RarePedSim Documentation [Version 1.0rc]

Biao Li, Gao T. Wang and Suzanne M. Leal

Last updated: September 2, 2015

Contents

Chapter 1

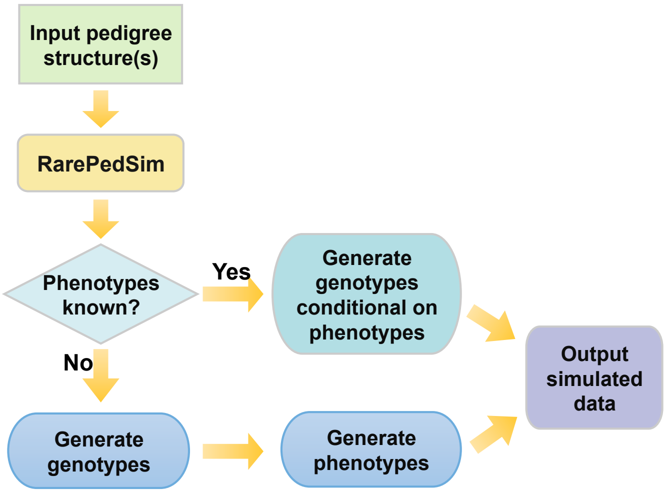
Getting Started with RarePedSim

1.1 Introduction

`RarePedSim`, a unified simulation program for rare variant sequence-based pedigree data analysis, provides an efficient and effective manner to generate gene/region-level pedigree segregating data conditional and unconditional on phenotypes.

- **Variant data information** – site-specific minor allele frequencies, positions and functionalities can be 1) simulated by RarePedSim using forward-time simulation with state-of-the-art demographic, multi-locus fitness effect and purifying selection models, or 2) extrapolated from exome or whole genome sequence data such as NHLBI-Exome Sequencing Project or 1000 Genomes Project.
- **Pedigree structure(s)** - allows for any arbitrarily complex pedigrees including missing data and consanguinity.
- **Mendelian trait model** - allows for varying mode of inheritance, (reduced) penetrance, phenocopy effect, allelic heterogeneity and locus heterogeneity.
- **Complex trait model** - logistic odds ratio (LOGIT) model for qualitative traits; linear mean-shift (LNR) model for quantitative traits.

See figure below for the data simulation workflow using `RarePedSim`



1.2 Citing RarePedSim

If you use `RarePedSim` in any published work, please cite both the software (as an electronic resource/URL) and the manuscript that describes the methods.

- Package: RarePedSim
- Authors: Biao Li, Gao T. Wang and Suzanne M. Leal
- License: GNU General Public License (http://www.gnu.org/license/)
- URL: http://www.bioinformatics.org/simped/rare/
- Manuscript: Bioinformatics ...

1.2.1 Reporting problems, bugs and questions

If you have any problems using `RarePedSim` or would like to report a bug, please follow these steps:

When `RarePedSim` does not properly generate the **.ped* files in standard linkage format or fails to generate output files, please feel free to contact me:

`biaol` AT `bcm` DOT `edu`

but also please consider the following before doing so:

- Please go through this manual to get more familiar with `RarePedSim` commands if you have not done so yet.
- Please check the screen output, which may contain important ERROR information. Frequently, it need re-specify command arguments and/or options according to specific `RarePedSim` requirements.
- Please check the format of your input files, **.sfs*, **.conf* and/or **.ped*. E.g. Are columns in **.sfs* file delimited by single space? Is **.ped* file in standard linkage format? Does each row have correct number of values, which should be equal to number of columns? etc..

- Please check RarePedSim website if RarePedSim documentation or the program itself has been updated, sometimes the syntax of an option may change.
- If the above steps do not resolve your problem, please send an email preferably with the following specific information:
  - The complete screen ERROR messages
  - The type of machine you were using
  - Versions of Python installed
  - Ideally, please try to make some reduced sized input files that can replicate the problem, which may be zipped and sent as an attachment; any data sent to me for the purpose of debugging will be immediately deleted after that problem is resolved.

> **Important**
>
> We are willing and able to advise on the use of specific features implemented in RarePedSim, to diagnose whether they are working as intended and to give a generic description of a procedure or method, if it is unclear from reading the documentation.

1.3 Installation

1.3.1 Download

Please visit RarePedSim website to download the latest version of the program. We recommend to use our pre-built installation package (*.bundle) for easy installation on Mac (built on OS 10.8.5) and Linux (built on Ubuntu 11.10). For incompatible platforms, you will need to install from source code (*.src.tar.gz).

1.3.2 Installation package for Mac OS and Linux

We have built executable bundles on Mac (v10.8.5) and Linux (Ubuntu v11.10) for easy installation.

> **Note**
>
> These pre-built installation bundles should be upward compatible with Mac OS version > 10.8.5 and Ubuntu version > 11.10. If you might experience any problem using these bundles please report to us or compile and install RarePedSim from source code.

After it has been downloaded from the website you may make it executable and run the installer in terminal/command prompt. E.g.

BASH

```bash
chmod +x RarePedSim-<version>.bundle
./RarePedSim-<version>.bundle
```

The default installation directory /usr/local does require root privilege. Alternatively you can specify a local folder, such as $HOME/local, under your user account to have RarePedSim installed to $HOME/local/bin. You also need to add this folder to your system's PATH, such as,

```bash
export PATH=$HOME/local/bin:$PATH
```

1.3.3 Installation from source

To successfully compile the program from source `RarePedSim` requires to install the following dependency programs and Python packages:

programs

- Python (version 2.7.3+)
- GCC (version 4.7+)
- swig

Python packages

- numpy
- scipy
- apsw
- joblib
- progressbar
- simuPOP
- SEQPower

▪ Installation of dependencies on Linux OS (Debian, Ubuntu like)

```bash
apt-get install gcc g++ python python-numpy python-scipy python-apsw python-joblib python-progressbar swig python-dev build-essentia\
l libbz2-dev
```

then install `simuPOP` [1] and `SEQPower` [2]

▪ Installation of dependencies on Mac OS

- Install `Xcode` [3], select **Preferences** from `Xcode` menu and choose to install **Command Line Tools**
- Download and install `MacPorts` [4]

[1] `simuPOP` http://simupop.sourceforge.net/Main/Download
[2] `SEQPower` http://www.bioinformatics.org/spower/installation
[3] `Xcode` https://developer.apple.com/downloads/index.action
[4] `MacPorts` http://www.macports.org/install.php

- Use `MacPorts` to install gcc (v4.7+) and enable gcc (v4.7+) as system default gcc compiler, e.g. run

  ```BASH
  sudo port install gcc48
  sudo port select --set gcc mp-gcc48
  ```

- Download and install Anaconda [5] scientific Python distribution
- Use `MacPorts` to install `swig` and `swig-python`

  ```BASH
  sudo port install swig swig-python
  ```

- Download and install
  - `apsw` [6]

    ```BASH
    unzip apsw-<version>.zip; cd apsw-<version>
    python setup.py install
    ```

  - `progressbar` [7]

    ```BASH
    tar -xvzf progressbar-<version>.tar.gz; cd progressbar-<version>
    python setup.py install
    ```

  - `joblib` [8]

