The *fGWAS* Package

Version 0.20

Zhong Wang

1. Introduction

The *fGWAS* package aims to identify significant SNPs that control longitudinal phenotypic traits and estimate their additive and dominant genetic effects based on the Functional Mapping model. This model is a cornerstone for identifying the relation between genes and longitudinal traits in this *fGWAS* package. This guide gives a brief instruction on how to perform the tasks of SNP identification by the *fGWAS* package. The outline of this guide is as follows:

We refer to Luo et al. (2010) and Das et al (2011) for the theoretical foundation of this package. If you benefit from this software, please cite the following papers in your work:

1. Luo, J., Berg, A., Ahn, K., Das, K., Li, J., Wang, Z., ... & Wu, R. (2010). Functional Genome-Wide Association Studies of Longitudinal Traits. In Handbook of Adaptive Designs in Pharmaceutical and Clinical Development (pp. 23-1). CRC Press..

2. Das, K., Li, J., Wang, Z., Tong, C., Fu, G., Li, Y., ... & Wu, R. (2011). A dynamic model for genome-wide association studies. Human genetics, 129(6), 629-639.

2. Installation

The *fGWAS* package depends on the ***snpStats***, ***mvtnorm***, ***lme4***, ***nlme***, ***minpack.lm,*****snowfall,** and ***parallel*** (R<2.14.0) packages (specifically, the ***parallel*** package is included since R 2.14.0), so these packages should be installed firstly. To install *fGWAS*, download the package file and type the appropriate command in a Linux terminal or a R console window.

1) Linux terminal

```
$ R CMD INSTALL fGWAS_0.20.tar.gz
```

2) R console window in Linux environment

```
>install.packages("fGWAS_0.20.tar.gz")
```

3) For Windows users, please download Windows version and install the package through the menu item "Install package(s) from local zip files".

Before the package works for your computation jobs, the package importation is necessary by the following R command:

```
>library(fGWAS)
```

3. Statistical model

The *fGWAS* model has been elaborated in Luo 2010 and Das 2011. Here we review some equations to explain its mechanism.

3.1 The *fGWAS* model

In the *fGWAS* model, a number of important covariates, which are either discrete or continuous, along with additive and dominant effects are integrated into one statistical framework. Let $y_i = (y_i(t_{i1}), \ldots, y_i(t_{im}))$ denotes the vector of trait values for subject *i* measured at multiple times $T_i = (t_{i1}, \ldots, t_{im})$. Considering a SNP *S* with 2 alleles (*A* and *a*) and 3 genotypes: *AA* (coded by 0), *Aa* (coded by 1), and *aa* (coded by 2), the response value y_i measured at multiple time points for subject *i* is dissected in equation (1),

$$\widetilde{y_i}(\mathrm{T_i}) = \mu + \mathrm{X_i'}\alpha + \sum_{\mathrm{j}=0}^{2} \xi(\mathrm{j} == \mathrm{S_i}\ ) * \mathrm{g_j}(\mathrm{T_i}) + e_i(\mathrm{T_i}) + \varepsilon_i(\mathrm{T_i}), \tag{1}$$

where μ is the overall mean, X_i is the $p \times 1$ vector of discrete or continuous covariates, α is the vector of regression coefficients of *p* covariates, ξ is the indicator function of the effects of SNPs, $g_j(T_i)$ is the mean value for genotype *j* at time T_i, and $e_i(T_i)$ and $\varepsilon_i(T_i)$ are the permanent and random errors at time T_i respectively. The permanent errors together with random errors are called the residual errors, which are assumed to follow a multivariate normal distribution with zero mean vector and some kinds of covariance structure.

$$\xi(\mathrm{j} == \mathrm{S_i}\ ) \begin{cases} 1, \text{if the genotype of SNP } \mathrm{S_i} \text{ is coded as j} \\ 0, \text{if the genotype of SNP } \mathrm{S_i} \text{ is NOT coded as j} \end{cases} \tag{2}$$

The significance test of the genetic effect of a SNP empirically can be performed by the hypothesis test based on the equation as follows:

$$\begin{cases} \mathrm{H0}: \mathrm{g_0} == \mathrm{g_1} == \mathrm{g_2} \\ \mathrm{H1}: \text{At least one equality in the } \mathrm{H_0} \text{ does not hold} \end{cases} \tag{3}$$

