "cat log{}.txt |grep possible|sed s/'After filtering, kept'//g |sed s/'out of a possible 2324132

```

```

Sites'//g |sort > SNP_count_min-meanDP_{}.txt" ::: {6..15}
paste $(ls|grep SNP_count_min-meanDP_|sort -t " " -k4 -g) > merge.txt

## SNP filtering and matrix
task(){
mkdir ${DIR}/VCF_imputation/${i};

vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --chr chr1 --chr chr2 --chr chr3 --chr chr4 --chr
chr5 --chr chr6 --chr chr7 --chr chr8 --chr chr9 --chr chr10 --chr chr11 --chr chr12 --chr chr13 --chr chr14 --chr
chr15 --chr chr16 --chr chr17 --chr chr18 --chr chr19 --chr chr20 --chr chr21 --chr chr22 --remove-indels --min-
alleles 2 --max-alleles 2 --minQ 10 --min-meanDP 10 --max-meanDP 500 --minDP 10 --maf 0.01 --hwe 0.05 --
max-missing ${i} --recode --stdout > ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.vcf;
zgrep -v "##" ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.vcf | awk '{print $1"\t"$2}' >
${DIR}/VCF_imputation/${i}/position.list;
cat -n ${DIR}/VCF_imputation/${i}/position.list | awk '{print $2"\t"$1}' >
${DIR}/VCF_imputation/${i}/Chom.map;
bgzip ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.vcf;
tabix -p vcf ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.vcf.gz;
vcftools --gzvcf ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.vcf.gz --plink --chrom-map
${DIR}/VCF_imputation/${i}/Chom.map --out
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered;
plink1 --ped ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.ped --map
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.map --recodeA --out
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered --noweb;
echo "empty" > ${DIR}/VCF_imputation/${i}/IDlist;
zgrep "#CHROM" ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.vcf.gz | cut -f10- | sed -e
"s/\t/\n/g" >> ${DIR}/VCF_imputation/${i}/IDlist;
sed -e "s/empty//g" ${DIR}/VCF_imputation/${i}/IDlist > ${DIR}/VCF_imputation/${i}/IDs.txt
cat ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- >
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered2.raw;
paste -d " " ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered2.raw >
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered3.raw;
sed -n "1p" ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- | sed -e "s/\n/g"
> ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.pos;
less ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.pos|sed -e "s/_/\t/g"|sed -e "s/:/\t/g"|sed -e
"s/-/\t/g"|sed "s/"chr"/\t/g" | awk '{print $1, $2}'>${DIR}/VCF_imputation/${i}/lin.map
}

# multiple #for i in $(seq 0 0.1 1);do task "$i" &done

```

```

#single
i = 0.8
task "$i"

## Imputation
#for i in $(seq 0 0.1 0.9)
#do
java -jar ~/GS2018_SNPcalling/VCF_imputation/20200114_linkimpute/LinkImpute.jar
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.raw
${DIR}/VCF_imputation/${i}/Imputed.raw;
cat ${DIR}/VCF_imputation/${i}/Imputed.raw | cut -d " " -f7- >
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered2.raw;
paste -d" " ${DIR}/VCF_imputation/${i}/IDs.txt
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered2.raw >
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered3.raw;
sed -n "1p" ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- | sed -e "s/\n/g"
> ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.pos;
less ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.pos|sed -e "s/_t/g"|sed -e "s:/t/g"|sed -e
"s:-/t/g"|sed "s://"chr"//g"| awk '{print $1, $2}'>${DIR}/VCF_imputation/${i}/lin.map
#done

## missing rate
i=0.8
cat ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- >
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered2_m.raw;
paste -d" " ${DIR}/VCF_imputation/${i}/IDs.txt
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered2_m.raw >
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered3_m.raw;
##

#for i in $(seq 0 0.1 1)
#do

less ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.pos|sed -e "s/_t/g"|sed -e "s:/t/g"|sed -e
"s:-/t/g"|sed "s://"chr"//g"| awk '{print $1, $2}'>${DIR}/VCF_imputation/${i}/lin.map
#done

```

```
###
```

```
vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --chr chr1 --chr chr2 --chr chr3 --chr chr4 --chr  
chr5 --chr chr6 --chr chr7 --chr chr8 --chr chr9 --chr chr10 --chr chr11 --chr chr12 --chr chr13 --chr chr14 --chr  
chr15 --chr chr16 --chr chr17 --chr chr18 --chr chr19 --chr chr20 --chr chr21 --chr chr22 --remove-indels --min-  
alleles 2 --max-alleles 2 --minQ 10 --min-meanDP 10 --max-meanDP 500 --maf 0.01 --hwe 0.05 --max-missing  
1 --recode --stdout > ${DIR}/VCF_imputation/1/Hetero_realigned_cov10_filtered.vcf

zgrep -v "##" ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.vcf | awk '{print $1"\t"$2}' >  
${DIR}/VCF_imputation/${i}/position.list
cat -n ${DIR}/VCF_imputation/${i}/position.list | awk '{print $2"\t"$1}' >  
${DIR}/VCF_imputation/${i}/Chom.map
bgzip ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.vcf
tabix -p vcf ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.vcf.gz
vcftools --gzvcf ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.vcf.gz --plink --chrom-map  
${DIR}/VCF_imputation/${i}/Chom.map --out ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered
plink1 --ped ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.ped --map  
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.map --recodeA --out  
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered --noweb
echo "empty" > ${DIR}/VCF_imputation/${i}/IDlist
zgrep "#CHROM" ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.vcf.gz | cut -f10- | sed -e  
"s/\t/n/g" >> ${DIR}/VCF_imputation/${i}/IDlist
sed -e "s/empty//g" ${DIR}/VCF_imputation/${i}/IDlist > ${DIR}/VCF_imputation/${i}/IDs.txt
cat ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- >  
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered2.raw
paste -d " " ${DIR}/VCF_imputation/${i}/IDs.txt  
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered2.raw >  
${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered3.raw
sed -n "1p" ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- | sed -e "s/\n/g"  
> ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.pos
less ${DIR}/VCF_imputation/${i}/Hetero_realigned_cov10_filtered.pos|sed -e "s/_/\t/g"|sed -e "s/:/\t/g"|sed -e  
"s/-/\t/g"|awk '{print $2,$3+$5}'>${DIR}/VCF_imputation/${i}/lin.map
```

