JASA A&CS Reproducibility Initiative - Author Contributions Checklist Form

The purpose of the Author Contributions Checklist (ACC) Form is to document the code and data supporting a manuscript, and describe how to reproduce its main results.

As of Sept. 1, 2016, the ACC Form must be included with all new submissions to JASA A&CS.

This document is the initial version of the template that will be provided to authors. The JASA Associate Editors for Reproducibility will update this document with more detailed instructions and information about best practices for many of the listed requirements over time.

Data

Abstract (Mandatory)

In the paper, real exome chip data were analyzed from UCLA/Pittsburgh family-based study of age-related macular degeneration. An individual was considered affected with AMD according to the "C" diagnostic scheme previously defined; an individual was considered unaffected if they were unaffected and at least 65 years old at last exam. After sample quality checks using a thorough and rigorous data cleaning pipeline, which included checks for chromosomal aberrations, gender, Hardy-Weinberg equilibrium, relatedness, duplicates, and genotype quality, 976 genotyped individuals of European ancestry were available for analysis. To completely connect pedigrees, we included non-genotyped individuals who shared the same family with those 976 genotyped individuals. The connected pedigrees contained 2,727 pedigree members, 1,275 with a known AMD trait (1,031 affected and 244 unaffected). A total number of 111,547 autosomal variants were included in the study; 30,096 were common, and 81,451 were rare. In the analysis, we adjusted for gender since it is significantly associated with AMD in the null model (p-value = 0.00197).

Availability (Mandatory)

The subset of the age-related macular degeneration data that is permitted for sharing by the University of Pittsburgh institutional review board forms part of the larger "International Age-Related Macular Degeneration Genomics Consortium" data set that is available from the database of Genotypes and Phenotypes (dbGaP) under accession phs001039.v1.p1.

Restrictions (if data will not be made publicly available, justify why not).

A portion of the age-related macular degeneration data analyzed here was not consented for sharing (because it was gathered years before these data sharing initiatives began) and so cannot be made publicly available.

Description (Mandatory if data available)

Permissions (demonstrate that author has legitimate access to data)

These data are ours, as they were originally collected and generated by co-authors Drs. Gorin and Conley in a long-term collaboration with Dr. Weeks.

Licensing information
Link to data (e.g., dataverse.org, datadryad.org; this need not be the actual link at time of submission but if not, it should indicate where the data will be deposited if the manuscript is accepted)
https://www.ncbi.nlm.nih.gov/projects/gap/cgi-bin/study.cgi?study_id=phs001039.v1.p1

Data provenance, including identifier or link to original data if different than above
File format
Metadata (including data dictionary)
Version information

Optional Information (complete as necessary)
Unique identifier / DOI

Code

Abstract (Mandatory)
Short high level description
We have implemented our new statistics in R and made them available in the PedGFLMM R package, along with detailed function-level documentation as well as a vignette. We have also made our simulation code, which is in R, available in a GitHub repository.

Description (Mandatory)
How delivered (R package, Shiny app, etc.)
The PedGFLMM R package is available from https://github.com/DanielEWeeks/PedGFLMM
Its vignette is available at
https://github.com/DanielEWeeks/PedGFLMM/blob/master/PedGFLMM_vignette.pdf

The simulation R code we used is available from the GitHub repository:
https://github.com/DanielEWeeks/PedGFLMM-simulation-code
The documentation is available at that page too, visible by scrolling down (It is also appended in the Appendix of this document below).

Licensing information (default is MIT License)
GNU General Public License, version 2

Link to code/repository (e.g., github.com, bitbucket.org; this need not be the actual link at time of submission but if not, it should indicate where the code will be deposited if the manuscript is accepted)
PedGFLMM R package: https://github.com/DanielEWeeks/PedGFLMM
PedGFLMM simulation code: https://github.com/DanielEWeeks/PedGFLMM-simulation-code

Version information (e.g., for a Git repository, the number or branch+commit)
The PedGFLMM R package: version: 1.0.0; Branch: Master.
The PedGFLMM simulation code: Tag: v1.0.0; Branch: Master

