Package JKGWAS User Tutorial

Pabitra Joshi and Lindsey Kornowske

21 March 2021

Contents

Joshi Kornowske Genome Wide Association Study (JKGWAS)

Introduction

The JKGWAS package provides tools to conduct Genome Wide Association Study (GWAS) by generalized linear model (GLM) and to visualize the results. GWAS by GLM improves the true positive detection rate relative to GWAS by correlation and the functions provided in this package allow the user to model known covariates and principal components as fixed effects in addition to the numeric genomic data in order to improve phenotype prediction. Where applicable, issues related to modeling non-invertable matrices are automatically solved by detecting Principal Components with linear dependence on the covariates. GWAS results are easily visualized by QQ and Manhattan plot functions.

Description of package functions

This package "JKGWAS" contains a list of functions that you can find in the git repository and description of which is also mentioned in "JKGWAS_0.1.0.pdf". All the functions have their description, arguments and parameters clearly mentioned. When you install the package from github you can see the function and their description just by typing ??function_name in the R-console. This will let you know that what are the input parameters you need to give to get the output.

Getting started

The devtools package is required to download the JKGWAS package from Github. If the user has not installed devtools, they should do so by running the first line in the chunk below.

```r
#install.packages("devtools", dependencies = TRUE);
library(devtools);
```

Once devtools has been installed, the install_github() function can be used to access the JKGWAS package. The following chunk will do this automatically.

```r
#install JKGWAS
install_github("lindseymaek/HORT545/JKGWAS");
library(JKGWAS);
```

The JKGWAS package accepts genomic, phenotype, and covariate data, and visualizes these with genomic map data. Sample datasets may be found in the lindseymaek/HORT545 git repository, one level above the JKGWAS package. All credit for the curation of these datasets belongs to the Dr. Zhiwu Zhang laboratory; for more information on these data, the user should go to . The data are as follows:

- CROPS545_Covariates.txt - Covariate (CV) data, with two factors
- CROPS545_Phenotype.txt - Phenotype data
- mdp_SNP_information.txt - Genomic map data
- mdp_numeric.txt - Numeric genotype data, with column-wise snps

Once downloaded to a local directory, import the data as follows:

```r
## define the variable userDataDirectory as the path to the folder in which the datasets are stored
userDataDirectory = c("Users/JKGWASdata/examplepath/")

# covariate data
CV = read.csv(file = "userDataDirectory/CROPS545_Covariates.txt", header = TRUE, sep = "");

# genomic map data
SNP = read.csv(file = "userDataDirectory/mdp_SNP_information.txt", header = TRUE, sep = "");

# numeric genotype data
X = read.csv(file = "userDataDirectory/mdp_numeric.txt", header = TRUE, sep ="");

# phenotype data
y = read.csv(file = "userDataDirectory/CROPS545_Phenotype.txt", header = TRUE, sep = "");
```

The variable names used here correspond to the names of the arguments in the JKGWAS functions. This is the data that will be used for the remainder of this tutorial.

In order to simulate phenotype data with QTNs for a desired heritability value, we also recommend that you utilize the G2P() function also provided by the Zhang laboratory. This will be used later on in the tutorial.

```
#source G2P
source("http://www.zzlab.net/StaGen/2021/R/G2P.R")
```

Required Inputs

There are three data components required to run the JKGLM() and JKPCA() functions in JKGWAS, the SNP/marker data, the phenotype data, and the genetic map data. The map data should contain columns having chromosome number and positions of each SNP. The JKGLM function will calculate the pvalues for each SNP as modeled by the user-inputs, these p-values are required inputs for, the JKQQ and JKManhattan functions.

• **SNP/marker data** - The SNP data must be numeric coded as "0", "1" and "2" where the 0 stands for homozygous for parent A, 2 stands for homozygous for parent B and 1 is for heterozygous SNP calls. Below is an example showing the first 5 rows and columns of our tutorial SNP data.

```
X[1:5,1:5] #SNP/marker data
```

```
##    taxa PZB00859.1 PZA01271.1 PZA03613.2 PZA03613.1
## 1 33-16          2          0          0          2
## 2 38-11          2          2          0          2
## 3  4226          2          0          0          2
## 4  4722          2          2          0          2
## 5  A188          0          0          0          2
```