```
##SNP fitlering 20211026. maxmissing = 0.7
```

```

DIR=/home/lin/2020AutophenotypingNGS
# Post_VCF_2wSNPs211026
mkdir ${DIR}/Post_VCF_2wSNPs211026
filter=${DIR}/Post_VCF_2wSNPs211026

task(){
mkdir ${filter}/${i};
vcftools --gzvcf ${DIR}/VCF/Output_jonit_call_cohort.vcf.gz --chr chr1 --chr chr2 --chr chr3 --chr chr4 --chr
chr5 --chr chr6 --chr chr7 --chr chr8 --chr chr9 --chr chr10 --chr chr11 --chr chr12 --chr chr13 --chr chr14 --chr
chr15 --chr chr16 --chr chr17 --chr chr18 --chr chr19 --chr chr20 --chr chr21 --chr chr22 --remove-indels --min-
alleles 2 --max-alleles 2 --minQ 20 --min-meanDP 20 --minDP 10 --max-meanDP 300 --maf 0.01 --hwe 0.05 --
max-missing ${i} --recode --stdout > ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf;
zgrep -v "##" ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf | awk '{print $1"\t"$2}' >
${filter}/${i}/position.list;
cat -n ${filter}/${i}/position.list | awk '{print $2"\t"$1}' > ${filter}/${i}/Chom.map;
bgzip ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf;
tabix -p vcf ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf.gz;
vcftools --gzvcf ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf.gz --plink --chrom-map
${filter}/${i}/Chom.map --out ${filter}/${i}/Hetero_realigned_cov10_filtered;
plink1 --ped ${filter}/${i}/Hetero_realigned_cov10_filtered.ped --map
${filter}/${i}/Hetero_realigned_cov10_filtered.map --recodeA --out
${filter}/${i}/Hetero_realigned_cov10_filtered --noweb;
echo "empty" > ${filter}/${i}/IDlist;
zgrep "#CHROM" ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf.gz | cut -f10- | sed -e "s/\t\n/g" >>
${filter}/${i}/IDlist;
sed -e "s/empty//g" ${filter}/${i}/IDlist > ${filter}/${i}/IDs.txt
cat ${filter}/${i}/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- >
${filter}/${i}/Hetero_realigned_cov10_filtered2.raw;
paste -d " " ${filter}/${i}/IDs.txt ${filter}/${i}/Hetero_realigned_cov10_filtered2.raw >
${filter}/${i}/Hetero_realigned_cov10_filtered3.raw;
sed -n "1p" ${filter}/${i}/Hetero_realigned_cov10_filtered.raw | cut -d " " -f7- | sed -e "s/ \n/g" >
${filter}/${i}/Hetero_realigned_cov10_filtered.pos;
less ${filter}/${i}/Hetero_realigned_cov10_filtered.pos|sed -e "s/_\t/g"|sed -e "s/:_\t/g"|sed -e "s/-_\t/g"|sed
"s/_chr//g"|awk '{print $1, $2}'>${filter}/${i}/lin.map
}

# for i in $(seq 0 0.1 1);do task "$i" &done
i=0.7
task "$i"

```

```
kept 16471 out of a possible 2466708 Sites
```

```
# imputaion
Im_soft=/home/lin/GS2018_SNPcalling/VCF_imputation/20200114_linkimpute
mkdir ${DIR}/VCF_imputation_211026
imputation=${DIR}/VCF_imputation_211026
taskIM(){
mkdir ${imputation}/${i}
java -jar ${Im_soft}/LinkImpute.jar ${filter}/${i}/Hetero_realigned_cov10_filtered.raw
${imputation}/${i}/Imputed.raw;
cat ${imputation}/${i}/Imputed.raw | cut -d " " -f7- >
${imputation}/${i}/Hetero_realigned_cov10_filtered2.raw;
cp ${filter}/${i}/IDs.txt ${imputation}/${i}/IDs.txt
cp ${filter}/${i}/lin.map ${imputation}/${i}/lin.map
paste -d " " ${imputation}/${i}/IDs.txt ${imputation}/${i}/Hetero_realigned_cov10_filtered2.raw >
${imputation}/${i}/Hetero_realigned_cov10_filtered3.raw;
}

taskIM "$i"

####
# Ne = 27.6
# vcf2genepop https://github.com/z0on/2bRAD_denovo/blob/master/vcf2genepop.pl

gunzip ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf.gz
/home/lin/GS2018_SNPcalling/Haplo_20210102/vcf2genepop.pl
vcf=${filter}/${i}/Hetero_realigned_cov10_filtered.vcf > ${filter}/filtered_Ne.gen
bgzip ${filter}/${i}/Hetero_realigned_cov10_filtered.vcf

awk '/chr/ { print; print "pop"; next }1' ${filter}/filtered_Ne.gen >> ${filter}/filtered_Ne_pop.gen
#!/bin/bash

#module add java/12.0.2
#module load parallel
#module load gatk
```

```

statsdir=/home/lin/2020AutophenotypingNGS/script
list_dir=/home/lin/2020AutophenotypingNGS/script/sample1234.list
TRIM_dir=/home/lin/2020AutophenotypingNGS/TRIM
list_dir_f0=/home/lin/2020AutophenotypingNGS/script/sample5_F0.list

```