    ```BASH
    tar -xvzf -joblib-<version>.tar.gz; cd joblib-<version>
    python setup.py install
    ```

- Install `simuPOP` [9] and `SEQPower` [10]

Compile and install `RarePedSim`

```BASH
tar -xvzf RarePedSim-<version>.src.tar.gz
cd RarePedSim-<version>.src
python setup.py install
```

or specify `lib` and `bin` directories for local installation without root privilege, and add these directories to system's PATH

```BASH
python setup.py install --install-platlib=/path/to/lib --install-scripts=/path/to/bin
export PATH=/path/to/bin:$PATH
export PYTHONPATH=/path/to/lib:$PYTHONPATH
```

[5] Anaconda http://continuum.io/downloads

[6] `apsw` https://code.google.com/p/apsw/downloads/list

[7] `progressbar` http://code.google.com/p/python-progressbar/downloads/list

[8] `joblib` https://pypi.python.org/pypi/joblib#downloads

[9] `simuPOP` http://simupop.sourceforge.net/Main/Download

[10] `SEQPower` http://www.bioinformatics.org/spower/installation

Chapter 2

Reference Manual

2.1 Introduction

RarePedSim uses subcommand system that is similar to svn. Currently, it consists of two subcommands/modules, RarePedSim generate and RarePedSim srv, where the former one can simulate genotypes and phenotypes for specified pedigree structure(s) and phenotypic model, and the latter simulates rare variant data and creates site frequency spectrum (sfs) files. The *.sfs file contains variant data information in defined format and is a required input for RarePedSim generate command.

2.2 Program Commands

To display the command interface for all modules

```
rarepedsim -h
```

OUTPUT

```
usage: rarepedsim [-h] [--version,] {srv,generate} ...
Program to simulate pedigree-based gene/region-level genotype and phenotype data for
complex and Mendelian trait rare variant studies given any pedigree structures
positional arguments:
  {srv,generate}
    srv           create site frequency spectrum (sfs) from simulated rare
                  variant sequence data
    generate      generate genotype and phenotype for specified pedigree
                  structure(s)
optional arguments:
  -h, --help      show this help message and exit
  --version,      show program's version number and exit
Biao Li (biaol@bcm.edu) (c) 2014-2015. License: GNU General Public License
(http://www.gnu.org/licenses/)
```

To display the command interface for a specific module

```
rarepedsim generate -h
```

OUTPUT

```
usage: rarepedsim generate [-h] -s FILE -c FILE -p FILE [-o FILE] [-r INT]
                           [-g INT] [-e FLOAT] [-j INT] [-d NUM] [-b {-1,0,1}]
                           [-m {bz2,gz,zip}] [-f]
optional arguments:
  -h, --help            show this help message and exit
  -s FILE, --sfs_file FILE
                        Load site frequency spectrum file (*.sfs) file
  -c FILE, --config_file FILE
                        Load configuration file (*.conf) that includes
                        arguments and parameter values of phenotype model
  -p FILE, --ped_file FILE
                        Load linkage format file (*.ped) with user-specified
                        pedigree structure(s)
  -o FILE, --output_folder FILE
                        Specify output folder name (prefix),
                        /path/to/output_folder/ (default to ./output).
                        Simulation results will be saved in LINKAGE (ped)
                        format
  -r INT, --num_reps INT
                        Specify number of replicated data sets to generate per
                        gene (default to 1)
  -g INT, --num_genes INT
                        Specify number of genes to generate (default to 1), if
                        -1 use all available genes in arg.sfs_file
  -e FLOAT, --rec_rate FLOAT
                        Recombination rate on the gene region, default to 0
                        and max at 0.5
  -j INT, --num_jobs INT
                        Specify the number of jobs (CPUs) to use for
                        multiprocessing computation (default to -1). For -1
                        all CPUs are used; for 1 no parallel; for --num_jobs
                        below -1, CPU#s+1+num_jobs are used, e.g. for -2 all
                        CPUs but one are used.
  -d NUM, --seed NUM    Specify seed for random number generator, if left
                        unspecified the current system time will be used
  -b {-1,0,1}, --verbose {-1,0,1}
                        Specify screen output mode (-1, 0 or 1, default to 1);
                        where -1 -- quiet, no screen output; 0 -- minimum,
                        minimum output of program running progress; 1 --
                        regular, regular output of simulation progress and
                        time spent
  -m {bz2,gz,zip}, --compress {bz2,gz,zip}
                        Choose file format to compress simulated data, default
                        to None - no compression
  -f, --vcf             Also output simulated data in VCF(*.vcf) format in
                        addition to LINKAGE(*.ped) format.
```

```
rarepedsim srv -h
```

OUTPUT

```
usage: rarepedsim srv [-h] -c FILE
optional arguments:
  -h, --help            show this help message and exit
  -c FILE, --config_file FILE
                        Load configuration file (*.conf) which contains
                        arguments and parameter values to generate site allele
                        frequency spectrum (sfs) by forward-time evolutionary
                        simulation
```

2.3 Options to Generate Pedigree Data

Use command `rarepedsim generate` to simulate genotype and/or phenotype data for complex and Mendelian trait rare variant studies given any user-specified pedigree structures.

2.3.1 Input files

- `--sfs_file` **[required]**

The input site-specific frequency spectrum data file (`*.sfs` file) should have seven columns.

- **1st column**: Gene/region name. This column defines a genomic region/unit for variants with the same group name.
- **2nd column**: Chromosome name.
- **3rd column**: Variant position. A numeric value which represents either a relative or an absolute physical position.
- **4th column**: Reference allele (optional).
- **5th column**: Alternative allele(s) (optional, delimited by commas).
- **6th column**: MAF. Minor allele frequency of each variant site.
- **7th column**: Annotation score. This defines the functionality of a variant. In simulated data it can be quantities such as selection coefficients; in real sequence data it can be an annotation score such as `SIFT` or `Polyphen2` values. The annotation score is meaningful when there exists some cut-offs such that neutral, protective and deleterious variants can be defined by the scores compared to the cut-offs.

For example, an input data file with variant site information of gene CCDC85C retrieved from the ExAC [1] (Exome Aggregation Consortium) database should look like

OUTPUT

```
...
CCDC85C 14      100000999       A       G       0.000263782643102       0
CCDC85C 14      100001003       T       C       0.0100237404379         0
CCDC85C 14      100001011       C       A       0.000263782643102       1
CCDC85C 14      100001037       G       A       0.000131891321551       0
CCDC85C 14      100001060       G       A       0.0023746701847         0
CCDC85C 14      100001061       C       A       0.000131926121372       0
CCDC85C 14      100002302       T       C       0.009761663286          0
...
```

Alternatively, if reference and alternative alleles are unknown or not needed they can be omitted from the `sfs` file. Thus, a file of 5 columns can also be allowed and information about reference and alternative alleles will be specified as missing in the output `vcf` files if `--vcf` option is used.

[1] ExAC `http://exac.broadinstitute.org/`

OUTPUT

```
...
CCDC85C 14      100000999          0.000263782643102       0
CCDC85C 14      100001003          0.0100237404379         0
CCDC85C 14      100001011          0.000263782643102       1
CCDC85C 14      100001037          0.000131891321551       0
CCDC85C 14      100001060          0.0023746701847         0
CCDC85C 14      100001061          0.000131926121372       0
CCDC85C 14      100002302          0.009761663286          0
...
```

Note

> In the input `sfs` file, lines starting with "#" are comments and thus be ignored.