• **Phenotype data** - The phenotypic data should be integers, negative or positive that correspond to each individual.To run the JKGLM function you should first remove the taxa column of the phenotypic data. Below is the first 5 column and row of phenotypic data we used in our tutorial.

```
y[1:5,1:2]#phenotypic data
```

```
##    Taxa        Obs
## 1 33-16 -1.2731037
## 2 38-11  0.5515271
## 3  4226 -0.2549273
## 4  4722 -6.1229643
## 5  A188 -2.5939825
```

• **Genetic Map Data** - The other information needed to run JKGLM package is genetic map of the SNPs. The heading of this map data should have marker name,chromosome number and chromosome position.This piece of information will be needed to make manhattan plot.Below is the first 5 row of map data.

```
SNP[1:5,1:3]# map data
```

```
##          SNP Chromosome Position
## 1 PZB00859.1          1   157104
## 2 PZA01271.1          1  1947984
## 3 PZA03613.2          1  2914066
## 4 PZA03613.1          1  2914171
## 5 PZA03614.2          1  2915078
```

Optional Inputs

• **Covariate data** - Finally we have included covariate data in the package. The covariates mush be integers much like the phenotype values. If you do not provide the covariate information then it can also run without the information by using other inputs. The covariate data should have information on factors. First r row of the covariate data id shown below.

```
CV[1:5,1:3] #Covariate data
```

```
##    Taxa  FactorA  FactorB
## 1 33-16 2.531331 5.501464
## 2 38-11 2.633860 4.655691
## 3  4226 1.890695 6.136883
## 4  4722 1.856035 7.841858
## 5  A188 2.552629 5.409450
```

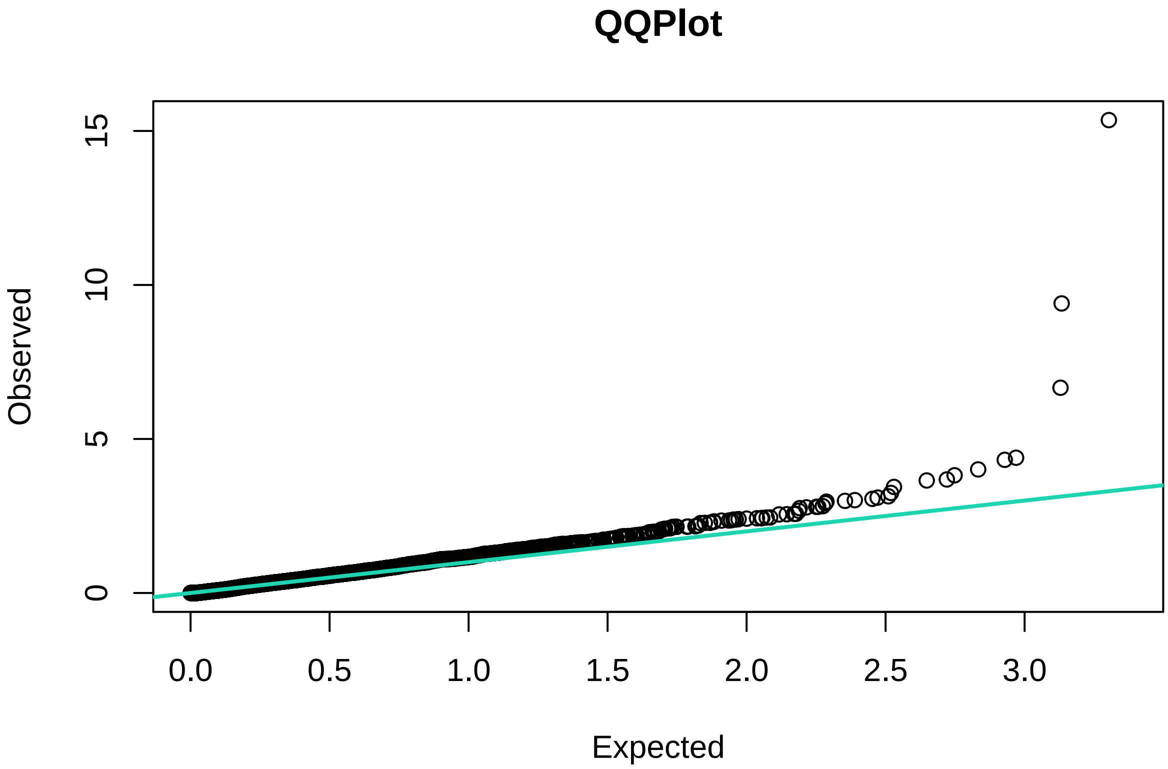
- **Principal Components** - Other imputs include the Principal Components, which can be generated from JKPCA(), or be provided by the user. The components must be organized columnwise, with an equal number of observations as X. The user can also determine the number of principal components to use, with the default being set to five. The significance threshold (Cutoff) which can be set to the exact -log(10) of the p-value you want or the default of 0.05/number of SNPs (Bonferroni Correction). There are also some options you can use to suppress the plots automatically generated.