```
### Use gnu-parallel to use multiple cores
```

```
### within one script
```

```

touch ${statsdir}/trined_paired_num_read_1.txt
touch ${statsdir}/trined_paired_num_read_2.txt
touch ${statsdir}/trined_paired_bp_1.txt
touch ${statsdir}/trined_paired_bp_2.txt
sudo chmod 777 ${statsdir}/trined_paired_num_read_1.txt
sudo chmod 777 ${statsdir}/trined_paired_num_read_2.txt
sudo chmod 777 ${statsdir}/trined_paired_bp_1.txt
sudo chmod 777 ${statsdir}/trined_paired_bp_2.txt

```

```

cat $list_dir | parallel --verbose -j 12 "sudo chmod 777 ${TRIM_dir}/{}/JST_GT_{ }_R1_val_1.fq.gz| sudo chmod 777 ${TRIM_dir}/{}/JST_GT_{ }_R2_val_2.fq.gz|sudo seqstats ${TRIM_dir}/{}/JST_GT_{ }_R1_val_1.fq.gz|sed -n 1p|cut -f 2 >>${statsdir}/trined_paired_num_read_1.txt|sudo seqstats ${TRIM_dir}/{}/JST_GT_{ }_R2_val_2.fq.gz|sed -n 1p|cut -f 2 >>${statsdir}/trined_paired_num_read_2.txt|sudo seqstats ${TRIM_dir}/{}/JST_GT_{ }_R1_val_1.fq.gz|sed -n 2p|cut -f 2|cut -d' '| -f1 >>${statsdir}/trined_paired_bp_1.txt|sudo seqstats ${TRIM_dir}/{}/JST_GT_{ }_R2_val_2.fq.gz|sed -n 2p|cut -f 2|cut -d' '| -f1 >>${statsdir}/trined_paired_bp_2.txt"

```

```

touch ${statsdir}/f0_trined_paired_num_read_1.txt
touch ${statsdir}/f0_trined_paired_num_read_2.txt
touch ${statsdir}/f0_trined_paired_bp_1.txt
touch ${statsdir}/f0_trined_paired_bp_2.txt
sudo chmod 777 ${statsdir}/f0_trined_paired_num_read_1.txt
sudo chmod 777 ${statsdir}/f0_trined_paired_num_read_2.txt
sudo chmod 777 ${statsdir}/f0_trined_paired_bp_1.txt
sudo chmod 777 ${statsdir}/f0_trined_paired_bp_2.txt

```

```

cat $list_dir_f0 | parallel --verbose -j 12 "sudo chmod 777 ${TRIM_dir}/{}/F0_{ }_R1_val_1.fq.gz| sudo chmod 777 ${TRIM_dir}/{}/F0_{ }_R2_val_2.fq.gz|sudo seqstats ${TRIM_dir}/{}/F0_{ }_R1_val_1.fq.gz|sed -n 1p|cut -f 2 >>${statsdir}/f0_trined_paired_num_read_1.txt|sudo seqstats ${TRIM_dir}/{}/F0_{ }_R2_val_2.fq.gz|sed -n 1p|cut -f 2 >>${statsdir}/f0_trined_paired_num_read_2.txt|sudo seqstats ${TRIM_dir}/{}/F0_{ }_R1_val_1.fq.gz

```

```
|sed -n 2p|cut -f 2|cut -d' ' -f1 >>${statsdir}/f0_trined_paired_bp_1.txt|sudo seqstats  
$TRIM_dir/{}|F0_{}|R2_val_2.fq.gz|sed -n 2p|cut -f 2|cut -d' ' -f1 >>${statsdir}/f0_trined_paired_bp_2.txt"
```

5.1.3 R script for genotyping

```
# raw
rtb1<-read.table("raw_paired_bp_1.txt")
rtb2<-read.table("raw_paired_bp_2.txt")
rtn1<-read.table("raw_paired_num_read_1.txt")
rtn2<-read.table("raw_paired_num_read_2.txt")
rftb1<-read.table("f0_raw_paired_bp_1.txt")
rftb2<-read.table("f0_raw_paired_bp_2.txt")
rftn1<-read.table("f0_raw_paired_num_read_1.txt")
rftn2<-read.table("f0_raw_paired_num_read_2.txt")
# trim
tb1<-read.table("trined_paired_bp_1.txt")
tb2<-read.table("trined_paired_bp_2.txt")
tn1<-read.table("trined_paired_num_read_1.txt")
tn2<-read.table("trined_paired_num_read_2.txt")
ftb1<-read.table("f0_trined_paired_bp_1.txt")
ftb2<-read.table("f0_trined_paired_bp_2.txt")
ftn1<-read.table("f0_trined_paired_num_read_1.txt")
ftn2<-read.table("f0_trined_paired_num_read_2.txt")

allraw = rbind((rftn1+rftn2),(rtn1+rtn2))$V1
mean(allraw);sd(allraw)

alltrim = rbind((ftn1+ftn2),(tn1+tn2))$V1
mean(alltrim);sd(alltrim)

allraw_b = rbind((rftb1+rftb2),(rtb1+rtb2))$V1
mean(allraw_b);sd(allraw_b)
mean(allraw_b / allraw)

alltrim_b = rbind((ftb1+ftb2),(tb1+tb2))$V1
mean(alltrim_b);sd(alltrim_b)
mean(alltrim_b / alltrim)

(sum(tb1)+sum(tb2))
(sum(ftb1)+sum(ftb2))
(sum(tb1)+sum(tb2)+sum(ftb1)+sum(ftb2))
```