In general, the test statistic for hypothesis (3) is a log-likelihood ratio (LR) test, which compares the difference of likelihood values between the hypothesis of no genetic curve and the hypothesis of genetic curves based on the Maximum Likelihood Estimate (MLE) method defined by equation(4). In equation (4), L_0 and L_1 indicate the likelihood values under the null hypothesis and the alternative hypothesis.

$$LR = -2log(\frac{L_0}{L_1}) \tag{4}$$

The Maximum Likelihood Estimate is a convenient method to do parameter estimation under the null or alternative hypothesis. In *fGWAS*, we use the multivariate normal distribution to build the likelihood function in equation (5) and (6).

$$\mathrm{L_0(y)} = \prod_{\mathrm{i}=1}^{\mathrm{n0+n1+n3}} \mathrm{f}(y_i) \tag{5}$$

$$\mathrm{L_1(y)} = \prod_{\mathrm{i}=1}^{\mathrm{n0}} \mathrm{f_0}(y_i) \prod_{\mathrm{i}=1}^{\mathrm{n1}} \mathrm{f_1}(y_i) \prod_{\mathrm{i}=1}^{\mathrm{n2}} \mathrm{f_2}(y_i) \tag{6}$$

The function of multivariate normal distribution contains two important parts, residual vector and covariance structure. We decompose the residual vector and expand the function in equation (7),

$$f_j(y_i) = \frac{1}{(2\pi)^{\frac{m}{2}}|\Sigma|^{\frac{1}{2}}} e^{[-(y_i-\mu-X_i'\alpha-g_j(T_i))'\Sigma^{-1}(y_i-\mu-X_i'\alpha-g_j(T_i))/2]} \quad (7)$$

where Σ is the covariance structure and g_j is the mean vector of genetic curve. The *fGWAS* package implements 9 curves in the current version, including the logistic growth curve, the nonparametric method, and the composite curve. Appendix A shows the details of the 9 curves. Covariance structure presents the correlation between measured values at multiple time points. In the *fGWAS* package, 13 covariance structures have been implemented, including the most frequently used structures, such as "AR1" and "SAD1". Appendix B shows the details of the 13 covariance structures.

MLE algorithm is employed to estimate the likelihood values and all the parameters, including the covariate coefficients, the covariance parameters, and the curve parameters for each genotype. After *fGWAS* estimates the parameters for each SNP and calculates the LR values, significant SNPs can be identified with the aid of the specific p-value criteria.

3.2 Computation methods

Traditionally, MLE is employed to calculate the Likelihood Ratio under the null hypothesis and the alternative hypothesis, stated as equation (6) and (7). The stringent method to calculate the LR and estimate the parameters is to get the global optimized value for all parameters, including the biological curve and the covariance structure in one likelihood function. Because it is hard to select appropriate initial values and quickly approach to the global maximum point, the optimization is thought to be a time-consuming and unpredictable process. *fGWAS* implements this stringent method using the *optim* function in R, where the calculation process is very slow when MLE solves the estimation for all the parameters simultaneously, including the covariates, curve, and covariance parameters.

In order to reduce computation time, we propose a faster method, which first estimates the covariate and curve parameters together using the method of least squares, and then estimates the covariance parameters using MLE. For those insignificant SNPs, which indicate no genetic difference between 2 genotypes or among 3 genotypes, the LR results generated by the 'fast' method roughly approximate the results of the stringent method, but for the significant SNPs, two methods give almost identical results. We think it is reasonable to ignore insignificant SNPs and focus on significant ones in the view of GWAS.

4. Data format

The *fGWAS* package is aiming to detect the marginal genetic effects through the analysis of data files with appropriately formatted genotype and phenotype information. There are two genotypic data formats and two phenotypic data formats used in this package.

4.1. Phenotypic data

For the *fGWAS* analysis, the phenotypic data take the CSV (Comma-separated values) file containing individual identification numbers, covariates, as well as response values.