- **QTN** - The user can provide a vector of the positions of the known QTNs, in order to visualize the true positive detection rate by Manhattan plot with JKManhattan().

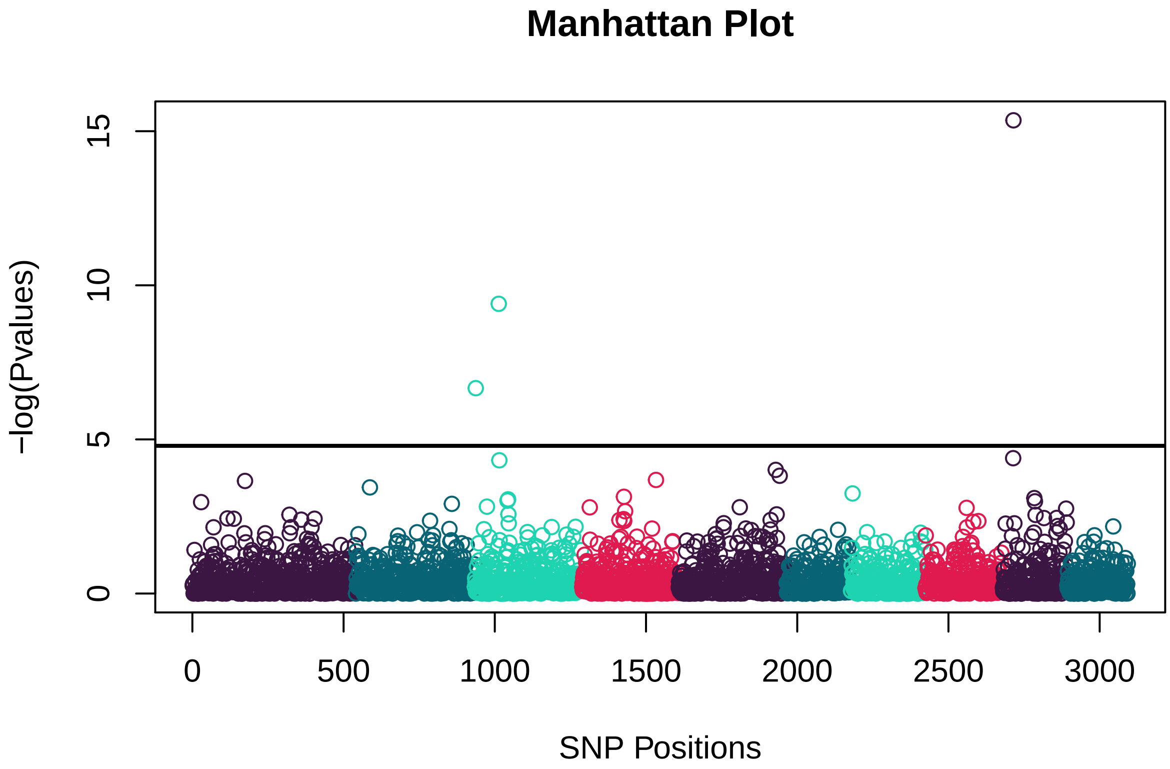
Outputs and Examples

After you use the JKGWAS package,the outputs you get includes information on principal components, p-values of the SNPs, manhattan plot, QQplot.We can also include the false positives as the output.Examples of the output are shown below.

```
## Get Principal Components with JKPCA()
PC = JKPCA(X, CV, npc = 10);
## Perform GWAS by GLM with JKGLM()
Pvals = JKGLM(X = X, y = y, CV, PC);
## Visualize GWAS by QQ Plot with JKQQ()
JKQQ(Pvals);
```