- `--config_file` **[required]**

The input configuration file (`*.conf`) includes options and parameter values of the phenotypic/disease model. Please use provided templates to update parameters. Use `MendelianPhenotype.conf` for Mendelian trait and `ComplexPhenotype.conf` for complex trait simulation, respectively. For complete description of each option refer to **Appendix**.

Important

> Download zipped data package `data.zip` file from `RarePedSim` download page [a], unzip it and use the provided template `*.conf` file to specify phenotypic model parameter values.
>
> [a] `RarePedSim` download page http://www.bioinformatics.org/simped/rare/download/

- `--ped_file` **[required]**

This input file should contain information of pedigree structure, sex and phenotype (optional) of each sample. It is required to have 6 columns (phenotypes are known) which should follow the first 6 columns of the `LINKAGE format` [2] convention. The 6 columns are Family ID, Individual ID, Paternal ID, Maternal ID, Sex and Phenotype. An additional 7th column can be specified to indicate missingness of individuals' genotypes (0 for missing and 1 for existing). Without including a 7th column all individuals are assumed to be genotyped.

[2] `LINKAGE format` http://www.jurgott.org/linkage/LinkagePC.html

⚠ **Warning**

- Individual IDs are required to be unique within each family
- Columns need to be whitespace(s) delimited, such as space(s) and tab(s).
- Sex for male and female is required to be coded by `1` or `m` or `male`, and `2` or `f` or `female`, respectively (case-insensitive)
- For qualitative (case-control) trait, controls and cases are required to be coded by `1` and `2`, respectively. A missing value can be coded by `0` or `na` or `n/a` or `none` or `null` or `missing` or `unknown` (case-insensitive)
- Quantitative trait values are assumed to be Gaussian distributed and required to be standardized. A missing value can be coded by `na` or `n/a` or `none` or `null` or `missing` or `unknown` (case-insensitive).

2.3.2 Additional input options

- **`--rec_rate` (default to `0`)**

Specify recombination rate on the gene/region.

Note

We assume that recombination can occur at most once on the specified gene/region if `--rec_rate` $>$ `0`, because the occurrence of one crossover usually inhibits the formation of another crossover in its vicinity due to *positive interference*.

2.3.3 Output and runtime options

- **`--output_folder` [default to `output`]**

Specify an absolute path to the output folder (/path/to/output_folder/). Simulation results will be saved in `LINKAGE (ped)` and `Variant Call (vcf)` (optional) formats.

- **`--num_genes` [default to 3]**

Specify number of genes/regions to generate. Use `-1` to generate for all available genes/regions contained in `--sfs_file`.

- **`--num_reps` [default to 1]**

Specify number of replicates to generate for each gene/region.

- **`--num_jobs` [default to -1]**

Specify the number of jobs (CPUs) to use for `RarePedSim`.

- -1 – all CPUs are used
- 1 – single CPU is used (no parallel computation)
- $<$ -1 – No.CPUs + 1 + `num_jobs` are used (e.g. for -2 all CPUs but one are used).

▪ `--seed` **[default to `None`]**

Seed for random number generator. If left unspecified the current system time will be used.

▪ `--verbose` **[default to 1]**

Screen output mode, where

- 1 – output all simulation progress and information
- 0 – minimum output of simulation progress and information
- -1 – no screen output

▪ `--compress` **[default to None]**

No compression on simulated data if left unspecified, or choose compressed file format (`gzip`, `bgzip` or `zip`).

> **Note**
> Data compression speed depends on disk I/O. In case of poor disk performance and/or large data volume to compress you may experience long wait time.

2.4 Options to Simulate Rare Variants Sequence Data

Use command `rarepedsim srv` to perform a forward time simulation on an evolutionary process under a general Wright-Fisher model with arbitrary demographic, selective and mutational effects. The output results in a site frequency spectrum (`sfs`) file, which contains variant data information in defined format and is a required input for `rarepedsim generate` command.

This module has been implemented and adapted from `srv` [3], a forward time simulation model, which is superior to other methods, e.g. coalescent simulation, since it can incorporate realistic evolutionary scenarios that consist of multi-stage demography with varying population sizes, variable mutation schemes, locus-specific selection coefficients including purifying selection and multi-locus fitness effect. In default setting, we allow about 37% of variant sites to be synonymous according to NHLBI Exome sequence data [4], those that have purifying selection coefficients greater than 10-5 to be deleterious and those less than -10-5 to be protective (Ref [5]). Using existing demographic models we can reconstruct variant data for complex populations.

[3] `srv` http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3164177/

[4] NHLBI Exome sequence data http://www.sciencemag.org/content/337/6090/64.abstract

[5] Ref http://www.ncbi.nlm.nih.gov/pubmed/23834317

Variant data information can also be retrieved from real sequence data, such as exome variant data discovered from the NHLBI Exome Sequencing Project.

2.4.1 Input file and runtime option

- `--config_file` **[required]**

The input configuration file (`*.conf`) includes arguments and parameter values of the demographic model. Please use provided template, `SimulateRareVariant.conf` to update parameters. For complete description of each parameter refer to **Appendix**.

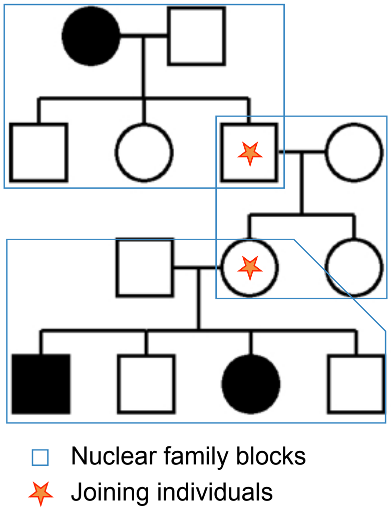
Chapter 3

Methods

3.1 Generation of Pedigree Segregating Data for Mendelian Trait

For Mendelian trait studies, phenotypes are generally given. Thus, genotypes are to be simulated conditional on observed discrete phenotypic values, user-specified (reduced) penetrance model and variant data information (contained in the input `sfs file`).

For any arbitrarily complex pedigree structure we can split it into N blocks of nuclear families connected by joining individuals (JIs), those who have both parent(s) and 1-degree related offspring contained in the pedigree (non-founders with offspring). For example, as shown in the figure below, a three generational pedigree has $N = 3$ nuclear family blocks connected by two JIs, who are marked by stars.



Within the i^{th} block, $i \in \{1, 2, ..., N\}$, there are only parents and their 1-degree related offspring, where we denote n_i as number of individuals contained in the block; $\mathbf{T}^{(i)} = \left\{T_1^{(i)}, T_2^{(i)}, ..., T_{n_i}^{(i)}\right\}$ as phenotypes of individuals (subscripts 1 - father, 2 - mother, $\{3, 4, ..., n_i\}$ - offspring), where $T_j^{(i)} = t_j^{(i)}$ is the phenotype of the j^{th} individual in the i^{th} nuclear family block and $T_j^{(i)} \in \{0, 1, 2\}$ (0 - unknown, 1 - unaffected/control, 2 - affected/case).