4.1.1. Longitudinal data file

The *fGWAS* model, which can estimate time varying curves for additive and dominant effects of each significant SNP, requires the longitudinal traits as phenotypic data. The example below lists one ID column (ID) and phenotype columns (Y_1, Y_2, Y_3, Y_4, Y_5) which stand for longitudinal phenotypic values. The missing data should be encoded as NA.

```
ID, Y_1,Y_2,Y_3,Y_4,Y_5
1, 18.853,14.289,11.529,15.920,22.203
2, 17.853,12.289,13.529,15.920,NA
…
```

4.1.2. Measured time file

The *fGWAS* package requires the **measured time file** for the longitudinal traits. The example below lists one ID column (ID) and measurement time columns (Z_1, Z_2, Z_3, Z_4, Z_5) which stand for measurement times.

```
ID, Z_1,Z_2,Z_3,Z_4,Z_5
1, 40, 42, 43, 48,50
2, 38, 39, 40, 41,NA
…
```

4.1.3. Covariate data file

The example below demonstrates a covariate data file containing ID and two covariates (X_1, X_2) .

```
ID,X_1,X_2
1,0,0.663,33.72
2,1,0.728,36.78
3,0,NA,38.92
…
```

The covariate data file should be only filled with numerical values. There are limited data checks in the current version, and the package requires the individual IDs in all phenotype files to be consistent with those in the genotype file. In this example, one row represents one subject. The covariate data file is optional but the phenotype longitudinal data file is required to call the functions of *fGWAS* model.

4.2. Genotypic data

Genotypic data are supposed to be huge if hundreds of thousands SNPs are stored. In general, PLINK is a very common and powerful tool to compress, convert, and analyze these big data. This package not only employs PLINK (Purcell, Shaun, et al. 2007) to pack genotype data, but also allows a user-defined, simple format to store some small genotype data which are produced by small experiments or outputted by other tools.

4.2.3. PLINK format

Given the advantages of storage space and loading time, the binary PLINK files are used by default in this package. If the binary data file is not readily available, the following command can convert the common PED and MAP paired files into binary group files, which include one binary file (*.bed) and two plain text files (*.bim and *.fam) that can be viewed with a standard text editor.

```
$ plink --file mydata --out mydata --make-bed
```

1) *bed* file

The *bed* file is a compressed binary file containing genotype information. If you try to view it, you will only see lots of strange characters on the screen.

2) *bim* file

The *bim* file is an extended MAP file where each line describes a single individual, and it must contain exactly 4 columns: chromosome, SNP identifier, genetic distance and base-pair position (bp units). The following two extra columns are allele names.

```
0    ss66369915    0    0    G    A
0    ss66112992    0    0    G    A
…
```

3) *fam* file

Phenotypic information are stored in the *fam* file, where one row represents one subject and the first six columns are mandatory: Family ID, Individual ID, Paternal ID, Maternal ID, Sex (1=male; 2=female; other=unknown) and phenotype. However, the phenotype defined here is ***not*** used in this package, so a separate phenotype data will be supplied.

```
957 2274 13631 2615 2 30.6367470222222
137 2349 0 0 2 34.4484237154545
…
```

4.2.4 Simple format

In addition to the PLINK format, a user-defined format named simple format is designed to store small amount of SNPs for users who do not use PLINK. The genotypic data are stored in the CSV format, where each line describes a single SNP and must starts with 5 columns of chromosome information (SNP name, chromosome number, position, reference allele, alternate allele). Three genotypes (aa=0, Aa=1, AA=2) and missing data (coded as -1 or NA) are valid SNP values.

```
SNP.Name,Chr,Pos,Ref.Allele, Atl.Allele, Sub1,Sub2,Sub3,Sub4,Sub5, …
SNP1,1,1, A,T, 2, 0, 1, 1, 1, …
SNP2,1,2, C,G, 2, 1, 1, 0, 0, …
…
```

5. Work flow

5.1. Preparing data

The *fGWAS* package provides a computation framework for the GWA study. However, it does not provide any quality control functions, so the ***quality control*** is supposedly finished according to the experiment requirement. Please note that imputation for missing SNPs is not required in the *fGWAS* algorithm.