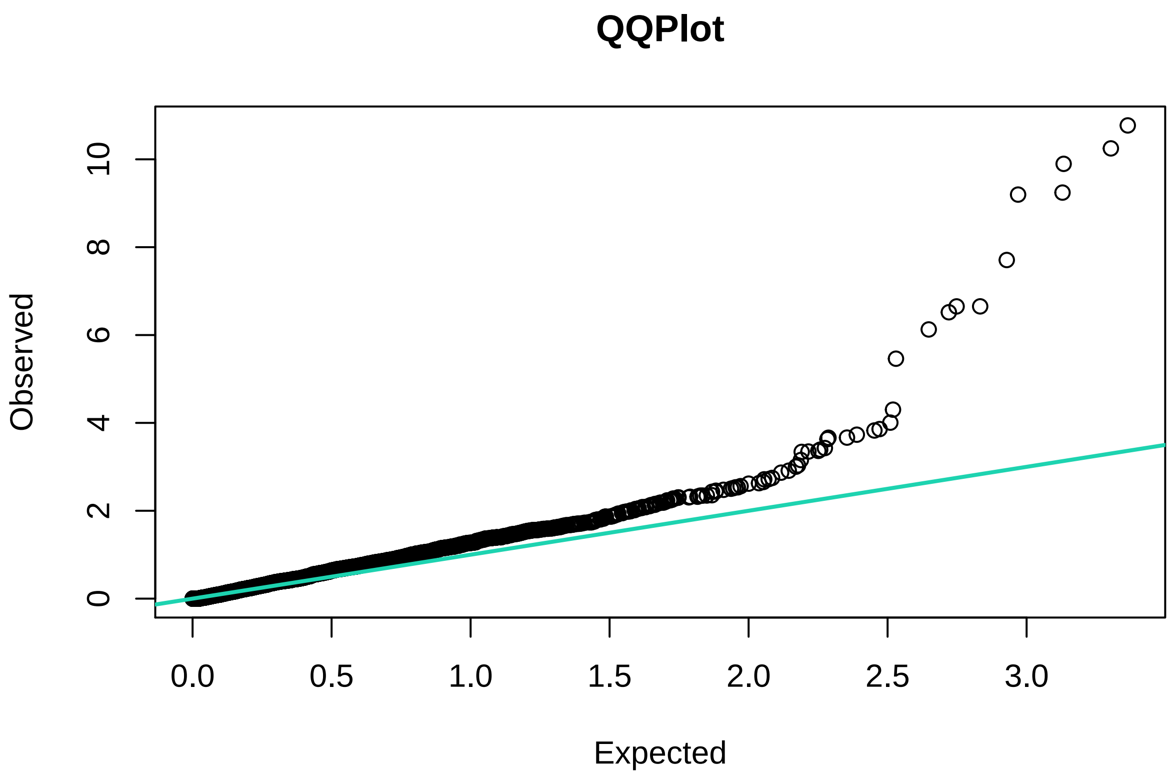
```
## Visualize GWAS by Manhattan Plot with JKManhattan()
JKManhattan(Pvals = Pvals, SNP = SNP,sigcutoff = NULL );
```