On our computational cluster where the most time-consuming simulations and analyses were done, we used gcc version 4.8.5 and R version 3.5.2 with the following packages and versions:

```
R version 3.5.2

attached base packages:
 [1] grid      parallel  splines   stats     graphics  grDevices utils
 [8] datasets  methods   base

other attached packages:
 [1] forcats_0.4.0      stringr_1.4.0      dplyr_0.8.1        purrr_0.3.2
 [5] readr_1.3.1        tidyr_0.8.3        tibble_2.1.3
ggplot2_3.1.1
 [9] tidyverse_1.2.1    pedgene_3.2        kinship2_1.8.4
quadprog_1.5-7
[13] survey_3.36        survival_2.44-1.1  CompQuadForm_1.4.3 glmm_1.3.0
[17] doParallel_1.0.14  iterators_1.0.10   foreach_1.4.4      mvtnorm_1.0-
10
[21] trust_0.1-7        pedigreemm_0.3-3   lme4_1.1-21        nlme_3.1-140
[25] MASS_7.3-51.4      fda_2.4.8          Matrix_1.2-17

loaded via a namespace (and not attached):
 [1] Rcpp_1.0.1       lubridate_1.7.4  lattice_0.20-38  assertthat_0.2.1
 [5] R6_2.4.0         cellranger_1.1.0 plyr_1.8.4       backports_1.1.4
 [9] httr_1.4.0       pillar_1.4.1     itertools_0.1-3  rlang_0.3.4
[13] lazyeval_0.2.2   readxl_1.3.1     rstudioapi_0.10  minqa_1.2.4
[17] nloptr_1.2.1     munsell_0.5.0    broom_0.5.2      compiler_3.5.2
[21] modelr_0.1.4     pkgconfig_2.0.2  mitools_2.4      tidyselect_0.2.5
[25] codetools_0.2-16 withr_2.1.2      crayon_1.3.4     jsonlite_1.6
[29] gtable_0.3.0     DBI_1.0.0        magrittr_1.5     scales_1.0.0
[33] cli_1.1.0        stringi_1.4.3    xml2_1.2.0       generics_0.0.2
[37] boot_1.3-22      tools_3.5.2      glue_1.3.1       hms_0.4.2
[41] colorspace_1.4-1 rvest_0.3.4      haven_2.1.0
```

Optional Information (complete as necessary)
Hardware requirements (e.g., operating system with version number, access to cluster, GPUs, etc.)
Supporting software requirements (e.g., libraries and dependencies, including version numbers)
Unique identifier/DOI

Instructions for Use

Reproducibility (Mandatory)
What is to be reproduced (e.g., "All tables and figure from paper", "Tables 1, 3", etc.)

The tables and figures from the paper that are based on the simulated data. The results of the analyses of the real data cannot be exactly reproduced by others because not all of the underlying data were consented for public sharing.

How to reproduce analyses (e.g., workflow information, makefile, wrapper scripts)
A detailed description of how to reproduce the analyses can be found in the "ReadME.docx" in the GitHub repository for the PedGFLMM simulation code:
https://github.com/DanielEWeeks/PedGFLMM-simulation-code

(The content of the "ReadME.docx" is appended below).

Replication (Optional)
How to use software in other settings (or links to such information, e.g., R package vignettes, demos or other examples)
Please see our R package vignette at
https://github.com/DanielEWeeks/PedGFLMM/blob/master/PedGFLMM_vignette.pdf

Notes

Other relevant information, in particular how to access the data and code if not yet made publicly available.

Appendix

The contents of the "ReadME.docx" from the GitHub repository for the PedGFLMM simulation code: https://github.com/DanielEWeeks/PedGFLMM-simulation-code

Simulation code for Jiang et al (2020)

The PedGFLMM R package
If you would like to apply our statistics yourself, please use the PedGFLMM R package, which is available at https://github.com/DanielEWeeks/PedGFLMM
Note also that within the PedGFLMM R package, we provide an illustrative example of how to apply our statistical functions to real data as reformatted and imported via our Mega2 C program and Mega2R R package (https://cran.r-project.org/package=Mega2R). We suggest that would be an easier way for the interested reader to apply our statistics to their own data in an automated manner, across genes or user-defined regions.

Simulation code

Here we present example simulation code that was used to generate results in our paper:

Jiang YD, Chiu CY, Yan Q, Chen W, Gorin MB, Conley YP, Lakhal-Chaieb ML, Cook RJ, Amos CI, Wilson AF, Bailey-Wilson JE, McMahon FJ, Vazquez AI, Yuan A, Zhong XG, Xiong MM, Weeks DE, and Fan RZ (2020) Gene-based association testing of dichotomous traits with generalized linear mixed models for family data.

