Homework 4

Sudha

2023-04-28

```
source("http://zzlab.net/GAPIT/gapit_functions.txt")
library(ggplot2)
library(e1071)
library(BLR)
library(BGLR)


# read SNPs data file
myGD=read.table(file="geno_numeric.txt", head=T, check.names = FALSE)

# Marker postion
myGM=read.table(file="snp_info.txt", head=T, check.names = F)

dim(myGM)
dim(myGD)

# write function


gs_mas = function(h2, NQTN){
  # Simulate phenotype using QTN

  mySim =GAPIT.Phenotype.Simulation(
    GD=myGD,GM=myGM,
    h2=h2,NQTN=NQTN,
    QTNDist="normal")

  # split both genotype and phenotype to training and testing set

  n_row=nrow(myGD) # no. of lines
  test=sample(n_row,round(n_row/5),replace=F)
  train=-test

  X.train  <- myGD[train,] #training genotype
  y.train = mySim$Y[train,] # training phenotype
  y.test = mySim$Y[test,] #testing phenotype
  #dim(y.train);dim(y.test)

  # run GWAS using BLINK
  gwas_blink <- GAPIT(Y = y.train,
                      GD = myGD,GM = myGM,PCA.total = 3,
                      QTN.position=mySim$QTN.position,
```

```
                    model = "BLINK",
                    file.output = FALSE,
                    memo = 'GWAS')
myCV=gwas_blink$PCA
sig_snps_pos <- which(gwas_blink$GWAS$P.value <= 0.05/length(gwas_blink$GWAS$P.value))
sig_snps_pos = sig_snps_pos +1
sig_snps = myGD[,(sig_snps_pos)]
myQTN <- cbind(gwas_blink$PCA,sig_snps )

#thresh = 0.05/length(gwas_blink$GWAS$P.value)
#index = order(gwas_blink$GWAS$P.value, decreasing = F)
#myQTN=cbind(gwas_blink$PCA, myGD[index[1:15]])

# MAS using the markers from blink (glm)

mas_blink <- GAPIT(
  Y=y.train,
  CV=myQTN,
  model="GLM",
  SNP.test=FALSE,
  memo="MAS +glm",
  file.output = F)

order=match(mySim$Y[,1],mas_blink$Pred[,1])
myPred=mas_blink$Pred[order,]

train_u = mySim$u[train]
train_y = y.train[2]
train_pred_mas_glm = myPred[train,8]
test_y = y.test[2]
test_u = mySim$u[test]
test_pred_mas_glm = myPred[test,8]

train_cor_mas_glm = cor(train_u, train_pred_mas_glm)^2
test_cor_mas_glm = cor(test_u, test_pred_mas_glm)^2

#MAS plus gBLUP

mas_gblup <- GAPIT(
  Y=y.train,
  GD=myGD, GM=myGM,
  CV=myQTN,
  model="gBLUP",
  file.output = F,
  SNP.test=FALSE,
  memo="MAS+gBLUP")

order=match(mySim$Y[,1],mas_gblup$Pred[,1])
myPred=mas_gblup$Pred[order,]

train_pred_mas_gblup = as.numeric(myPred[train,8])
test_pred_mas_gblup = as.numeric(myPred[test,8])
```

```
  train_cor_mas_gblup = cor(train_u, train_pred_mas_gblup)^2
  test_cor_mas_gblup = cor(test_u, test_pred_mas_gblup)^2

  # genomic selection (gBLUP)

  gs_gblup <- GAPIT(Y=y.train,
                    GD=myGD, GM=myGM, PCA.total=3,
                    file.output = F,
                    SNP.test = F,
                    model="gBLUP",
                    memo = 'gBLUP')

  order=match(mySim$Y[,1],gs_gblup$Pred[,1])
  myPred=gs_gblup$Pred[order,]

  train_pred_gs = myPred[train,5]
  test_pred_gs = myPred[test,5]

  train_cor_gs = cor(train_u, train_pred_gs)^2
  test_cor_gs = cor(test_u, test_pred_gs)^2

  #Regression with SVM

  df_train = cbind(y.train[2],myCV[train,-1], myGD[train,-1])
  df_test = cbind(y.test[,2],myCV[test,-1], myGD[test,-1])
  #run svm
  modelsvm = svm(Sim ~ ., data = df_train)

  #Predict using SVM regression
  pred_test_svm = predict(modelsvm, newdata = df_test)
  pred_train_svm = predict(modelsvm, newdata = df_train)

  train_pred_svm = as.numeric(pred_train_svm)
  test_pred_svm = as.numeric(pred_test_svm)

  train_cor_svm = cor(train_y, train_pred_svm)^2
  test_cor_svm = cor(test_y, test_pred_svm)^2


  results = list(train_cor_mas_glm = train_cor_mas_glm, test_cor_mas_glm = test_cor_mas_glm,
                 train_cor_mas_gblup = train_cor_mas_gblup, test_cor_mas_gblup = test_cor_mas_gblup,
                 train_cor_gs = train_cor_gs, test_cor_gs = test_cor_gs,
                 train_cor_svm = train_cor_svm, test_cor_svm = test_cor_svm,
                 train_y = train_y, train_u = train_u ,train_pred_mas_gblup = train_pred_mas_gblup,
                 train_pred_mas_glm = train_pred_mas_glm, train_pred_gs= train_pred_gs,