```
mean(unlist(tn1+tn2))
sd(unlist(tn1+tn2))
mean(unlist(ftn1+ftn2))
sd(unlist(ftn1+ftn2))
mean(c(unlist(tn1+tn2),unlist(ftn1+ftn2)))
sd(c(unlist(tn1+tn2),unlist(ftn1+ftn2)))
```

5.1.4 Python script for population structure

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm
import seaborn as sns; sns.set()
import matplotlib as mpl
mpl.__version__
# Loading data
pheno = pd.read_csv("./pheno_cleaned_ngs.csv")
#I = pd.read_table("./Hetero_realigned_cov10_filtered3.raw",sep="\s+")
I = pd.read_table("./reorderG.txt",sep="\s+")-1
# bcw=pheno["BCW"]
from sklearn.manifold import TSNE
model = TSNE(random_state=0,perplexity=20)
tsne5 = model.fit_transform(I)
lin_wide=1.28
font_size=12
%matplotlib inline
%config InlineBackend.figure_format = 'svg'
plt.style.use('seaborn-white')
plt.style.use('seaborn-paper')
plt.rc('font',size=font_size)
plt.rc('lines',lw=lin_wide)
plt.rcParams['font.family'] = 'serif'
plt.rcParams['font.sans-serif'] = 'Times New Roman'
#plt.rcParams['axes.formatter.use_mathtext'] = True
#plt.rcParams['mathtext.bf'] = 'serif:normal'
#plt.rcParams['text.hinting'] = 'none'
#plt.rcParams['text.usetex'] = True
#plt.rcParams['axes.labelweight'] = 'normal'

plt.rcParams['axes.linewidth'] = lin_wide
plt.rcParams['xtick.major.width'] = lin_wide
plt.rcParams['xtick.minor.width'] = lin_wide
plt.rcParams['xtick.labelsize'] = font_size
plt.rcParams['ytick.major.width'] = lin_wide
plt.rcParams['ytick.minor.width'] = lin_wide
```

```
plt.rcParams['ytick.labelsize'] = font_size
plt.rcParams['axes.spines.bottom'] = True
# plt.style.use('bmh')
plt.scatter(tsne5[:, 0], tsne5[:, 1], edgecolor='k', linewidth=lin_wide, s=s_value)
plt.xlabel("t-SNE coordinate 1", size=font_size)
plt.ylabel("t-SNE coordinate 2", size=font_size)

plt.gcf().set_size_inches(6, 6)
plt.savefig("C5tSNE_HC.pdf", format="pdf")
plt.show()
```

5.1.5 R script for heritability and genetic correlation

```
reorderG<-read.table("GS2019_long/reorderG.txt")
x<-as.matrix(reorderG)-1
pheno<-read.csv("GS2019_long/pheno_cleaned_ngs.csv")
n<-dim(x)[1]

library("sommer")
packageVersion("sommer")
A <- A.mat(x,n.core=8)
row.names(A)=1:n;colnames(A)=1:n
data <- data.frame(SL1=SL1,SL2=SL2,SL3=SL3,SL4=SL4,id=factor(1:n))
attach(data)
ans.m <- mmer(cbind(SL1,SL2,SL3,SL4)~1,random=~ vs(id, Gu=A, Gtc=unsm(4)),
               rcov=~ vs(units, Gtc=unsm(4)),
               data=data)

(sp_cor<-cov2cor(ans.m$sigma$`u:id`))

pin(ans.m, h2~V1/(V1+V11))
pin(ans.m, h2~V5/(V5+V15))
pin(ans.m, h2~V8/(V8+V18))
pin(ans.m, h2~V10/(V10+V20))
```

5.1.6 R script for GWAS

```
R.Version()$version.string  
library("ggplot2")  
library("ggpubr")  
library("extrafont")  
print(paste("ggplot2 version",packageVersion("ggplot2")))  
print(paste("ggpubr version",packageVersion("ggpubr")))  
print(paste("extrafont version",packageVersion("extrafont")))  
# remotes::install_version("Rttf2pt1", version = "1.3.8")  
# ttf_import()  
# fonts()  
# loadfonts(device = "pdf")  
  
phenoRaw <- read.csv("./200818JST210113.csv")  
# attach(pheno)  
str(phenoRaw)  
  
# Filtration  
pheno_cleaned<-phenoRaw[-740,]  
str(pheno_cleaned)  
  
pheno<-data.frame(  
  SL1 = pheno_cleaned$X200107_SL,  
  SL2 = pheno_cleaned$X2005SL,  
  SL3 = pheno_cleaned$X200818_SL,  
  SL4 = pheno_cleaned$X210113_SL,  
  gender = as.character(pheno_cleaned$X210113_sex),  
  original_tank = as.character(pheno_cleaned$Original_tank),  
  feeding_tank = as.character(pheno_cleaned$Feeding_tank)  
)  
n<-length(pheno$SL1)  
attach(pheno)  
str(pheno)  
  
# Loading raw G matrix  
geno<-read.table("./Imputation/Hetero_realigned_cov10_filtered3.raw", header=T, row.names=1)  
dim(geno)
```

```

g_order<-c()
for (i in as.character(phenoRaw$X200415_GID)){
  g_order<-c(g_order,match(i,rownames(geno)))
  #print(i)
}
g_order_na.omit<-na.omit(g_order)
reorderG<-geno[g_order_na.omit,]

(rm<-which(as.character(phenoRaw$X200415_GID) %in% rownames(geno)==FALSE))
as.character(phenoRaw$X200415_GID)[rm]
#G_id = 106 was not genotyped

row.names(reorderG)=1:dim(reorderG)[1]
x <- as.matrix(reorderG)-1
str(x)
write.table(reorderG,"reorderG.txt")
write.csv(pheno,"pheno_cleaned_ngs.csv")

# GBLUP
library(rrBLUP)
print(paste("rrBLUP version",packageVersion("rrBLUP")))

# genomic relationship matrix
A <- A.mat(x)
row.names(A)=1:n;colnames(A)=1:n

lin_map<-read.table("./Imputation/lin.map")
g <- data.frame(rownames(lin_map),lin_map$V1, lin_map$V2, t(x))
rownames(g)<-1:nrow(g)
colnames(g)<-c("marker", "chrom", "pos", rownames(x))
# Bonferroni-corrected significance threshold
(thred <- round(-log10(0.05/nrow(g)),3))

# SL1
lengthframe <-data.frame(1:n, SL1)
colnames(lengthframe)<-c("gid", "length")
# Performing GWAS