In case of suggestions and questions and/or problems, please contact us via e-mail (rf740@georgetown.edu).

For a mapping of the various functions to the results in the paper, please see Table 1 (below on page 14).

In this repository, we have this directory structure:

```
.
├── PedGFLMM
│   └── data
├── PedGFLMM_non_dynamic
│   └── data
├── analysis_code
└── simulations
    ├── COSI_50_Ped
    │   ├── Type_I_Error_Rare
    │   ├── Type_I_Error_Rare_and_Common
    │   ├── power_codes
    │   │   └── test_C=1_Delta=0.5
    │   └── power_test_C=1_Delta=0.50
    │       ├── Fig_Pow
    │       ├── Fig_Pow_orig
    │       ├── Power_causal_rare
    │       ├── Power_causal_rare_and_common
    ├── Input
    ├── SKAT
    │   └── Simulation_Code_Example
    └── Type_I_error_rate_aggregate
```

There are four main sub-directories:

1. The 'PedGFLMM' directory

```
PedGFLMM
├── MGAO.R
├── PedGFLMM_EXAMPLE.R
├── PedGFLMM_beta_smooth_only.R
├── PedGFLMM_doc.pdf
├── PedGFLMM_fixed_model.R
├── PedGLMM_additive_effect_model.R
├── data
```

```
│   ├── Ped.csv
│   ├── covariate.csv
│   ├── geno.csv
│   └── snpPos.csv
└── dynamic_rule.R
```

The 'PedGFLMM' directory contains the version of the PedGFLMM codes that were used in the simulations. These were reformatted and improved to form our well-documented 'PedGFLMM' R package, which is available at https://github.com/DanielEWeeks/PedGFLMM. The less-well-documented version of our code in the 'PedGFLMM' directory is provided here because these are the codes that were used by our simulation codes provided in the 'simulations' directory.

We encourage users to use the polished well-documented 'PedGFLMM' R package instead of the rougher code presented here.

In addition to providing the codes, in the 'data' subfolder, we provide an example toy data set in the format expected by our functions. The PedGFLMM R vignette clearly describes the expected format of these input data.

2. The 'simulations' directory

The 'simulations' directory contains the simulation codes, as well as some results of the simulations, along with the code used to generate the corresponding Figures and Table.

Directory structure of the 'simulations' folder:

```
simulations
├── COSI_50_Ped
│   ├── Type_I_Error_Rare
│   ├── Type_I_Error_Rare_and_Common
│   ├── power_codes
│   │   └── test_C=1_Delta=0.5
│   └── power_test_C=1_Delta=0.50
│       ├── Fig_Pow
│       ├── Fig_Pow_orig
│       ├── Power_causal_rare
│       ├── Power_causal_rare_and_common
├── Input
├── SKAT
│   └── Simulation_Code_Example
└── Type_I_error_rate_aggregate
```

3. The 'PedGFLMM_non_dynamic' directory

```
PedGFLMM_non_dynamic/
├── PedGFLMM_EXAMPLE.R
├── PedGFLMM_beta_smooth_only.R
├── PedGFLMM_doc.pdf
├── PedGFLMM_fixed_model.R
├── PedGLMM_additive_effect_model.R
└── data
    ├── Ped.csv
    ├── covariate.csv
    ├── geno.csv
    └── snpPos.csv
```

The 'PedGFLMM_non_dynamic' directory contains functions that were used in the simulations where we did not use the dynamic rule to set the number of basis functions. These functions can easily be used in the simulation framework below by calling these files instead of the ones in the "PedGFLMM" directory. For more details about these specific functions, see the "PedGFLMM_doc.pdf" file in this directory.

4. The 'analysis_code' directory

```
analysis_code/
├── README.md
└── data_analysis.R
└── dynamic_rule.R
```

The 'analysis_code' directory contains a copy of the code we used to analyze the real data. Note that a modified dynamic rule, as defined by the `dynamic_rule.R` file in this folder, was used to analyze the real data.