                 train_pred_svm = train_pred_svm,
                 test_y = test_y, test_u = test_u, test_pred_mas_gblup = test_pred_mas_gblup,
                 test_pred_mas_glm = test_pred_mas_glm, test_pred_gs= test_pred_gs,
                 test_pred_svm = test_pred_svm)
  return(results)
}
```

```
results_25_10 = replicate(30, gs_mas(0.25, 10))
results_50_10 = replicate(30, gs_mas(0.50, 10))
results_75_10 = replicate(30, gs_mas(0.75, 10))

results_25_25 = replicate(30, gs_mas(0.25, 25))
results_50_25 = replicate(30, gs_mas(0.50, 25))
results_75_25 = replicate(30, gs_mas(0.75, 25))


results_25_50 = replicate(30, gs_mas(0.25, 50))
results_50_50 = replicate(30, gs_mas(0.50, 50))
results_75_50 = replicate(30, gs_mas(0.75, 50))


results_25_100 = replicate(30, gs_mas(0.25, 100))
results_50_100 = replicate(30, gs_mas(0.50, 100))
results_75_100 = replicate(30, gs_mas(0.75, 100))

results_25_500 = replicate(30, gs_mas(0.25, 500))
results_50_500 = replicate(30, gs_mas(0.50, 500))
results_75_500 = replicate(30, gs_mas(0.75, 500))



library(tidyverse)
#transpose data (makes it easier to graph)
gs_mas_replicate<-t(results_75_500)
cor_only = as.data.frame(gs_mas_replicate[1:30,1:8])
cor_only <- as.data.frame(lapply(cor_only, function(x) unlist(x)))
head(cor_only)

# Melt the data frame using gather()
df_melted <- gather(cor_only, key = "Method", value = "Correlation", 1:8)
df_melted$heritability = "0.75"
df_melted$no_QTN = "500"

tail(df_melted)
#df= df_melted
df = rbind(df, df_melted)
dim(df)

df_split <- separate(df, Method, into = c("dataset", "Method"), sep = "_cor_")
df_split$x_ordered <- factor(df_split$no_QTN, levels = c("10", "25", "50", "100", "500"))
df_split$dataset_ordered <- factor(df_split$dataset, levels = c("train", "test"))
head(df_split)
write.csv(df_split, 'final_results_statgenomics.csv')


jpeg('boxplot_gs_mas.jpg',width=12,height=8,units='in',res=300)
d = ggplot(df_split)+
  geom_boxplot(aes(x = x_ordered, y = Correlation, color = dataset_ordered))+
  facet_grid(Method~heritability)+
  scale_colour_manual(values = c("tomato",'darkolivegreen4'))+
```

```
  xlab('No. of QTNs')+ ylab('Correlation')+
  theme(strip.text.x = element_text(size = 14),
          strip.text.y = element_text(size = 14),
          legend.text = element_text(size =12),
          legend.title = element_text(size =14))+
  guides(color = guide_legend('dataset', override.aes = list(alpha =1)))
d

dev.off()


# Results


#transpose data (makes it easier to graph)
gs_mas_replicate<-t(results_75_500)
cor_only = gs_mas_replicate[1:30,1:8]

avg_cor = apply(matrix(as.numeric(cor_only), nrow =3 , ncol = 8), 2, mean)
sd_cor = apply(matrix(as.numeric(cor_only), nrow =3 , ncol = 8), 2, sd)

summary_cor = data.frame(rbind(avg_cor, sd_cor))
names(summary_cor) = c("train_cor_mas_glm", "test_cor_mas_glm" ,
                       "train_cor_mas_gblup", "test_cor_mas_gblup",
                       "train_cor_gs", "test_cor_gs",
                       "train_cor_svm" ,"test_cor_svm")
summary_cor$heritablity = "0.75"
summary_cor$no_QTN = "500"
summary_cor

summary_cor_1 = summary_cor[1,]
summary_cor_2 = summary_cor[2,]
# convert wide data frame to long format
my_long_df_1 <- melt(summary_cor_1, id.vars = c('heritablity', 'no_QTN'),
                     variable.name = "Method", value.name = "correlation")
my_long_df_2 <- melt(summary_cor_2, id.vars = c('heritablity', 'no_QTN'),
                     variable.name = "Method", value.name = "sd")
# print the long data frame
df_transit = cbind(my_long_df_1, my_long_df_2[4])

#df_all = df_transit
df_all = rbind(df_all, df_transit)
dim(df_all)

df_split_1 <- separate(df_all, Method, into = c("dataset", "Method"), sep = "_cor_")
df_split_1$x_ordered <- factor(df_split_1$no_QTN, levels = c("10", "25", "50", "100", "500"))
df_split_1$dataset_ordered <- factor(df_split_1$dataset, levels = c("train", "test"))
head(df_split_1)
write.csv(df_split_1, 'final_correlation_results_statgenomics.csv')


jpeg('boxplot_correlation_only.jpg',width=12,height=8,units='in',res=300)