```

```

GWAS_length<-GWAS(lengthframe, g,K=A, n.PC = 10)
# SNP markers with lowest P-values
GWAS_length$chrom[head(order(GWAS_length$length,decreasing = TRUE),3)]
# Corresponding position (base pair)
GWAS_length$pos[head(order(GWAS_length$length,decreasing = TRUE),3)]]

# top100
top100<-cbind(rep("GS2019_long",100),GWAS_length$chrom[head(order(GWAS_length$length,decreasing = TRUE),100)],
                GWAS_length$pos[head(order(GWAS_length$length,decreasing = TRUE),100)])
colname<-c("Exp","chr","pos")
write.csv(top100,"GS2019long_GAWS_SL1_100.csv")

Chrom<-GWAS_length$chrom
GWAS_length$chrom<-factor(Chrom,levels=c(as.character(1:22)))
options(repr.plot.width=8, repr.plot.height=4)
line_th_unit<-0.6
my_y_title <- expression(paste("-", "log"[10], "(", italic("p"), ") for SL1"))
p1<-ggplot(GWAS_length, aes(x=1:dim(x)[2], y=length, color=chrom)) +
  ggpubr::theme_pubr(base_family="Times New Roman",base_size = 12,border = TRUE) +
  geom_point() +
  scale_color_manual(values = rep(c("black", "skyblue"), 24 )) +
  labs(x="Physical order of SNPs", y=my_y_title) +
  theme(legend.position="none",
        axis.text.y=element_text(size=12),
        axis.text.x=element_text(size=12),
        text=element_text(size=12,
                          family="Times New Roman")) +
  geom_hline(yintercept=thred, linetype="dashed", color = "red") +
  geom_text(aes(0,thred,label = thred, vjust = 1.2),color = "black",family="Times New Roman")
p1$theme$line$size<-line_th_unit
p1$theme$line$size<-line_th_unit
p1$theme$axis.ticks$size<-line_th_unit
p1$theme$axis.line$size<-line_th_unit
p1$theme$panel.grid$colour<-"black"
p1$theme$axis.ticks$colour<-"black"

# SL2

```

```

lengthframe <- data.frame(1:n, SL2)
colnames(lengthframe) <- c("gid", "length")
# Performing GWAS
GWAS_length <- GWAS(lengthframe, g,K=A, n.PC = 10)
# SNP markers with lowest P-values
GWAS_length$chrom[head(order(GWAS_length$length,decreasing = TRUE),3)]
# Corresponding position (base pair)
GWAS_length$pos[head(order(GWAS_length$length,decreasing = TRUE),3)]

# top100
top100<-cbind(rep("GS2019_long",100),GWAS_length$chrom[head(order(GWAS_length$length,decreasing = TRUE),100)],
                GWAS_length$pos[head(order(GWAS_length$length,decreasing = TRUE),100)])
colname<-c("Exp","chr","pos")
write.csv(top100,"GS2019long_GAWS_SL2_100.csv")

Chrom<-GWAS_length$chrom
GWAS_length$chrom<-factor(Chrom,levels=c(as.character(1:22)))
options(repr.plot.width=8, repr.plot.height=4)
my_y_title <- expression(paste("-", "log"[10], "(", italic("p"), ") for SL2"))
p2<-ggplot(GWAS_length, aes(x=1:dim(x)[2], y=length, color=chrom)) +
  ggpibr::theme_pubr(base_family="Times New Roman",base_size = 12,border = TRUE) +
  geom_point() +
  scale_color_manual(values = rep(c("black", "skyblue"), 24 )) +
  labs(x="Physical order of SNPs", y=my_y_title) +
  theme(legend.position="none",
        axis.text.y=element_text(size=12),
        axis.text.x=element_text(size=12),
        text=element_text(size=12,
                          family="Times New Roman")) +
  geom_hline(yintercept=thred, linetype="dashed", color = "red") +
  geom_text(aes(0,thred,label = thred, vjust = 1.2),color = "black",family="Times New Roman")
p2$theme$line$size<-line_th_unit
p2$theme$line$size<-line_th_unit
p2$theme$axis.ticks$size<-line_th_unit
p2$theme$axis.line$size<-line_th_unit
p2$theme$panel.grid$colour<-"black"
p2$theme$axis.ticks$colour<-"black"

```

```

# SL3
lengthframe <-data.frame(1:n, SL3)
colnames(lengthframe) <-c("gid", "length")
# Performing GWAS
GWAS_length <-GWAS(lengthframe, g,K=A, n.PC = 10)
# SNP markers with lowest P-values
GWAS_length$chrom[head(order(GWAS_length$length,decreasing = TRUE),3)]
# Corresponding position (base pair)
GWAS_length$pos[head(order(GWAS_length$length,decreasing = TRUE),3)]

# top100
top100<-cbind(rep("GS2019_long",100),GWAS_length$chrom[head(order(GWAS_length$length,decreasing = TRUE),100)],
                GWAS_length$pos[head(order(GWAS_length$length,decreasing = TRUE),100)])
colname<-c("Exp","chr","pos")
write.csv(top100,"GS2019long_GAWS_SL3_100.csv")

Chrom<-GWAS_length$chrom
GWAS_length$chrom<-factor(Chrom,levels=c(as.character(1:22)))
options(repr.plot.width=8, repr.plot.height=4)
my_y_title <- expression(paste("-", "log"[10], "(", italic("p"), ") for SL3"))
p3<-ggplot(GWAS_length, aes(x=1:dim(x)[2], y=length, color=chrom)) +
  ggpubr::theme_pubr(base_family="Times New Roman",base_size = 12,border = TRUE) +
  geom_point() +
  scale_color_manual(values = rep(c("black", "skyblue"), 24 )) +
  labs(x="Physical order of SNPs", y=my_y_title) +
  theme(legend.position="none",
        axis.text.y=element_text(size=12),
        axis.text.x=element_text(size=12),
        text=element_text(size=12,
                          family="Times New Roman")) +
  geom_hline(yintercept=thred, linetype="dashed", color = "red") +
  geom_text(aes(0,thred,label = thred, vjust = 1.2),color = "black",family="Times New Roman")
p3$theme$line$size<-line_th_unit
p3$theme$line$size<-line_th_unit
p3$theme$axis.ticks$size<-line_th_unit
p3$theme$axis.line$size<-line_th_unit