While our complete real data set is unavailable due to consent issues, we have included our real analysis code in this GitHub repository as a model. However, actually within the PedGFLMM R package, we provide an illustrative example of how to apply our statistical functions to real data as reformatted and imported via our Mega2 and Mega2R R package (https://cran.r-project.org/package=Mega2R), and would suggest that would be an easier way for the interested reader to apply our statistics to their own data. Please see the PedGFLMM R package vignette for further details.

Type I Error and Power Simulations

The code used for the Type I Error and Power Simulations is documented below. For additional details of how the simulation studies were done, please see the "Simulation Studies" section of our Jiang et al. paper (citation above).

For the calculations described below to get power levels and type I error rates, one has to run the R codes on a Unix/Linux machine (or on a Macintosh). Our simulations were originally carried out on the helix/biowulf computational cluster at NIH.

Type I Error Simulations Timing:

For the Type I Error study, to complete all simulations to get the Table 3, it would have taken about 5 days if we were able to start all jobs in parallel on our computer cluster at the same time. Note, we did 400 seeds each with 2,500 replicates to have a total of 3 million replicates for each case. The shortest time was 50 hours for the case of 6 kb Rare case, and the longest was 104 hours for the case of 21 kb Rare and common case. In practice, it took us almost 2 weeks to get all results for Table 3, since we were not able to start all jobs at the same time.

On an Intel Xeon E5-2690 v3 2.6 GHz processor, using the code in the "Type_I_error_Rare_and_Common_region_size_6k.R", each replicate took approximately 101.8 seconds to complete. Similarly based on running "Type_I_error_Rare_region_size_6k.R" for 5 replicates, each replicate took approximately 111.6 seconds to complete.

INSTALLATION

Compilation of C functions in the Util.c file

```
├── simulations
│   ├── Make.sh
│   ├── Util.c
│   ├── Util_Read_Me.docx
```

Our simulation scripts rely on some C functions that are defined in the Util.c file, which is in the 'simulations' folder. Before the simulation scripts below can be run, the Util.c file has to be compiled to generate a 'Util.so' file. This works on Unix and Macintosh OS X machines.
On a Unix machine or in a Terminal window on a Macintosh OS X machine (with a C compiler installed), the 'Util.so' can be generated via these commands, to be issued in the 'simulations' folder:

```
chmod +x Make.sh
./Make.sh
```

R libraries required for the simulations

The simulation scripts can only be run after the following required R packages have been installed:
fda
MASS
Matrix
nlme
glmm
pedgene
pedigreemm
tidyverse

Decompress haplotype file

```
├── simulations
│   ├── SKAT
│   │   └── Simulation_Code_Example
│   │       ├── out.50.hap-1.zip
│   │       └── out.50.pos-1
```

To carry out the simulations below, first you must unzip the
"`out.50.hap-1.zip`"
file in the "`simulations/SKAT/Simulation_Code_Example`" folder.

Reproducing the simulation studies

We carried out simulations under a number of conditions and variations. Here we describe here how to reproduce the Type I and Power simulations using the dynamic rule as applied to the 50 pedigree template.

Reproducing the Type I Error simulations

```
├── simulations
│   ├── COSI_50_Ped
│   │   ├── Type_I_error_Rare_and_Common_region_size_6k.R
│   │   ├── Type_I_error_Rare_region_size_6k.R
│   │   ├── Type_I_error_func.R
```

Type I error calculation: we provide two R scripts, Type_I_error_Rare_region_size_6k.R and Type_I_error_Rare_and_Common_region_size_6k.R, in the **simulations/COSI_50_Ped/** folder; these call the function in the Type_I_error_func.R file. The two scripts should generate two files, one is Type_I_error_region_size_6000_Nsim_2500_seed_1.csv in **simulations/COSI_50_Ped/Type_I_Error_Rare**, and the other is Type_I_error_region_size_6000_Nsim_2500_seed_1.csv in **simulations/COSI_50_Ped/Type_I_Error_Rare_and_Common.**

For more results, one can change the seed number to generate them.

On an Intel Xeon E5-2690 v3 2.6 GHz processor, running Type_I_error_Rare_region_size_6k.R to simulate 501 replicates took 9.9 hours, about 71.2 seconds per replicate (prior to our initial submission; code should be faster now).