5.2. Loading phenotype

The data analysis begins with the data loading. The function *fg.load.phenotype* is designed to load phenotype data, including covariate file, phenotype file, and measured time file. For example:

```
# Notice: the following data isn't provided in the package. Please use simulation to try each function.
file.phe.cov <- "./fhs-bmi-phe.csv"
file.phe.long <- "./fhs-bmi-cov.csv"
file.phe.time <- "./fhs-bmi-time.csv"
# Loading phenotype data with the specific curve type and covariance type
obj.phe <- fg.load.phenotype( file.phe.long, file.phe.cov, file.phe.time,
      curve.type="Legendre2", covariance.type="AR1" );
## or without the specific curve type and covariance type, which intend to use curve fitting
obj.phe2 <- fg.load.phenotype( file.phe.long, file.phe.cov, file.phe.time,
      file.plot.pdf="curve.fitting.pdf");
```

In the loading function, the user can specify the biological curve type and covariance structure type. If the user is not sure which curve or covariance structure is fitting for the data, unspecified parameters will launch the curve fitting process, which compare the AIC, BIC, or R^2 value for each curve type to find goodness of curve fitting. In the curve fitting process, the package outputs the AIC, BIC, R^2 values as well as the figure for each curve type. Please note that R^2 will be greater than 0 in theoretic if the fitted curve is better than the mean vector of all individuals, however in some cases, the R^2 values only approach 0 if the mean vector is a very coarse line. In some extreme cases, failed curve fitting will lead to the R^2 values deviate from normal range (0, 1) too far, the most likely reason is bad initial parameter values obtained from the real data. For these exceptional R^2, the user should check the PDF figure

generated by the parameter *file.plot.pdf* in the function *fg.load.phenotype.*

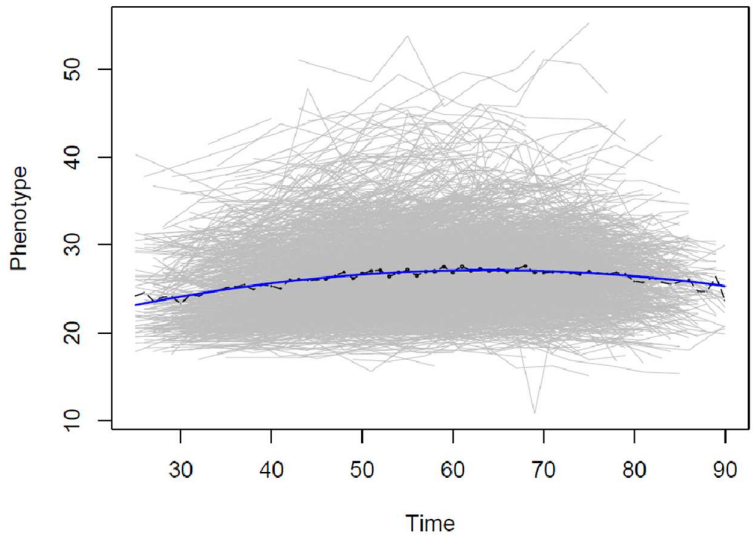
The package uses the user-defined CSV files to store the longitudinal traits rather than phenotypic item in PLINK file because PLINK file cannot keep the longitudinal data. Please note it and make sure the CVS file has correct format to store longitudinal traits.

The loading function returns a phenotype object (S3 object) used in the subsequent process, which includes the curve object, covariance structure, longitudinal traits, covariates, curve fitting results and so on. The following codes show the curve fitting results using the functions in *fGWAS*.

```
obj.phe <- fg.load.phenotype( file.phe.long, file.phe.cov, file.phe.time,
      curve.type=NULL, covariance.type=NULL, file.plot.pdf="curve.fitting.pdf" );
# show the curve fitting results
show(obj.phe$summary.curve$summary)
# show the covariance selection results
show(obj.phe$summary.covariance$summary)
# show summary information of phenotype object
obj.phe;
# plot all phenotype traits and fitted curve in one PDF figure
plot(obj.phe, file.pdf="plot.curve.fitting.pdf");
```

Here we list the summary information of phenotype object and the PDF figure.

```
> obj.phe
== Phenotype Object in fGWAS ==
  Longitudal value : fhs-bmi-phe.csv
   -- Individual count : 1678
  Longitudal time : fhs-bmi-time.csv
   -- Time count : 8
  Covariate file : fhs-bmi-cov.csv
   -- Covariate count : 6
   -- Intercept : YES
   --  Estimate  values  :  55.74923  -1.553526  -2.702604  7.852765  -5.915346
7.209973 3.805997
  Curve type : auto
   -- Estimate type : Legendre2
   -- Estimate values : -27.24468 1.081318 -1.852532
   -- R2  : 0.03025292
  Covariate type : auto
   -- Estimated type : TOEPH
   -- Estimate values : 0.9082581 0.8627943 0.8143804 …
```