```

```

p3$theme$panel.grid$colour<-"black"
p3$theme$axis.ticks$colour<-"black"

# SL4
lengthframe <-data.frame(1:n, SL4)
colnames(lengthframe) <-c("gid", "length")
# Performing GWAS
GWAS_length <-GWAS(lengthframe, g,K=A, n.PC = 10)
# SNP markers with lowest P-values
GWAS_length$chrom[head(order(GWAS_length$length,decreasing = TRUE),3)]
# Corresponding position (base pair)
GWAS_length$pos[head(order(GWAS_length$length,decreasing = TRUE),3)]

# top100
top100<-cbind(rep("GS2019_long",100),GWAS_length$chrom[head(order(GWAS_length$length,decreasing = TRUE),100)],
                GWAS_length$pos[head(order(GWAS_length$length,decreasing = TRUE),100)])
colname<-c("Exp","chr","pos")
write.csv(top100,"GS2019long_GAWS_SL4_100.csv")

Chrom<-GWAS_length$chrom
GWAS_length$chrom<-factor(Chrom,levels=c(as.character(1:22)))
options(repr.plot.width=8, repr.plot.height=4)
my_y_title <- expression(paste("-", "log"[10], "(", italic("p"), ") for SL4"))
p4<-ggplot(GWAS_length, aes(x=1:dim(x)[2], y=length, color=chrom)) +
  ggpibr::theme_pubr(base_family="Times New Roman",base_size = 12,border = TRUE) +
  geom_point() +
  scale_color_manual(values = rep(c("black", "skyblue"), 24 )) +
  labs(x="Physical order of SNPs", y=my_y_title) +
  theme(legend.position="none",
        axis.text.y=element_text(size=12),
        axis.text.x=element_text(size=12),
        text=element_text(size=12,
                          family="Times New Roman")) +
  geom_hline(yintercept=thred, linetype="dashed", color = "red") +
  geom_text(aes(0,thred,label = thred, vjust = 1.2),color = "black",family="Times New Roman")
p4$theme$line$size<-line_th_unit
p4$theme$line$size<-line_th_unit

```

```
p4$theme$axis.ticks$size<-line_th_unit  
p4$theme$axis.line$size<-line_th_unit  
p4$theme$panel.grid$colour<-"black"  
p4$theme$axis.ticks$colour<-"black"  
  
options(repr.plot.width=6, repr.plot.height=12)  
figure <- ggpubr::ggarrange(p1, p2,p3,p4,  
  labels = c("a", "b","c","d"),  
  ncol = 2,  
  nrow = 2,  
  font.label = list(size = 14,  
    color = "black",  
    face = "plain",  
    family = "Times New Roman"))  
figure  
  
ggsave("Fig.pdf",device="pdf",units="in",width =6, height = 12 )
```

Section 5.2

5.2.1 R script for predictive abilities of genomic prediction

```
# The same codes were used for loading genotypes, phenotypes and genomic relationship matrix in Section 5.1
library(doParallel)
library(foreach)
cl<-makeCluster(16)

# Parameters for cross validation
repeats <- 10
n.fold <- 5
n.sample <- length(pheno$SL1)

registerDoParallel(cl)
system.time({
  GBLUP<-foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {
      bcw_test <- data
      bcw_test$SL1[id == i] <- NA
      res <- kin.blup(bcw_test, K=A, geno="gid", pheno="SL1")
      cor(data$SL1[id==i],res$pred[id==i])
    }
  }
})
stopImplicitCluster()

registerDoParallel(cl)
system.time({
  GBLUP2<-foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {
      bcw_test <- data
      bcw_test$SL2[id == i] <- NA
      res <- kin.blup(bcw_test, K=A, geno="gid", pheno="SL2")
      cor(data$SL2[id==i],res$pred[id==i])
    }
  }
})
```

```

        }
    })
stopImplicitCluster()

registerDoParallel(cl)
system.time({
  GBLUP3<-foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {
      bcw_test <- data
      bcw_test$SL3[id == i] <- NA
      res <- kin.blup(bcw_test, K=A, geno="gid", pheno="SL3")
      cor(data$SL3[id==i],res$pred[id==i])
    }
  }
})
stopImplicitCluster()

registerDoParallel(cl)
system.time({
  GBLUP4<-foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {
      bcw_test <- data
      bcw_test$SL4[id == i] <- NA
      res <- kin.blup(bcw_test, K=A, geno="gid", pheno="SL4")
      cor(data$SL4[id==i],res$pred[id==i])
    }
  }
})
stopImplicitCluster()

library("BGLR")
packageVersion("BGLR")

registerDoParallel(cl)

```

```

system.time({
BC1 <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$SL1[id == i] <- NA
    fmBC=BGLR(y=bew_test$SL1,ETA=list(list(X=x,model='BayesC')),
      nIter=10000,burnIn=2000,verbose = FALSE)
    cor(data$SL1[id == i],fmBC$yHat[id == i])
  }
}
))
stopImplicitCluster()

registerDoParallel(cl)
system.time({
BC2 <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$SL2[id == i] <- NA
    fmBC=BGLR(y=bew_test$SL2,ETA=list(list(X=x,model='BayesC')),
      nIter=10000,burnIn=2000,verbose = FALSE)
    cor(data$SL2[id == i],fmBC$yHat[id == i])
  }
}
))
stopImplicitCluster()

registerDoParallel(cl)
system.time({
BC3 <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data