How to run *'Type_I_error_Rare_region_size_6k.*R'

This file is in the "`simulations/COSI_50_Ped`" folder.
Open it in RStudio and click on the 'Source' button.
Or start up R and source the file via this R command:
source("Type_I_error_Rare_region_size_6k.R")
The 'Type_I_error_Rare_and_Common_region_size_6k.R' R script can be run in the same manner.

Output of the Type I Error simulation programs:

The 'Type_I_error_Rare_region_size_6k.R' R script writes its output into the 'simulations/COSI_50_Ped/Type_I_Error_Rare' folder.

The 'Type_I_error_Rare_and_Common_region_size_6k.R' R script writes its output into the 'simulations/COSI_50_Ped/Type_I_Error_Rare_and_Common**'** folder.

Reproducing the Power simulations

Power level calculation: we provide twelve R scripts in **simulations/COSI_50_Ped/power_codes/test_C=1_Delta=0.5**, which call the function Power_func.R. Then one should be able to generate all the power levels in **simulations/COSI_50_Ped/power_test_C=1_Delta=0.5**

```
power_codes
   └── test_C=1_Delta=0.5
       ├── Power_GFLMM_rare_and_common_region_size_15k_18k_21k_Neg_beta_pct_00.R
       ├── Power_GFLMM_rare_and_common_region_size_15k_18k_21k_Neg_beta_pct_20.R
       ├── Power_GFLMM_rare_and_common_region_size_15k_18k_21k_Neg_beta_pct_50.R
       ├── Power_GFLMM_rare_and_common_region_size_6k_9k_12k_Neg_beta_pct_00.R
       ├── Power_GFLMM_rare_and_common_region_size_6k_9k_12k_Neg_beta_pct_20.R
       ├── Power_GFLMM_rare_and_common_region_size_6k_9k_12k_Neg_beta_pct_50.R
       ├── Power_GFLMM_rare_region_size_15k_18k_21k_Neg_beta_pct_00.R
       ├── Power_GFLMM_rare_region_size_15k_18k_21k_Neg_beta_pct_20.R
       ├── Power_GFLMM_rare_region_size_15k_18k_21k_Neg_beta_pct_50.R
       ├── Power_GFLMM_rare_region_size_6k_9k_12k_Neg_beta_pct_00.R
       ├── Power_GFLMM_rare_region_size_6k_9k_12k_Neg_beta_pct_20.R
       └── Power_GFLMM_rare_region_size_6k_9k_12k_Neg_beta_pct_50.R
```

Power Simulations Timing:

On an Intel Xeon E5-2690 v3 2.6 GHz processor, running only the first call to the power function "power_func" in the Power_GFLMM_rare_and_common_region_size_15k_18k_21k_Neg_beta_pct_00.R file to simulate 100 replicates took 2.59 hours, about 93.2 seconds per replicate.

For the power comparisons with the revised code, we set the upper limit of the total number of replicates as 3,000 and stop the simulation if all of our proposed methods get 1,000 valid samples (without any error or warnings). Note, in "power_codes", each job files includes nine cases of simulations (e.g., calls to the "power_func" function). When we did the computer experiments, we split each job files into three, which includes three simulations only. This took us almost 5 days to get all results.

How to run the 'Power_GFLMM_rare_and_common_region_size_15k_18k_21k_Neg_beta_pct_00.R' power simulation script:

Open it in RStudio and click on the ‘Source’ button.

Or start up R and source the file via this R command:

source("Power_GFLMM_rare_and_common_region_size_15k_18k_21k_Neg_beta_pct_00.R")

The other R scripts can be run in the same manner.

Output of the Power simulation programs:

Each of the power simulation R scripts writes its output into one of the two subfolders

Power_causal_rare

Power_causal_rare_and_common

in the **simulations/COSI_50_Ped/power_test_C=1_Delta=0.5** folder.

Reproducing the Figures

Here we provide and document the code that we used to generate the bar plot figures based on the 50 pedigree power simulations.

```
├── simulations
│   ├── bar_dynamic_function_plot.R
│   ├── bar_dynamic_result_plot.R
```

Using as input the results of our previous (time-consuming) simulations, one may source *bar_dynamic_result_plot.R* in the **simulations** folder to generate Figures in the **simulations/COSI_50_Ped/power_test_C=1_Delta=0.50/Fig_Pow/** folder, which are the Figures A.1 – A.12 in the main text. If you have not carried out any simulations, then these should match the original set of Figures which are in the **simulations/COSI_50_Ped/power_test_C=1_Delta=0.50/Fig_Pow_orig** folder. If you carry out simulations above, these may change some of the results in the simulation output directories which form the input data for these Figures – in such a case, if you desire to use our original input, just download everything again.