Phenotype traits and curve fitting

Although we do not wrap the above data into the *fGWAS* package, we provide the simulation to demonstrate all functions. As for the details of simulation, please check section 6. The following codes show how to simulate and load phenotype data.

```
obj.sim <- fg.simulate( "Logistic", "SAD1", 2000, 800, 1:6,
   phe.missing=0.05, snp.missing=0.05,
   sig.pos=301, plink.format=TRUE, file.prefix = "temp.fwgas.test1" );
# load the phenotype traits generated by the simulation
obj.phe <- fg.load.phenotype("temp.fwgas.test1.pheY.csv", NULL, "temp.fwgas.test1.pheT.csv",
   curve.type = "Logistic", "SAD1", file.plot.pdf = NULL,
   intercept = FALSE,  options = list(verbose=TRUE))
# show the brief information
obj.phe;
```

5.3. Loading genotype

PLINK data file is a very widely used application for analyzing genotypic data. The *fGWAS* package uses this format as the main genotype input. The DNA sequencing raw data, such as Illumina or SOLiD, can be converted to PLINK format with the aid of bio-information tools. In addition to genotype raw data, VCF file developed by 1000 genome projects can store SNPs, indels, and larger structural variants. Although the VCF format cannot be loaded directly in the *fGWAS* package, it is feasible to do the analysis after the conversion to PLINK format using the PLINK1.9 or ***vcftools*** package.

The *fGWAS* package loads PLINK data through the package *snpStats*(Clayton and Leung 2007), which stores genotype matrices in memory. This feature makes the computational node intensive in memory usage and data loading for large-scale genotype data. The *fGWAS* package exerts the PLINK 1.9 to split the big data file and extracts appropriate data during the genotype data access. Therefore, we strongly suggest using the PLINK 1.9 to work with the *snpStats* package when calling the genome load function as follows:

```
file.plink.bed = ./bmi-c1c2-qc2.bed"
```

```
file.plink.bim = "./bmi-c1c2-qc2.bim"
file.plink.fam = "./bmi-c1c2-qc2.fam"
obj.gen <- fg.load.plink( file.plink.bed, file.plink.bim, file.plink.fam, plink.command="your.plink.1.9");
obj.gen
```

In the function *fg.load.plink*, we prefer to use the binary PLINK data file (BED/BIM/MAP) rather than PED/MAP paired file. Please note that PLINK 1.07 is not a good option due to its slow performance. If the PLINK command is not specified, all genotype data in the PLINK data file will be loaded into memory.

The *fg.load.plink* function returns a genotype object (S3 object) containing a genotype reader and some basic information of the genotype data. This genotype object is used in the subsequent steps together with the phenotype object.

The function *fg.load.simple* is an alternative method to load genotype data encoded in the simple format described in 4.2.2. This function is intended to load small genotype data or the SNPP calling results exported by the other tools. The following codes show how to simulate the data and load genotype data encoded in user-define simple format.

```
# simulate the phenotype and genotype data and generate a simple SNP data file
objx <- fg.simulate( "Logistic", "SAD1", 2000, 800, 1:6, phe.missing=0.05, snp.missing=0.05,
    sig.pos=301, plink.format=FALSE, file.prefix = "test1.fgwas");
# load the SNP data generated from the simulation
obj.gen <- fg.load.simple("test1.fgwas.geno.tab");
# show the brief information
obj.gen;
```

The brief information in the genotype object is showed as follows

```
> obj.gen
== Genotype Object in fGWAS ==
  Plink bed :
  Plink bim :
  Plink fam :
  Data file : test1.fgwas.geno.tab
  SNP count : 800
  Total individuals : 2000
  Reference matrix object of class "fg.dm.simple"
  Data type:
  Description:
  SNP Count:  800
  Individual Count:  2000
  Individual Used:  2000
```

5.4 Starting SNP scanning

The function *fg.snpscan* performs the SNP scanning to estimate the LR value and parameters of covariates, curve, and covariance under the hypotheses. In addition to the phenotype object and genotype object, there are some parameters to control the scanning process, such as the computational method, SNP subset, overrode curve type and covariance structure type and other optional values.