We define events $\mathbf{X}^{(i)} = \left\{X_1^{(i)}, X_2^{(i)}, ..., X_{n_i}^{(i)}\right\}$, where $X_j^{(i)} = \left(X_{j,1}^{(i)}, X_{j,2}^{(i)}\right)$, denotes the genotypic pattern of the j^{th} individual within i^{th} block, which is determined by two haplotype patterns.

Note

- A haplotype frequency is measured by the number of carried causal variants on the haplotype.
- For Mendelian trait simulation the probability of being affected of any individual depends on the number of carried disease penetrant haplotypes (0, 1, or 2) and mode of inheritance (dominant, recessive or compound recessive, etc).

For each nuclear family block, we can derive

$$\begin{aligned} P\left(\mathbf{X}^{(i)} \mid \mathbf{T}^{(i)}\right) = & \frac{P\left(\mathbf{T}^{(i)} \mid \mathbf{X}^{(i)}\right) \cdot P\left(\mathbf{X}^{(i)}\right)}{P\left(\mathbf{T}^{(i)}\right)} \\ & \propto P\left(T_1^{(i)} = t_1^{(i)}, ..., T_{n_i}^{(i)} = t_{n_i}^{(i)} \mid \mathbf{X}^{(i)}\right) \cdot P\left(\mathbf{X}^{(\mathbf{i})}\right) \\ & \propto \left[\prod_{j=1}^{n_i}\left(P\left(T_j^{(i)} \mid X_j^{(i)}\right) \cdot I\left(T_j^{(i)} \neq 0\right) + I\left(T_j^{(i)} = 0\right)\right)\right] \cdot \left[\prod_{j=3}^{n_i} P\left(X_j^{(i)} \mid X_1^{(i)}, X_2^{(i)}\right)\right] \\ & \cdot \left[\prod_{j=1}^{2}\left(P\left(X_j^{(i)}\right) \cdot I\left(X_j^{(i)}\, is\, founder\right) + I\left(X_j^{(i)}\, is\, not\, founder\right)\right)\right] \end{aligned}$$

Based on the equation above, joint likelihoods of genotypic patterns of all JIs (denoted by $\mathbf{J}$ can be inferred, $L\left(\mathbf{X}^{(\mathbf{J})} \mid \mathbf{T}\right)$, given known phenotypes $\mathbf{T}$, disease penetrance model $P\left(T_j^{(i)} \mid X_j^{(i)}\right)$ and haplotype frequency distribution $P\left(X_j^{(i)}\right)$. Then based upon the joint likelihoods of the JIs the conditional genotypic likelihoods are updated for all pedigree members, since once $\mathbf{X}^{(\mathbf{J})}$ have been determined the joint likelihoods for individuals within each nuclear family block can be updated, $L\left(\mathbf{X}^{(i)} \mid \mathbf{T}^{(i)}, \mathbf{X}^{(\mathbf{J})}\right)$, and the probability distribution of genotypic pattern for each individual contained in the block can be obtained, $P\left(X_j^{(i)} \mid \mathbf{T}^{(i)}, \mathbf{X}_j^{(\mathbf{J})}\right)$.

3.2 Complex Trait Phenotypic Models

Given any arbitrary pedigree structures for complex trait rare variant association studies, `RarePedSim` can generate genotypes of founders based on variant data (the input `sfs file`) and non-founders according to principles of segregation. Then phenotypes can be generated based on genotypes and user-specified phenotypic models.

To simulate genetic associations, the functionality of variants within a region can be either unidirectional, e.g. all variants increase disease risk or bidirectional, e.g. some variants increase while other decrease quantitative trait values.

For qualitative traits such as disease status, `RarePedSim` implemented logistic model of odds ratio (LOGIT). The logistic model applies the logistic function with user specified odds ratio(s) to calculate the joint penetrance of variants in a region/gene for a given multi-locus genotype to determine the probability of being affected. Under a fixed effect model the given odds ratio is considered as constant for all rare variants, while in variable effects model the magnitude of odds ratios becomes site-specific and is determined by MAF ($\gamma \propto \frac{1}{p}$, γ - odds ratio, p - MAF).

For quantitative trait (QT) simulation the RarePedSim incorporated linear mean-shift model (LNR), where QT is assumed to be normally distributed and contribution of functional variants alters the QT distribution by a shifted mean but unchanged variance. The trait value is then computed by a linear model given multi-locus genotype.

Important

> Refer to **Appendix** for complete description of each parameter of the complex trait phenotype model.

3.3 Generation of Pedigree Segregating Data for Complex Trait

For complex trait if phenotypes are known and genotypes need to be generated conditional on phenotypes, `RarePedSim` can efficiently generate genotypes without ascertainment given that only rare variants are assumed to be causal under LOGIT or LNR fixed effect phenotype model. Additionally, for more complicated disease models users can ascertain from repeatedly simulated pedigrees those that have desired phenotypic patterns.

The same strategy is used here as that shown above for Mendelian trait simulation, where phenotypes, penetrance and haplotype frequency distribution are required (see **Generation of Pedigree Segregating Data for Mendelian Trait**). The only difference here is to obtain the penetrance. The penetrance in Mendelian trait simulation is an user input, whereas in Complex trait simulation it need to be derived from effect size of causal variants (odds ratio in LOGIT model and mean-shift in LNR model).

Note

> In order to efficiently generate genotype data conditional on known phenotypes we need to assume 1) fixed effect model, and 2) only rare variants being causal

Given that an individual carries n causal rare variants ($n \geqslant 1$):

For qualitative trait simulation using LOGIT model with odds ratio for a causal variant γ and disease baseline prevalence k, `RarePedSim` allows for the following mode of inheritance on the region of interest with multiple causal variant sites:

- *D* - Dominant
- *R* - Recessive
- *DAR* - Dominant Across Region, which allows for compound heterozygote across multiple causal variant sites
- *RAR* - Recessive Across Region, which allows for compound heterozygote across multiple causal variant sites
- *AAR* - Additive Across Region, where for an individual if only one haplotype carries causal variant(s) the increased odds ratio would be γ and if both haplotypes do the increased odds ratio would be $2 * \gamma - 1$

- *MAR* - Multiplicative Across Region, where for an individual if both haplotypes carry causal variants the increased odds ratio would γ^2
- *MAV* - Multiplicative Across Variant sites, where the increased odds ratio would be γ^n for an individual carrying a total number of n causal variants across all variant sites.

For example, if *MAV* mode of inheritance is applied, for any individual that carries n causal variants, the cumulative effect size is assumed to be $R = \gamma^n$ for a multi-locus genotype. Since $R = \frac{p/1-p}{k/1-k}$, we have $p = \frac{Rk}{1-k+Rk}$, where p denotes penetrance.

For quantitative trait simulation using LNR model with mean shift μ:

> **Warning**
>
> To generate quantitative trait phenotypes conditional on simulated genotypes `RarePedSim` uses standard Gaussian distribution ($\mu = 0, \sigma = 1$) as the null model for wildtype genotypes. If `RarePedSim` is used to generate genotypes conditional on known quantitative traits, phenotypes are assumed to be Gaussian distributed and required to be standardized.