How to run '*bar_dynamic_result_plot.R*':

Open it in RStudio and click on the 'Source' button.

Or start up R and source the file via this R command:

source("bar_dynamic_result_plot.R ")

Input for the '*bar_dynamic_result_plot.R'* Figure-generating program:

Within the 'simulations/COSI_50_Ped/power_test_C=1_Delta=0.50' folder, there are two folders 'Power_causal_rare' and 'Power_causal_rare_and_common' which contain the results of our previous (time-consuming) simulations.

Output of the '*bar_dynamic_result_plot.R'* Figure-generating program:

When this R program is run by being sourced, the resulting Figures are output to the 'simulations/COSI_50_Ped/power_test_C=1_Delta=0.50/Fig_Pow' folder. These should match the original set of Figures which are in the 'simulations/COSI_50_Ped/power_test_C=1_Delta=0.50/Fig_Pow_orig' folder.

Reproducing Table 3 - the empirical type I error rates

To reproduce Table 3, you first need to run the Type_I_error_Rare_region_size_6k.R and Type_I_error_Rare_and_Common_region_size_6k.R scripts described above to properly populate the **simulations/COSI_50_Ped/Type_I_Error_Rare** and **simulations/COSI_50_Ped/Type_I_Error_Rare_and_Common** folders with the desired set of results.

Note that the set of results you obtain may differ a bit from those presented in the paper because these are simulations that depend not only on the exact set of random number seeds used but may also depend on the machine on which it is run.

Then, to calculate your empirical type I error rates in Table 3 for 6 kb region based on your results of those (time-consuming) simulations, source the *50_Ped_result_Type_I_aggregate.R* file in **simulations/Type_I_error_rate_aggregate,** and the results will be generated in the **simulations/COSI_50_Ped/Type_I_Error_combined_Results** folder.

How to run '*50_Ped_result_Type_I_aggregate.R*':
After you have properly populated the the **simulations/COSI_50_Ped/Type_I_Error_Rare** and **simulations/COSI_50_Ped/Type_I_Error_Rare_and_Common** folders with the desired set of results, then you can proceed as follows:

Open it in RStudio and click on the 'Source' button.
Or start up R and source the file via this R command:
source("50_Ped_result_Type_I_aggregate.R'")

Note: When this file is sourced, it will generate these messages, which can be disregarded:
ERROR : cannot open the connection
Warning message:
In file(file, "rt") :
  cannot open file
'COSI_50_Ped/Type_I_Error_Rare_and_Common/Type_I_error_region_size_6000_Nsim_2500_seed_48.csv': No such file or directory

Input for the '50_Ped_result_Type_I_aggregate.R' function:
The input files, containing the results of our previous simulations, are located in the two folders 'Type_I_Error_Rare' and 'Type_I_Error_Rare_and_Common' within the 'simulations/COSI_50_Ped' folder.

Output of the '50_Ped_result_Type_I_aggregate.R' function:
The output will be generated in the two sub-folders 'Type_I_Error_Rare' and 'Type_I_Error_Rare_and_Common' of the 'simulations/COSI_50_Ped/Type_I_Error_combined_Results' folder.