The SNP scanning estimates the genetic effects (or genetic curve) for each SNP in the MLE process. It is a time-consuming process to do with the large-scale genotype, even in the parallel computing environment. The *fGWAS* package provides three methods to implement MLE, including *'fast'*, *'fgwas',* and *'optim-fgwas'*. The original *'fgwas'* method is slow and accurate for all SNPs; however, the 'fast' method is faster and less accurate for the insignificant SNPs. The combined model *'optim-fgwas'* employs *'fast'* method on all SNPs firstly and then adds the extra *'fgwas'* estimation for the significant SNPs based on the results from the first run, so it can keep its fast speed and accuracy on significant SNPs.

Although the curve type and covariate structure type can be assigned in the phenotype loading

function, these two types also can be changed before the SNP scanning. This allows the user to select the different curve type or covariance structure to compare the results.

Generally, all SNPs in genotype data file are involved in the SNP scanning. However, setting the parameter *snp.sub* can change this default rule. The parameter *snp.sub* accepts the index or name of SNPs to choose the subset of SNP scanning. For example:

```
r.fgwas<-fg.snpscan(obj.gen, obj.phe, method="fast", snp.sub=c(1000:3000), covariance.type="AR1", options=list( ncores=3, max.loop=5, verbose=TRUE))
```

Three types of control parameters in the function *fg.snpscan* allow the users to supply the following information.

1) The SNPs data is usually so huge that it may spend plenty of processing time in the program. To reduce computing time, it is desirable to adopt parallel computing. The parameter *ncores* controls the number of CPUs used to compute.
2) The parameter *max.loop* controls the replicate number of optimization process. The big number increases the possibility of finding global minimal likelihood value, but also increases the computational time.
3) The parameter *verbose* indicates whether the detailed information is outputted on the console.

The whole computation task may or may not involve curve fitting. The following example shows how to do a SNP scanning using the function *fg.snpscan*.

```
> r.fgwas <- fg.snpscan(obj.gen, obj.phe, method="fast", snp.sub=c(1000:3000),
covariance.type="AR1", options=list(ncores=3)));
[ BLASSO PLINK ] Procedure.
Checking the parameters ......
* Phenotypic Data File =  /work/bmi-pheno-age-mean.csv
* PLINK BED File =  /work/FHS-bmi-v1-chr2.bed
* PLINK BIM File =  /work/FHS-bmi-v1-chr2.bim
* PLINK FAM File =  /work/FHS-bmi-v1-chr2.fam
* Response Variable = Y
* Covariate Columns =
* fGWAS Filter Used = Yes
```

This function returns a result object with the class name *'fgwas.scan.obj'*, which can be operated to get further information described in the followings section, e.g., extracting significant SNPs, drawing the curve for each genotype, and drawing the Manhattan figure.

5.5 Extracting significant SNPs.

The result object contains the scanning results, such as SNP information, genetic curve parameters, covariate parameters, and covariance parameters under the null hypothesis and the alternative hypothesis. The *print* or *show* command is a convenient way to check the significant SNPs in *fGWAS*, which can sort the p-values of SNP and show the top 20 SNPs in the R console. For examples:

```
> r.fgwas;
== Result from ' fGWAS ' method ==
 SNP =  431670
            INDEX       NAME CHR       POS         MAF NMISS           pv
rs7876326  424440  rs7876326  23  23626268 0.470481928    18 3.074300e-13
rs5970594  424442  rs5970594  23  23626473 0.471419976    16 3.441645e-13
rs4452953  424444  rs4452953  23  23636793 0.471522782    10 4.528907e-13
rs5905239  428900  rs5905239  23 115429097 0.328228228    13 5.553860e-13
rs7063064  428901  rs7063064  23 115441019 0.329224293    15 8.610324e-13
rs4824343  428898  rs4824343  23 115427053 0.330815710    23 1.098705e-12
rs4824344  428899  rs4824344  23 115427277 0.328820698    16 1.287038e-12
……
Check all SNPs using this variable: 'your_object$ret.fgwas$result'.
```