```

```

bcw_test$SL3[id == i] <- NA
fmBC=BGLR(y=bcw_test$SL3,ETA=list(list(X=x,model='BayesC')),
            nIter=10000,burnIn=2000,verbose = FALSE)
cor(data$SL3[id == i],fmBC$yHat[id == i])
}
}
})
stopImplicitCluster()

registerDoParallel(cl)
system.time({
BC4 <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$SL4[id == i] <- NA
    fmBC=BGLR(y=bcw_test$SL4,ETA=list(list(X=x,model='BayesC')),
                nIter=10000,burnIn=2000,verbose = FALSE)
    cor(data$SL4[id == i],fmBC$yHat[id == i])
  }
}
})
stopImplicitCluster()

# Reproducing kernel
p<-ncol(x)
D<-(as.matrix(dist(x,method='euclidean'))^2)/p
h<-0.5;K<-exp(-h*D)
ETA<-list(list(K=K,model='RKHS'))

registerDoParallel(cl)
system.time({
RKHS1 <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data

```

```

bcw_test$SL1[id == i] <- NA
fmRKHS=BGLR(y=bcw_test$SL1,ETA=ETA,nIter=10000,burnIn=2000,verbose = FALSE)
cor(data$SL1[id == i],fmRKHS$yHat[id == i])
}
}
})
stopImplicitCluster()

registerDoParallel(cl)
system.time({
RKHS2 <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$SL2[id == i] <- NA
    fmRKHS=BGLR(y=bcw_test$SL2,ETA=ETA,nIter=10000,burnIn=2000,verbose = FALSE)
    cor(data$SL2[id == i],fmRKHS$yHat[id == i])
  }
}
})
stopImplicitCluster()

registerDoParallel(cl)
system.time({
RKHS3 <- foreach(j=1:repeats,.combine = "rbind") %do% {
  set.seed(100+3*j+1)
  id <- sample(1:n.sample %% n.fold) + 1
  foreach(i=1:n.fold,.packages="BGLR") %dopar% {
    bcw_test <- data
    bcw_test$SL3[id == i] <- NA
    fmRKHS=BGLR(y=bcw_test$SL3,ETA=ETA,nIter=10000,burnIn=2000,verbose = FALSE)
    cor(data$SL3[id == i],fmRKHS$yHat[id == i])
  }
}
})
stopImplicitCluster()

```

```

registerDoParallel(cl)
system.time({
  RKHS4 <- foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="BGLR") %dopar% {
      bcw_test <- data
      bcw_test$SL4[id == i] <- NA
      fmRKHS=BGLR(y=bcw_test$SL4,ETA=ETA,nIter=10000,burnIn=2000,verbose = FALSE)
      cor(data$SL4[id == i],fmRKHS$yHat[id == i])
    }
  }
})
stopImplicitCluster()

# data
Acc<-data.frame(unlist(GLBLUP),unlist(BC1),unlist(RKHS1),
                 unlist(GLBLUP2),unlist(BC2),unlist(RKHS2),
                 unlist(GLBLUP3),unlist(BC3),unlist(RKHS3),
                 unlist(GLBLUP4),unlist(BC4),unlist(RKHS4))

colnames(Acc)<-c("GLBLUP_SL1","BayesC_SL1","RKHS_SL1",
                 "GLBLUP_SL2","BayesC_SL2","RKHS_SL2",
                 "GLBLUP_SL3","BayesC_SL3","RKHS_SL3",
                 "GLBLUP_SL4","BayesC_SL4","RKHS_SL4"
)
# Predictive ability
(summary<-data.frame(PA_mean=sapply(Acc,function(x) round(mean(x),digits = 3)),
PA_SE=sapply(Acc,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3))))
# Accuracy
#[1] "h2 of SL1 0.573"
#[1] "h2 of SL2 0.621"
#[1] "h2 of SL3 0.622"
#[1] "h2 of SL4 0.617"
h2_SL = 0.573
Acc1<-data.frame(GLBLUP_SL=Acc[,c(1,2)])
(summary<-data.frame(Acc_mean=sapply(Acc1,function(x) round(mean(x/sqrt(h2_SL)),digits = 3))),

```

```
Acc_SE=sapply(Acc1,function(x) round(sd(x/sqrt(h2_SL))/sqrt(repeats*n.fold),digits = 3))))
```

h2_SL = 0.621

```
Acc1<-data.frame(GLBLUP_SL=Acc[,c(4,5)])
```

```
(summary<-data.frame(Acc_mean=sapply(Acc1,function(x) round(mean(x/sqrt(h2_SL)),digits = 3))),
```

```
Acc_SE=sapply(Acc1,function(x) round(sd(x/sqrt(h2_SL))/sqrt(repeats*n.fold),digits = 3))))
```

h2_SL = 0.622

```
Acc1<-data.frame(GLBLUP_SL=Acc[,c(7,8)])
```

```
(summary<-data.frame(Acc_mean=sapply(Acc1,function(x) round(mean(x/sqrt(h2_SL)),digits = 3))),
```

```
Acc_SE=sapply(Acc1,function(x) round(sd(x/sqrt(h2_SL))/sqrt(repeats*n.fold),digits = 3))))
```

h2_SL = 0.617

```
Acc1<-data.frame(GLBLUP_SL=Acc[,c(10,11)])
```

```
(summary<-data.frame(Acc_mean=sapply(Acc1,function(x) round(mean(x/sqrt(h2_SL)),digits = 3))),
```

```
Acc_SE=sapply(Acc1,function(x) round(sd(x/sqrt(h2_SL))/sqrt(repeats*n.fold),digits = 3))))
```

5.2.2 R script for correlation between GEBVs for SL at different time points

The same codes were used for loading genotypes, phenotypes and genomic relationship matrix in Section 5.1

```
res1 <- kin.blup(data, K=A, geno="gid", pheno="SL1")
```

```
res2 <- kin.blup(data, K=A, geno="gid", pheno="SL2")
```

```
res3 <- kin.blup(data, K=A, geno="gid", pheno="SL3")
```

```
res4 <- kin.blup(data, K=A, geno="gid", pheno="SL4")
```

```
cor.test(res1$pred, res2$pred)
```

```
cor.test(res1$pred, res3$pred)
```

```
cor.test(res1$pred, res4$pred)
```

```
cor.test(res2$pred, res3$pred)
```

```
cor.test(res2$pred, res4$pred)
```

```
cor.test(res3$pred, res4$pred)
```