Table 1: Mapping of the functions to the results in the paper

Script file name	Function	How it relates to the manuscript
MGAO.R	Perform the principal component analysis discussed by Gao et al., 2008	Evaluate the dimension of genotype data to determine the number of basis functions
PedGFLMM_EXAMPLE.R	Provide working examples to build the GFLMM	Give illustrative examples
PedGFLMM_beta_smooth_only.R	Estimate the GFLMM statistics under the beta-smooth only model	Implement Equation (2), revised by Equation (6)
PedGFLMM_fixed_model.R	Estimate the GFLMM statistics under the fixed model	Implement Equation (1), revised by Equation (5)
PedGLMM_additive_effect_model	Estimate the GFLMM statistics under the additive model	Implement Equation (3)
dynamic_rule.R	Apply the dynamic rule to determine the functional data analysis parameters	Implement the method discussed in Functional Data Analysis Parameters and Dynamic Rule
Type_I_error_Rare_and_Common_region_size_6k.R	Simulate type I error with a mix of rare and common genetic variants within a 6 kb region	Generate type I error rates in Table 3
Type_I_error_Rare_region_size_6k.R	Simulate type I error with rare genetic variants within a 6 kb region	Generate type I error rates in Table 3
Power_GFLMM_rare_and_common_region_size_6k_9k_12k_Neg_beta_pct_00.R	Simulate power with a mix of rare and common genetic variants with 0% negative-effect causal variants within a 6 kb, 9 kb, or 12 kb region	Generate empirical power in Figure A.1-A.3
Power_GFLMM_rare_and_common_region_size_6k_9k_12k_Neg_beta_pct_20.R	Simulate power with a mix of rare and common genetic variants with 20% negative-effect causal variants within a 6 kb, 9 kb, or 12 kb region	Generate empirical power in Figure A.1-A.3
Power_GFLMM_rare_and_common_region_size_6k_9k_12k_Neg_beta_pct_50.R	Simulate power with a mix of rare and common genetic variants with 50% negative-effect causal variants within a 6 kb, 9 kb, or 12 kb region	Generate empirical power in Figure A.1-A.3
Power_GFLMM_rare_and_common_region_size_15k_18k_21k_Neg_beta_pct_00.R	Simulate power with a mix of rare and common genetic variants with 0% negative-effect causal variants within a 15 kb, 18 kb, or 21 kb region	Generate empirical power in Figure A.4-A.6
Power_GFLMM_rare_and_common_region_size_15k_18k_21k_Neg_beta_pct_20.R	Simulate power with a mix of rare and common genetic variants with 20% negative-effect causal variants within a 15 kb, 18 kb, or 21 kb region	Generate empirical power in Figure A.4-A.6
Power_GFLMM_rare_and_common_region_size_15k_18k_21k_Neg_beta_pct_50.R	Simulate power with a mix of rare and common genetic variants with 50% negative-effect causal variants within a 15 kb, 18 kb, or 21 kb region	Generate empirical power in Figure A.4-A.6
Power_GFLMM_rare_region_size_6k_9k_12k_Neg_beta_pct_00.R	Simulate power with rare genetic variants with 0% negative-effect causal variants within a 6 kb, 9 kb, or 12 kb region	Generate empirical power in Figure A.7-A.9
Power_GFLMM_rare_region_size_6k_9k_12k_Neg_beta_pct_20.R	Simulate power with rare genetic variants with 20% negative-effect causal variants within a 6 kb, 9 kb, or 12 kb region	Generate empirical power in Figure A.7-A.9

Power_GFLMM_rare_region_size_6k_9k_12k_Neg_beta_pct_50.R	Simulate power with rare genetic variants with 50% negative-effect causal variants within a 6 kb, 9 kb, or 12 kb region	Generate empirical power in Figure A.7-A.9
Power_GFLMM_rare_region_size_15k_18k_21k_Neg_beta_pct_00.R	Simulate power with rare genetic variants with 0% negative-effect causal variants within a 15 kb, 18 kb, or 21 kb region	Generate empirical power in Figure A.10-A.12
Power_GFLMM_rare_region_size_15k_18k_21k_Neg_beta_pct_20.R	Simulate power with rare genetic variants with 20% negative-effect causal variants within a 15 kb, 18 kb, or 21 kb region	Generate empirical power in Figure A.10-A.12
Power_GFLMM_rare_region_size_15k_18k_21k_Neg_beta_pct_50.R	Simulate power with rare genetic variants with 50% negative-effect causal variants within a 15 kb, 18 kb, or 21 kb region	Generate empirical power in Figure A.10-A.12
bar_dynamic_function_plot.R	Create the bar charts with 5%, 10%, and 15% causal genetic variants	Plot the bar charts illustrated by Figure A.1-A.12
bar_dynamic_result_plot.R	Provide existing results to generate the bar charts	Provide data source as an illustrative example to create the bar charts illustrated by Figure A.1-A.12
data_analysis.R	Apply the GFLMM to the real age-related macular degeneration data	Generate the test statistics in Table 1