Denote Gaussian distributed random variables $N_1(0, 1)$ and $N_2(n\mu, 1)$. Given a quantitative trait random variable T and its observed value $T = t$ denote event $W = 1$ as T drawn from N_1 and event $W = 2$ as T drawn from N_2, respectively. Then, we can obtain

$$\begin{aligned}
\frac{P\left(W = 1 \mid T = t\right)}{P\left(W = 2 \mid T = t\right)} &= \frac{P\left(T = t \mid W = 1\right) \cdot P\left(W = 1\right)}{P\left(T = t \mid W = 2\right) \cdot P\left((W = 2\right)} \\
&= \frac{\lim_{dt \to 0} \int_t^{t+dt} f_1(t)dt}{\lim_{dt \to 0} \int_t^{t+dt} f_2(t)dt} \cdot \frac{P\left(W = 1\right)}{P\left(W = 2\right)} \\
&= \frac{f_1(t)}{f_2(t)} \cdot \frac{P\left(W = 1\right)}{P\left(W = 2\right)}
\end{aligned}$$

Note if priors $P\left(W = 1\right) = P\left(W = 2\right)$, $\frac{P(W=1|T=t)}{P(W=2|T=t)} = \frac{f_1(t)}{f_2(t)}$.

Given an observed quantitative trait value of an individual we are able to infer the likelihood that from which Gaussian distribution the trait value is sampled, which indicates likelihood of the underlying number of causal variants (here denoted by n) carried by the individual.

Chapter 4

Quick Guide and Examples

RarePedSim was used to perform pedigree simulation on a number of examples including both Mendelian and Complex trait scenarios. All examples were run on a 64-bit Linux machine with Intel Core i7-4770k 3.5GHz CPU and 16GB RAM.

Important

- Download the zipped data file from RarePedSim download page [a], unzip it and go to data folder.
- Rare variant sequence data have been generated and results are provided (contained in the downloaded data folder) to use as input Gazave2013European1800.sfs file for all examples below. The Gazave2013European.sfs file has been generated using rarepedsim srv -c Gazave2013European.conf command, where in the configuration file the European demographic model from Gazave et. al., 2013 [b] and a European purifying selection model from Kyrukov et. al., 2009 [c] have been adopted to generate variant data information on genes (10 replicates) with 1,800 base pair. More details about rarepedsim srv can be found in **Reference Manual** and **Appendix**.
- For details about command options and features that are not covered by examples please refer to **Reference Manual** and **Methods** chapters.

[a] RarePedSim download page http://www.bioinformatics.org/simped/rare/download/

[b] Gazave et. al., 2013 http://www.pnas.org/content/111/2/757.abstract

[c] Kyrukov et. al., 2009 http://www.pnas.org/content/106/10/3871.full

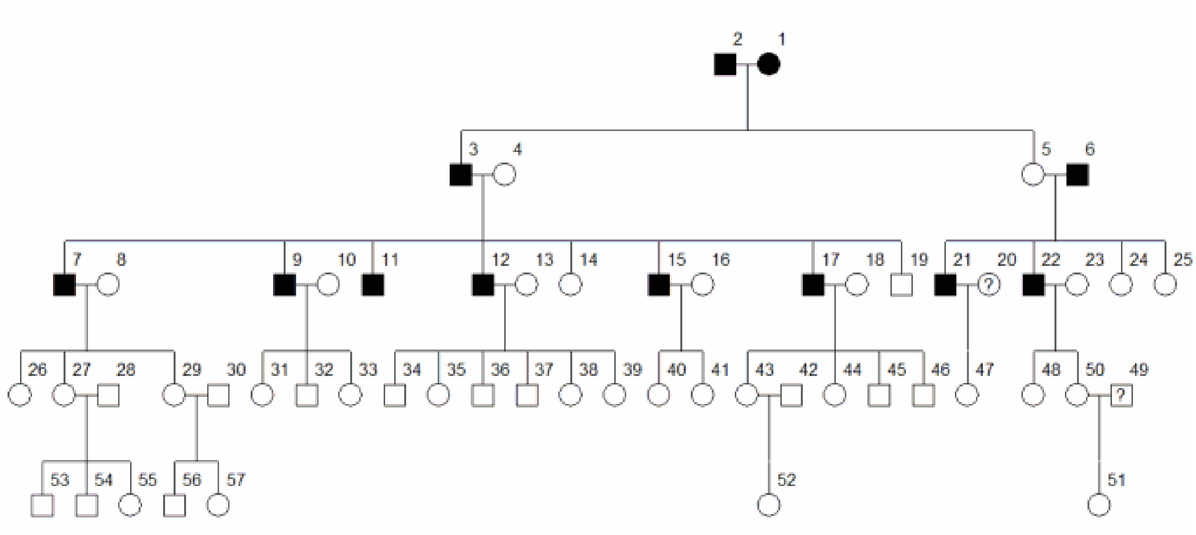
4.1 Mendelian Trait Simulation

4.1.1 A Multi-generational large pedigree

This example uses RarePedSim to generate genotypes on a multi-generational pedigree (shown below)

- 57 family members with known disease status
- 12 affected individuals

- Dominant model with reduced penetrance and locus heterogeneity on two susceptible genes



Run the following:

- Create a result directory under /tmp/ (or any other preferred directory)

Note

RarePedSim can also create result directory on the fly if it does not exist and print out a warning message that a new directory has been created.

BASH

```
mkdir /tmp/result_eg_1
```

- Go to downloaded data folder
- Run rarepedsim generate command

BASH

```
rarepedsim generate --sfs_file Gazave2013European1800.sfs --config_file MendelianTrait_eg1.conf --ped_file 57Inds.ped --output_folde\
r /tmp/result_eg_1/ --num_reps 20 --num_genes 3 --vcf
```

OUTPUT

```
INFO:Begin Mendelian trait simulation for pedigrees
INFO:Retrieving causal variant sites information from Gazave2013European1800.sfs
INFO:Calculating gene-level causal haplotype frequency
INFO:Begin analyzing pedigree-wise genotype frequencies for genes R1 and R2
[Parallel(n_jobs=-1)]: Done   1 out of   1 | elapsed:   15.9s remaining:    0.0s
[Parallel(n_jobs=-1)]: Done   1 out of   1 | elapsed:   15.9s finished
[Parallel(n_jobs=-1)]: Done   1 out of   1 | elapsed:   17.2s remaining:    0.0s
[Parallel(n_jobs=-1)]: Done   1 out of   1 | elapsed:   17.2s finished
INFO:Simulation in progress for gene pair R1_R2
 100% |...................................................................................| Time: 0:00:02
INFO:Saving simulated data for gene R1_R2 to /tmp/result_eg_1/R1_R2
INFO:Begin analyzing pedigree-wise genotype frequencies for genes R1 and R3
[Parallel(n_jobs=-1)]: Done   1 out of   1 | elapsed:   16.8s remaining:    0.0s
[Parallel(n_jobs=-1)]: Done   1 out of   1 | elapsed:   16.8s finished
INFO:Simulation in progress for gene pair R1_R3
 100% |...................................................................................| Time: 0:00:02
```

```
INFO:Saving simulated data for gene R1_R3 to /tmp/result_eg_1/R1_R3
INFO:Begin analyzing pedigree-wise genotype frequencies for genes R2 and R1
...
```

Tip

The input `--ped_file 57Inds.ped` is like:

Family_ID Individual_ID Father_ID Mother_ID Gender Phenotype
...
1 4 0 0 2 1
1 5 2 1 2 1
1 6 0 0 1 2
1 7 3 4 1 2
1 8 0 0 2 1
1 9 3 4 1 2
...