The result object not only contains the scanning results, but also the dependent variables, such as phenotype data, curve object, and covariance object. In order to get the data frame of the

scanning result, you need to use the summary command to obtain the data frame result for all SNPs. For example:

```
>df.fgwas <-summary(r.fgwas);
>head(df.fgwas);
```

Here we demonstrate the output of above command in the test example.

```
# the following two commands have same function.
> head(r.fgwas$ret.fgwas$result)
> head(df.fgwas)

INDEX   NAME CHR POS         MAF NMISS SNP0 SNP1 SNP2 GENO           LR        pv
1 ss66037954   0   0 0.036680698    15 1547  110    6    3 4.1396553769 0.6577837
2 ss66040774   0   0 0.094407697    15 1360  292   11    3 0.5618262791 0.9970028
3 ss66043486   0   0 0.020870602     1 1610   64    3    3 1.1576737634 0.9789237
4 ss66047046   0   0 0.146326655    31 1173  466    8    3 0.7249783654 0.9939361
5 ss66048035   0   0 0.002386635     2 1668    8    0    2 0.0002225726 0.9999991
6 ss66057959   0   0 0.060901340    36 1442  200    0    2 3.7476660303 0.2900324
……
```

For the significant SNPs in the result object, the function *fg.select.sigsnp* can extract the information of the significant SNPs, including SNP indexes, SNP names, p-values and parameters. The SNP index or name is a key to call the curve drawing.

```
r.sigsnps <- fg.select.sigsnp (r.fgwas);
head(r.sigsnps);
```

Here we demonstrate the output of above command in the test example.

```
> head(r.sigsnps);
INDEX      NAME    CHR       POS       MAF NMISS SNP0 SNP1 SNP2 GENO       LR           pv
424440 rs7876326  23  23626268 0.4704819    18  640  478  542    3 70.60376 3.074300e-13
424442 rs5970594  23  23626473 0.4714200    16  640  477  545    3 70.36484 3.441645e-13
424444 rs4452953  23  23636793 0.4715228    10  639  485  544    3 69.78354 4.528907e-13
428900 rs5905239  23 115429097 0.3282282    13  909  419  337    3 69.35138 5.553860e-13
428901 rs7063064  23 115441019 0.3292243    15  905  421  337    3 68.42204 8.610324e-13
428898 rs4824343  23 115427053 0.3308157    23  898  419  338    3 67.90508 1.098705e-12
```

5.6. Plotting figures

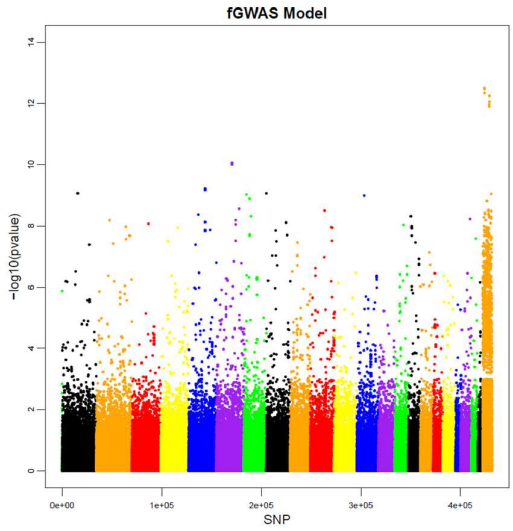
The function *plot* can output two types of PDF figure**,** including:

1) The Manhattan figure applying the *plot* function on the result object.

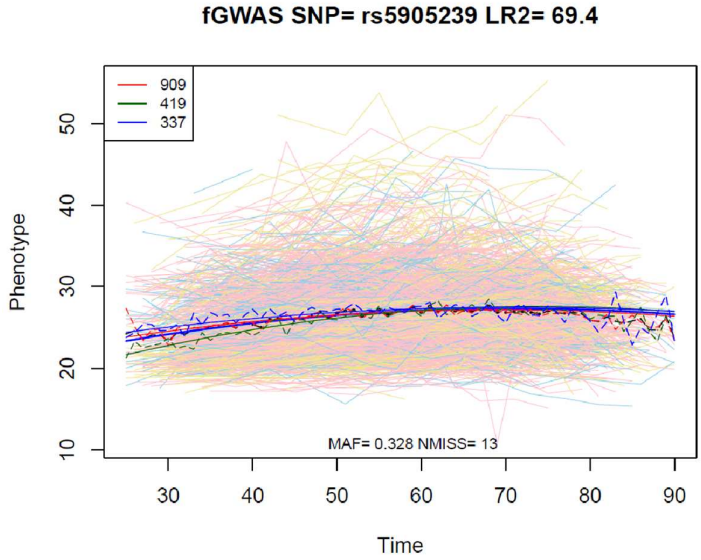
2) The figures of genetic effects for the specified SNPs using the function *plot.fgwas.curve.*