Section 5.3

5.3.1 R script for weighted genomic relationship matrix and GBLUP

```
# The same codes were used for loading genotypes, phenotypes and genomic relationship matrix in Section 5.1
library("biomaRt")
packageVersion("biomaRt")
ensembl98 <- useEnsembl(biomart = 'ensembl',
                           version = 98)
fugumart <- useDataset("trubripes_gene_ensembl",mart=ensembl98)

# https://www.kegg.jp/entry/pathway+tru00900
# Terpenoid backbone biosynthesis
tbb_entrezgene_id<-c("101073392", "101073997", "101063948", "101078757",
                      "101066905", "101076608", "101076038", "101079457",
                      "101062403", "101065291", "101070870", "105418230",
                      "101074637", "101069121", "101067613", "101071314",
                      "101076206", "101079983", "101079461", "101076024",
                      "101073203", "101073203", "101067237", "101065828")

# https://www.kegg.jp/entry/pathway+tru00100
# Steroid biosynthesis
sb_entrezgene_id<-c("101070689", "101073594", "101062227", "101062187",
                     "101069224", "115252790", "101077221", "101070460",
                     "101061359", "101072277", "101079626", "101071497",
                     "101075625", "101067117", "101071896", "115251974",
                     "101074906", "101069939", "101070169", "101064502",
                     "115250139", "101077060", "101072139", "105417935",
                     "101071742", "101062641", "101077963")

elist<-c(tbb_entrezgene_id,sb_entrezgene_id)
Resultlist<- getBM(attributes=c('ensembl_gene_id',"zfin_id_symbol","uniprot_gn_symbol",
                                 "chromosome_name","start_position","end_position"),
                     filters= 'entrezgene_id',
                     values= elist,
                     mart = fugumart)

str(tbb_entrezgene_id)
str(sb_entrezgene_id)
```

```

str(Resultlist)

lin_map["V3"]<-1:dim(lin_map)[1]
de_snp<-function(i,bp_plus,lin_map){
  re_bhat<-c()
  index<-lin_map$V1==Resultlist$chromosome_name[i]&lin_map$V2>=(Resultlist$start_position[i]-
  bp_plus)&lin_map$V2<=(Resultlist$end_position[i]+bp_plus)
  count_index<-sum(index)
  if (count_index==0){
    return(c())
  }
  else{
    re_bhat<-lin_map[index,]
    return(re_bhat)
  }
}

de_snp_list<-c()
for (i in seq(length(Resultlist$chromosome_name))){
  print(dim(de_snp(i,100000,lin_map))[1])
  de_snp_list<-rbind(de_snp_list,de_snp(i,100000,lin_map))
}
# de_snp_list$bhat<-round(abs(de_snp_list$bhat),4)
# summary(de_snp_list,maxsum = 22)

str(de_snp_list)

weighted_gp<-function(w){
  p <- apply(x+1, 2, mean)/2
  P <- matrix(rep(p*2, nrow(x)), ncol=ncol(x), nrow=nrow(x), byrow=TRUE)
  rownames(P) <- rownames(x)
  colnames(P) <- colnames(x)
  Z <- x+1 - P
  # sum 2pq to scale G to the A matrix
  q <- 1 - p
  sum2pq <- 2*sum(p*q) # note you can pull out the 2 (redundant)
  # G <- (Z %*% t(Z)) / sum2pq # A.mat
  #D <- diag(1 / (ncol(x)*(2*p*q)))
}

```

```

D <- diag((2*p*q))

# column and rownames
colnames(D) <- colnames(x)
rownames(D) <- colnames(x)

for (i in lin_map[de_snp_list$V3,]$V3){
  D[i,i]<-D[i,i]*w
}
G <- Z %*% D %*% t(Z)
registerDoParallel(12)
system.time({
  GBLUP_SL4_kegg<-foreach(j=1:repeats,.combine = "rbind") %do% {
    set.seed(100+3*j+1)
    id <- sample(1:n.sample %% n.fold) + 1
    foreach(i=1:n.fold,.packages="rrBLUP") %dopar% {
      bcw_test <- data
      bcw_test$SL4[id == i] <- NA
      res <- kin.blup(bcw_test, K=G, geno="gid", pheno="SL4")
      cor(data$SL4[id==i],res$pred[id==i])
    }
  }
})
stopImplicitCluster()

Acc<-data.frame(unlist(GBLUP_SL4_kegg))
colnames(Acc)<-c("GBLUP_SL4_kegg")
# Predictive ability
summary<-data.frame(PA_mean=sapply(Acc,function(x) round(mean(x),digits = 3)),
PA_SE=sapply(Acc,function(x) round(sd(x)/sqrt(repeats*n.fold),digits = 3)))
return(summary)
}

G1<-weighted_gp(1)
G1
G2<-weighted_gp(2)
G2
G5<-weighted_gp(5)

```

G5

G10<-weighted_gp(10)

G10

5.3.2 R script for effects of SNPs underling genes related to terpenoid backbone biosynthesis and steroid biosynthesis

```
# The same codes were used for loading genotypes, phenotypes and genomic relationship matrix in Section 5.1  
fmBC_SL4=BGLR(y=data$SL4,ETA=list(list(X=x,model='BayesC')),  
    nIter=10000,burnIn=2000,verbose = FALSE)  
#      fmBC$ETA[[[]]$b][as.numeric(GWAS_length$marker[head(order(GWAS_length$length,decreasing  
    TRUE),3)])]  
  
mean(abs(fmBC_SL4$ETA[[[]]$b))  
sd(abs(fmBC_SL4$ETA[[[]]$b))  
  
mean(abs(fmBC_SL4$ETA[[[]]$b)[lin_map[de_snp_list$V3,]$V3]))  
sd(abs(fmBC_SL4$ETA[[[]]$b)[lin_map[de_snp_list$V3,]$V3]))
```