Tip

The input `--sfs_file Boyko2008African1800.sfs` looks like:

#name chr position maf annotation
...
R2 1-2 1790 0.00001300 0.00000000
R2 1-2 1795 0.00000688 0.00771062
R2 1-2 1797 0.00001606 0.00591815
Replicate #3 gene length = 1800
R3 1-3 14 0.00000229 0.00001000
R3 1-3 25 0.00000076 0.00008347
R3 1-3 35 0.00000076 0.00027948
R3 1-3 37 0.00000765 0.00341390
...

Tip

The input `--config_file MendelianTrait_eg1.conf` is like:

TEXT

```
trait_type=Mendelian
[quality control]
def_rare=0.01
rare_only=True
[phenotype parameters]
moi=D
compound_hetero=False
penetrance=(0.001, 0.9, 1)
proportion_causal=1
mode=pairwise
locus_hetero=(0.75, 0.25)
[genotyping artifact]
missing_low_maf=1e-06
missing_sites=0.01
missing_calls=0.05
error_calls=0.05
```

Note

> For details about each option in `config_file` refer to **Appendix**.

- View results

```bash
cd /tmp/result_eg_1
cd R1
vim rep1.ped
vim rep1.vcf
```

Note

> Each marker is represented by two columns (starting at the 7th column), one for each allele

4.1.2 562 nuclear family samples

This example applies `RarePedSim` to simulate genotypes for 562 nuclear families.

- 2-6 offspring per family and at least one offspring being affected
- Parents being either unaffected, affected or unknown status
- Total number of individuals is 2,520 where 1,235 are affected, 932 unaffected and 353 unknown.
- Autosomal dominant model with reduced penetrance and phenocopy effect.

Run the following:

- Create a result folder under `/tmp/`

```bash
mkdir /tmp/result_eg_2
```

- Go to downloaded data folder
- Run `rarepedsim generate` command

```bash
rarepedsim generate --sfs_file Gazave2013European1800.sfs --config_file MendelianTrait_eg2.conf --ped_file 562NucFams.ped --output_f\
older /tmp/result_eg_2/ --num_reps 10 --num_genes 3 --vcf
```

```
INFO:Begin Mendelian trait simulation for pedigrees
INFO:Retrieving causal variant sites information from Gazave2013European1800.sfs
INFO:Calculating gene-level causal haplotype frequency
INFO:Begin analyzing pedigree-wise genotype frequencies for gene R1
[Parallel(n_jobs=-1)]: Done   1 out of 107 | elapsed:    0.0s remaining:    3.4s
[Parallel(n_jobs=-1)]: Done 112 out of 562 | elapsed:    0.6s remaining:    2.3s
[Parallel(n_jobs=-1)]: Done 225 out of 562 | elapsed:    1.2s remaining:    1.8s
[Parallel(n_jobs=-1)]: Done 338 out of 562 | elapsed:    1.6s remaining:    1.1s
[Parallel(n_jobs=-1)]: Done 451 out of 562 | elapsed:    2.0s remaining:    0.5s
[Parallel(n_jobs=-1)]: Done 562 out of 562 | elapsed:    2.6s finished
INFO:Simulation in progress for gene R1
 100% |........................................................................| Time: 0:00:16
INFO:Saving simulated data for gene R1 to /tmp/result_eg_2/R1
...
```

Tip

The input `--ped_file 562NucFams.ped` is like:

Family_ID Individual_ID Father_ID Mother_ID Gender Phenotype
...
44 185 181 182 1 1
45 186 0 0 1 1
45 187 0 0 2 0
45 188 186 187 2 2
45 189 186 187 1 2
45 190 186 187 1 1
46 191 0 0 1 1
46 192 0 0 2 0
46 193 191 192 2 2
...

Tip

The input `--config_file MendelianTrait_eg2.conf` looks like:

TEXT

```
trait_type=Mendelian
[quality control]
def_rare=0.01
rare_only=True
[phenotype parameters]
moi=D
compound_hetero=True
penetrance=(0.05, 0.45, 0.45)
proportion_causal=1
mode=single
locus_hetero=(1, 0)
[genotyping artifact]
missing_low_maf=1e-06
missing_sites=0.0
missing_calls=0.0
error_calls=0.0
```

Note

For details about each option in `config_file` refer to **Appendix**.

- View results

BASH

```bash
cd /tmp/result_eg_2
cd R1
vim rep1.ped
vim rep1.vcf
```

4.2 Complex Trait Simulation

4.2.1 Qualitative trait with LOGIT model

`RarePedSim` was used to generate genotypes for 2,000 trio samples.

- Affected offspring and unknown parental disease status.

- Logistic odds ratio model with baseline penetrance 0.01.
- Causal rare variants with fixed effect size odds ratio = 2.5.

Run the following:

- Create a result folder under `/tmp/`

```bash
mkdir tmp/result_eg_3
```

- Go to downloaded data folder
- Run `rarepedsim generate` command command

```bash
rarepedsim generate --sfs_file Gazave2013European1800.sfs --config_file ComplexTrait_eg3_qualitative.conf --ped_file 2000Trios.ped -\
-output_folder /tmp/result_eg_3/ --num_reps 2 --num_genes 2 --vcf
```

```
INFO:Begin Complex trait simulation for pedigrees
INFO:For fixed effect LOGIT model with rare variants being causal, genotypes can be generated conditional on provided
 phenotypes
INFO:Retrieving causal variant sites information from Gazave2013European1800.sfs
INFO:Begin analyzing pedigree-wise genotype frequencies for gene R1
[Parallel(n_jobs=-1)]: Done   1 out of 176 | elapsed:    0.0s remaining:    6.5s
[Parallel(n_jobs=-1)]: Done 398 out of 2000 | elapsed:    3.6s remaining:   14.5s
[Parallel(n_jobs=-1)]: Done 799 out of 2000 | elapsed:    7.1s remaining:   10.7s
[Parallel(n_jobs=-1)]: Done 1200 out of 2000 | elapsed:   10.9s remaining:    7.3s
[Parallel(n_jobs=-1)]: Done 1601 out of 2000 | elapsed:   14.8s remaining:    3.7s
[Parallel(n_jobs=-1)]: Done 2000 out of 2000 | elapsed:   18.4s finished
INFO:Simulation in progress for gene R1
 100% |..............................................................................| Time: 0:00:17
INFO:Saving simulated data for gene R1 to /tmp/result_eg_3/R1
INFO:Begin analyzing pedigree-wise genotype frequencies for gene R2
...
```

Tip

The input `--ped_file 2000Trios.ped` is like:

Family_ID Individual_ID Father_ID Mother_ID Gender Phenotype
...
8 22 0 0 1 0
8 23 0 0 2 0
8 24 22 23 1 2
9 25 0 0 1 0
9 26 0 0 2 0
9 27 25 26 1 2
10 28 0 0 1 0
10 29 0 0 2 0
10 30 28 29 1 2
...