For examples:

```
# draw Manhattan plot for all SNPs
plot (r.fgwas, file.pdf="demo.manhattan.pdf");
# draw genetic effects of each genotype for the significant SNPs
plot.fgwas.curve ( r.fgwas, snp.sub= r.sigsnps$INDEX[1:5], file.pdf="demo.sigeffect.pdf");
```



The Manhattan figure shows –log10(p-values) for each SNP along with X axis. For specified SNPs, usually significant SNPs, the genetic curves are drawn with the original phenotypic traits as the background. The example below shows how to draw the genetic curves for significant SNPs in one PDF file.



6 Simulation

The simulation function (*fg.simulate*) is provided for users who would like to try this package without any real data. It is a good way to understand the functions and learn how to use them. The simulation in this package uses the pre-defined parameters to create a data object. This data object generated by the simulation has the same structure as real data. All pre-defined parameters can be customized. The following table shows the simulation parameters in the function *fg.simulate*.

Table1： Simulation Parameters in *fGWAS*

Items	**Description**
curve.type	String, the curve type, such as "Logistic", "Legendre2"

covariance.type	String, the covariance structure type, such as "SAD1", "AR1"
n.obs	Integer, the size of samples
n.snp	Integer, the number of SNPs
time.points	Numeric, the measure time points for the longitudinal phenotype
par0	Numeric vector, indicating the curve parameters for the genotype AA
par1	Numeric vector, indicating the curve parameters for the genotype Aa
par2	Numeric vector, indicating the curve parameters for the genotype aa
par2	Numeric vector, indicating the parameters of the covariance matrix
par.X	Numeric vector, indicating the covariate coefficients.
phe.missing	Numeric, the missing rates of phenotype data
snp.missing	Numeric, the missing rates of genotype data.
sig.pos	Numeric, indicating the significant SNP position.
plink.format	Logical variable, indicating whether the PLINK data files are generated
file.prefix	String, the prefix file name for the simulation data.

The *fg.simulate* function returns a list containing the phenotype object and the genotype object. The following codes show how to create simulation data and how to change some parameters by R command.

```
# simple simulate
obj.simu<- fg.simulate()
#simulate using the customized parameters
obj.simu<- fg.simulate( "Logistic", "SAD1", 2000, 800, 1:6,
   phe.missing=0.05, snp.missing=0.05, sig.pos=301,
   plink.format=TRUE, file.prefix = "temp.fgwas" );
#show the genotype object
obj.simu$obj.gen;
#show the phenotype object
obj.simu$obj.phe;
#SNP scanning
r.fgwas <- fg.snpscan(obj.simu$obj.gen, obj.simu$obj.phe);
```

The simulation data can be exported into files in two formats, PLINK or simple format. The first command generates simulation data by default parameter values, and the second one creates data using some customized parameters.

7 Reference

1. Luo, J., Berg, A., Ahn, K., Das, K., Li, J., Wang, Z., ... & Wu, R. (2010). Functional Genome-Wide Association Studies of Longitudinal Traits. In Handbook of Adaptive Designs in Pharmaceutical and Clinical Development (pp. 23-1). CRC Press..

2. Das, K., Li, J., Wang, Z., Tong, C., Fu, G., Li, Y., ... & Wu, R. (2011). A dynamic model for genome-wide association studies. Human genetics, 129(6), 629-639.

3. Clayton, D., & Leung, H. T. (2007). An R package for analysis of whole-genome association studies. Human heredity, 64(1), 45-51.

4. Purcell, S., Neale, B., Todd-Brown, K., Thomas, L., Ferreira, M. A., Bender, D., ... & Sham, P. C. (2007). PLINK: a tool set for whole-genome association and population-based linkage analyses. The American Journal of Human Genetics, 81(3), 559-575.