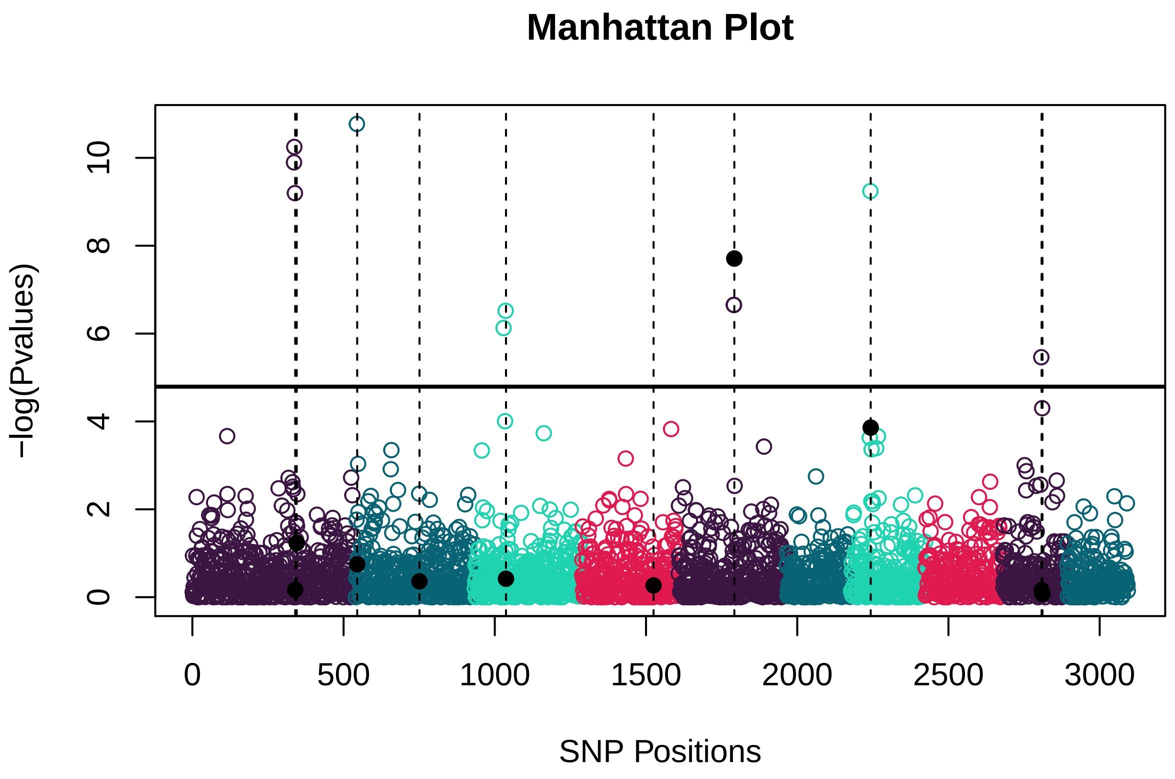
Examples with known QTNs

If the user knows the QTNs, they can provide them to the JKManhattan() function. Alternatively, the G2P() function can be used to simulate QTNs.

```
G2P.sim = G2P(X= X,
              h2= 0.75,
              alpha=1,
              NQTN=10,
              distribution="norm");

G2P.y = as.data.frame(G2P.sim$y);

Pvals.sim = JKGLM(X = X, y = G2P.y, CV, PC);

## Visualize GWAS by QQ Plot with JKQQ()
JKQQ(Pvals.sim);
```



```r
## Visualize GWAS by Manhattan Plot with JKManhattan()
JKManhattan(Pvals = Pvals.sim, SNP = SNP, sigcutoff = NULL, QTN = G2P.sim$QTN.position );
```



Frequently asked questions

1. Where can I get JKGWAS?

JKGWAS is currently available on github via Lindsey Kornowske's public repository found at https://github.com/lindseymaek/HORT545

2. How do I download JKGWAS?

Please refer the "Getting started" section above where I show how to download and install JKGWAS using devtools.

3. What SNP format does JKGWAS take?

Currently JKGWAS functions only accept genomic SNP data in the numerical format. Non-numeric columns will be automatically detected and removed by the functions.

4. What are the required inputs to run JKGWAS?

You will need SNP data in numerical form, phenotypic data and a genetic map that has SNP name, Chromosome and position on chromosome. Covariate data is optional ,but the other three components are prerequisite to run JKGLM. The output of JKPCA(), can be used as an input for the PC argument of JKGLM(), and the output of JKGLM() can be used as the input for the Pvals argument of JKQQ() and JKManhattan().

5. Where can I get help with issues regarding JKGWAS

You can contact the two developer of this package Lindsey Kornowske, at lindsey.kornowske@wsu.edu and Pabitra Joshi at pabitrajoshi77@gmail.com if you have any questions.

6. Are there other R packages that can perform GWAS?

Yes, the GAPIT package, authored by the Dr. Zhiwu Zhang laboratory can be found at http://www.zzlab.net/GAPIT/ and has many advanced tools for GWAS and visualization.

Further Information

More information about GWAS, including sample data, publications, R packages, and more, can be found at the Zhiwu Zhang Laboratory Website: http://www.zzlab.net/index.html If you are a StaGen545 student from the future - good luck on homework 4, we challenge you to out-awesome the JKGWAS package logo.