Tip

The input `--config_file ComplexTrait_eg3_qualitative.conf` looks like:

```TEXT
trait_type=Complex
[model]
model=LOGIT
[quality control]
def_rare=0.01
rare_only=True
...
[LOGIT model]
OR_rare_detrimental=2.5
...
[genotyping artifact]
missing_low_maf=0.001
missing_sites=0.001
missing_calls=0.001
error_calls=0.001
...
```

Note

For details about each option and its value contained in `ComplexTrait_eg3_qualitative.conf` refer to **Appendix**.

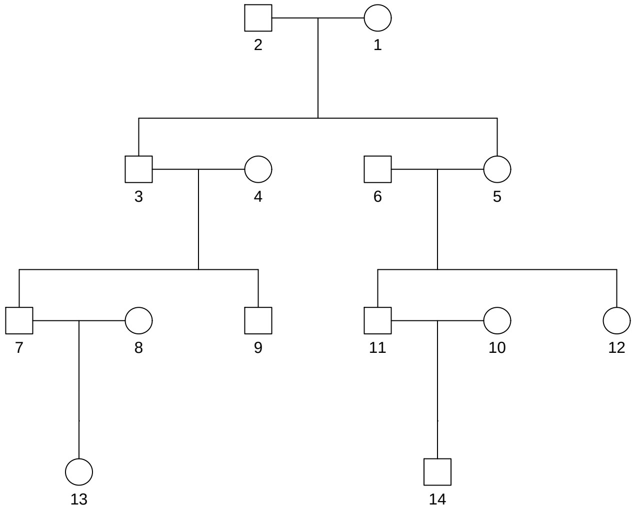
- View results

```BASH
cd /tmp/result_eg_3
cd R1
vim rep1.ped
vim rep1.vcf
```

4.2.2 Quantitative trait with LNR model

`RarePedSim` was used to generate genotypes for the multi-generational large pedigree (14 individuals in 4 generations) assuming quantitative phenotypes.

- Linear mean shift model with baseline quantitative trait $\sim Gaussian(0, 1)$
- Causal rare variants with fixed effect size of mean shift = 1.0.



Run the following:

- Create a result folder under `/tmp/`

BASH

```
mkdir tmp/result_eg_4
```

- Go to downloaded data folder
- Run `rarepedsim generate` command

BASH

```
rarepedsim generate --sfs_file Gazave2013European1800.sfs --config_file ComplexTrait_eg4_quantitative.conf --ped_file 4Gen.ped --out\
put_folder /tmp/result_eg_4/ --num_reps 100 --num_genes 5 --vcf
```

OUTPUT

```
INFO:Begin Complex trait simulation for pedigrees
INFO:For fixed effect LNR model with rare variants being causal, genotypes can be generated conditional on provided p
henotypes
INFO:Retrieving causal variant sites information from Gazave2013European1800.sfs
INFO:Begin analyzing pedigree-wise genotype frequencies for gene R1
[Parallel(n_jobs=-1)]: Done   1 out of   1 | elapsed:    2.8s remaining:    0.0s
[Parallel(n_jobs=-1)]: Done   1 out of   1 | elapsed:    2.8s finished
INFO:Simulation in progress for gene R1
 100% |..................................................................................| Time: 0:00:02
INFO:Saving simulated data for gene R1 to /tmp/result_eg_4/R1
INFO:Begin analyzing pedigree-wise genotype frequencies for gene R2
...
```

Tip

The input `--ped_file 4Gen.ped`:

Family_ID	Individual_ID	Father_ID	Mother_ID	Gender	Phenotype
1	1	0	0	2	1.7
1	2	0	0	1	1.2
1	3	2	1	1	1.2
1	4	0	0	2	0.3
1	5	2	1	2	2.0
1	6	0	0	1	0.1
1	7	3	4	1	0.7
1	8	0	0	2	2.1
1	9	3	4	1	1.9
1	10	0	0	2	3.3
1	11	6	5	1	0.9
1	12	6	5	2	1.6
1	13	7	8	2	1.3
1	14	11	10	1	2.4

Tip

The input `--config_file ComplexTrait_eg4_quantitative.conf` looks like:

TEXT

```
trait_type=Complex
[model]
model=LNR
...
[LNR model]
meanshift_rare_detrimental=1.0
...
[genotyping artifact]
missing_low_maf=None
missing_sites=None
missing_calls=None
error_calls=None
...
```

Note

For details about each option and its value contained in `ComplexTrait_eg4_quantitative.conf` refer to **Appendix**.

- View results

BASH

```bash
cd /tmp/result_eg_4
cd R1
vim rep1.ped
vim rep1.vcf
```

Chapter 5

Appendix

5.1 Phenotype Model Configuration Options

The input configuration file (*.conf) contains options to specify phenotype model. See below for description to each option.

> **Important**
> The first line of any *.conf file is required to specify the trait type (Mendelian/mendelian/M/m or Complex/complex/C/c), e.g. `trait_type=Mendelian`, `trait_type=C`.

5.1.1 Mendelian trait options

> **Note**
> Use provided template `MendelianPhenotype.conf` to update parameter values. Lines start with '[' and end with ']' are comments and will not be parsed by the program.

- **[quality control]**
  - `def_rare` (default to `0.01`) – definition of rare variant sites: variant site having MAF smaller than that will be considered as a 'rare' variant site; the opposite is 'common'.
  - `rare_only` (default to `True`) – only rare variant sites can be informative, i.e., sites with MAF > `def_rare` are non-informative.
- **[phenotype parameters]**
  - `moi` (default to `D`; choose between `D` - dominant and `R` - recessive) – mode of inheritance.
  - `compound_hetero` (default to `False`) – if or not to allow compound heterozygote (an individual may have different mutant alleles at a given locus)

- `penetrance` (default to `(0, 1, 1)`) – specify penetrance for Mendelian inheritance of 3 possible genotypes, *aa*, *Aa*, *AA*, respectively; where *A* denotes a haplotype that carries at least one causal variant and *a* for haplotype carrying no causal variant, e.g., `penetrance=(0.01, 0.9, 1)` specifies dominant mode with reduced penetrance on *Aa* and phenocopy effect on *aa*; `penetrance=(0, 0, 1)` specifies a fully penetrant recessive trait without phenocopy effect.
- `proportion_causal` (default to `1`) – proportion of informative variant sites to be causal, i.e., the random set of the rest $(1 - p) \times 100\%$ do not contribute to the phenotype in simulation yet will present as noise.
- `mode` (default to `single`, choose between `single` and `pairwise`) – use `pairwise` to simulate genotypes on two loci (will iterate over all combinations of pairs of genes) with locus heterogeneity, otherwise use `single` to simulate on each single gene.
- `locus_hetero` (default to `(0.5, 0.5)`) – probability of each gene locus being causal between paired genes (sum should be equal to 1).