d = ggplot(df_split_1)+
```

```
  geom_point(aes(x = as.numeric(no_QTN), y = correlation,
                 color = dataset_ordered), color = 'black',size = 2)+
  geom_line(aes(x = as.numeric(no_QTN), y = correlation,
                 color = dataset_ordered), size = 1)+
  facet_grid(Method~heritablity)+
  scale_colour_manual(values = c("tomato",'darkolivegreen4'))+
  xlab('No. of QTNs')+ ylab('Correlation')+
  theme(strip.text.x = element_text(size = 14),
        strip.text.y = element_text(size = 14),
        legend.text = element_text(size =12),
        legend.title = element_text(size =14))+
  guides(color = guide_legend('dataset', override.aes = list(alpha =1)))
d

dev.off()



jpeg('boxplot_standarddeviation_only.jpg',width=12,height=8,units='in',res=300)
d = ggplot(df_split_1)+
  geom_point(aes(x = as.numeric(no_QTN), y = sd,
                  color = dataset_ordered), color = 'black',size = 2)+
  geom_line(aes(x = as.numeric(no_QTN), y = sd,
                 color = dataset_ordered), size = 1)+
  facet_grid(Method~heritablity)+
  #scale_colour_manual(values = c("tomato",'darkolivegreen4'))+
  xlab('No. of QTNs')+ ylab('Standard deviation')+
  theme(strip.text.x = element_text(size = 14),
        strip.text.y = element_text(size = 14),
        legend.text = element_text(size =12),
        legend.title = element_text(size =14))+
  guides(color = guide_legend('dataset', override.aes = list(alpha =1)))
d
dev.off()



jpeg('boxplot_correlation_only.jpg',width=12,height=8,units='in',res=300)
d = ggplot(df_split_1)+
  geom_point(aes(x = as.numeric(no_QTN), y = correlation,
                  color = Method), color = 'black',size = 2)+
  geom_line(aes(x = as.numeric(no_QTN), y = correlation,
                 color = Method), size = 1)+
  facet_grid(dataset_ordered~heritablity)+
  #scale_colour_manual(values = c("tomato",'darkolivegreen4'))+
  xlab('No. of QTNs')+ ylab('Correlation')+
  theme(strip.text.x = element_text(size = 14),
        strip.text.y = element_text(size = 14),
        legend.text = element_text(size =12),
        legend.title = element_text(size =14))+
  guides(color = guide_legend('Method', override.aes = list(alpha =1)))
d
```

```
dev.off()



jpeg('boxplot_standarddeviation_only.jpg',width=12,height=8,units='in',res=300)
d = ggplot(df_split_1)+
  geom_point(aes(x = as.numeric(no_QTN), y = sd,
                 color = Method), color = 'black',size = 2)+
  geom_line(aes(x = as.numeric(no_QTN), y = sd,
                color = Method), size = 1)+
  facet_grid(dataset_ordered~heritablity)+
  #scale_colour_manual(values = c("tomato",'darkolivegreen4'))+
  xlab('No. of QTNs')+ ylab('Standard deviation')+
  theme(strip.text.x = element_text(size = 14),
        strip.text.y = element_text(size = 14),
        legend.text = element_text(size =12),
        legend.title = element_text(size =14))+
  guides(color = guide_legend('Method', override.aes = list(alpha =1)))
d
dev.off()




# Bayesian LASSO

library(BLR)
library(BGLR)

nIter=2000              #### number of iteration
burnIn=1500 #### burnin a part of iteration

myBLR =BLR(y=as.matrix(y.train[2]),
           XF=myCV[train,-1],
           XL=as.matrix(myGD[train,-1]),
           nIter=nIter,
           burnIn=burnIn)

pred.inf=as.matrix(myGD[test,-1])%*%myBLR$bL
ru2 <- cor(mySim$u[test],pred.inf)^2
plot(mySim$u[test],pred.inf)
mtext(paste("R square=",ru2,sep=""), side = 3)


# BGLR- BayesB

myBGLR =BGLR(y=as.matrix(y.train[2]),
             ETA=list(list(X=myCV[train,-1],model='FIXED',saveEffects=TRUE),
                      list(X=as.matrix(myGD[train,-1]),model='BayesB',saveEffects=TRUE)),
             nIter=nIter,
             burnIn=burnIn)

pred.inf=as.matrix(myGD[test,-1])%*%myBGLR$ETA[[2]]$b
```

```r
ru2 <- cor(mySim$u[test],pred.inf)^2
plot(mySim$u[test],pred.inf)
mtext(paste("R square=",ru2,sep=""), side = 3)


#df_wide <- df_split_1 %>%
  #pivot_wider(names_from = Method, values_from = c(correlation, sd)) %>%
 # select(-contains("dataset"))

#dim(df_wide)
#write.csv(df_wide, 'df_wide.csv')
```