[genotyping artifact]

- `missing_low_maf` (default to `None`) – variant sites having MAF $<$ P are set to be missing (e.g. 1e-06).
- `missing_sites` (default to `None`) – proportion of missing variant sites (e.g. 0.01).
- `missing_calls` (default to `None`) – proportion of missing genotype calls e.g. 0.05).
- `error_calls` (default to `None`) – proportion of error genotype calls (e.g. 0.05).

5.1.2 Complex trait options

Note

> Use provided template `ComplexPhenotype.conf` to update parameter values. Lines start with '[' and end with ']' are comments.

[model]

- `model` (default to `LOGIT`, choose from `LOGIT` - logistic odds ratio model and `LNR` - linear mean-shift quantitative trait model) – specify which complex trait phenotype model to use

[quality control]

- `def_rare` (default to `0.01`) – definition of rare variant sites: variant site having MAF smaller than that will be considered as a 'rare' variant site; the opposite is 'common'.
- `rare_only` (default to `True`, boolean parameter, choose between `True` and `False`) – only rare variant sites can be associated with the trait, i.e., sites with MAF $>$ `def_rare` are neutral.
- `def_neutral` (default to `(-1e-5, 1e-5)`) – annotation value cut-offs that defines a variant to be 'neutral' (e.g. synonymous, non-coding etc. that will not contribute to any phenotype); any variant with function_score falling in this range will be considered neutral.

- `def_protective` (default to `(-1, -1e-5)`) – annotation value cut-offs that defines a variant to be 'protective' (i.e., decrease disease risk or decrease quantitative traits value); any variant with function_score falling in this range will be considered protective.

▪ **[phenotype parameters]**

- `baseline_effect` (default to `0.01`) – penetrance of wildtype genotypes.
- `moi` (default to `D`) – mode of inheritance, choose from `D` - Dominant; `R` - Recessive; `DAR` - Dominant Across Region; `RAR` - Recessive Across Region; `AAR` - Additive Across Region; `MAR` - Multiplicative Across Region; `MAV` - Multiplicative Across Variant sites (more details about each moi can be found at Chapter 3.3).
- `proportion_causal` (default to `None`) – proportion of functional variants associated with the trait of interest, i.e., if a value p is specified, the random set of the rest $(1 - p) \times 100\%$ functional variants are non-causal: they do not contribute to the phenotype in simulations yet will present as noise.

▪ **[LOGIT model]**

Note

Options and parameter values for LOGIT model.

- `OR_rare_detrimental` (default to `1.0`) – odds ratio for detrimental rare variants (> 1.0).
- `OR_rare_protective` (default to `None`) – odds ratio for protective rare variants (< 1.0).
- `ORmax_rare_detrimental` (default to `None`) – maximum odds ratio for detrimental rare variants, applicable to variable effects model.
- `ORmin_rare_protective` (default to `None`) – minimum odds ratio for protective rare variants, applicable to variable effects model.
- `OR_common_detrimental` (default to `None`) – odds ratio for detrimental common variants, applicable to `rare_only=False`.
- `OR_common_protective` (default to `None`) – odds ratio for protective common variants, applicable to `rare_only=False`.

Note

These options specify the odds ratios of variants in the region of interest. When `OR_rare_detrimental` is used by itself, all detrimental rare variants will be assigned a fixed odds ratio as specified (fixed effect model). With the `ORmax_rare_detrimental` option, they together model variable effects with `OR_rare_detrimental` being the minimum odds ratio and `ORmax_rare_detrimental` will be assigned to the variant site having smallest MAF, and the minimum odds ratio to the one having largest MAF. Other odds ratios in between are interpolated based on max. and min. values. Similarly as above `OR_rare_protective` and `ORmin_rare_protective` are for protective variants. `OR_common_detrimental` and `OR_common_protective` are odds ratio for common detrimental and protective variants respectively. No variable effects model for common variants is available.

Important

> In order to efficiently simulate conditional genotypes without ascertainment for pedigrees based on known phenotypes using `RarePedSim`, a fixed effect model with only detrimental rare variants being causal is required (`rare_only=True` and only `OR_rare_detrimental` is specified). Otherwise, users need to ascertain from repeatedly simulated samples those that have desired phenotypic patterns. A simple ascertaining scheme is provided at the bottom of the configuration file. More complicated ones are subject to users' own discretion to ascertain from replicates of simulated data in result files (`*.ped or *.vcf`).

- **[LNR model]**

Note

> Options and parameter values for LNR model.

Warning

> If genotypes are generated conditional on known quantitative phenotypes, trait values are required to be normalized.

- `meanshift_rare_detrimental` (default to `0.0`) – mean shift in quantitative value due to detrimental rare variants.
- `meanshift_rare_protective` (default to `None`) – mean shift in quantitative value due to protective rare variants.
- `meanshiftmax_rare_detrimental` (default to `None`) – maximum mean shift in quantitative value due to detrimental rare variants, applicable to variable effects model.
- `meanshiftmax_rare_protective` (default to `None`) – maximum mean shift in quantitative value due to protective rare variants, applicable to variable effects model.
- `meanshift_common_detrimental` (default to `0.0`) – mean shift in quantitative value due to detrimental common variants, applicable to `rare_only=False`.
- `meanshift_common_protective` (default to `0.0`) mean shift in quantitative value due to protective common variants, applicable to `rare_only=False`.

Note

These options specify the effect of quantitative trait loci in the region of interest, modeled by the shift in mean QT value due to variants. The unit of mean shift is the standard deviation, e.g. `meanshift_rare_detrimental=1.0` means a detrimental mutation increases the mean value of QT by 1σ. When used by itself, all detrimental rare variants will be assigned a fixed effect size as specified. With the `meanshiftmax_rare_detrimental` option, they together model variable effects with `meanshift_rare_detrimental` being the minimum effect size and `meanshiftmax_rare_detrimental` being the maximum effect size for detrimental rare variants. In variable effects model the maximum effect will be assigned to the variant having smallest MAF, and the minimum effect to the one having largest MAF. Values in between are interpolated based on specified max. and min. values. Similarly as above, `meanshift_rare_protective` and `meanshiftmax_rare_protective` are for protective variants, which decrease the mean of QT as opposed to increasing it. `meanshift_common_detrimental` and `meanshift_common_protective` are effects for common detrimental and protective variants respectively. No variable effects model for common variants is available.

Important

In order to efficiently simulate conditional genotypes based on known quantitative values, a fixed effect model with only detrimental rare variants being causal is required (`rare_only=True` and only `meanshift_rare_detrimental` is specified).

- **[genotyping artifact]**

  - `missing_low_maf` (default to `None`) – variant sites having MAF < P are set to be missing.
  - `missing_sites` (default to `None`) – proportion of missing variant sites.
  - `missing_calls` (default to `None`) – proportion of missing genotype calls.
  - `error_calls` (default to `None`) – proportion of error genotype calls.

- **[other]**

  - `max_vars` (default to `3`) – maximum number of causal rare variant sites that each individual may carry on the gene/region of interest.

    Note

    Allowing more than 4 causal rare variant sites per individual is both unrealistic and computationally intractable, as most of rare variant sites are in low allele frequencies.

  - `ascertainment_qualitative` – (optional) ascertaining scheme to obtain conditional genotypes generated under LOGIT variable effects model or `rare_only=False` (represented by a list of non-negative integers where the 1st value denotes required minimum number of affected individuals in the last generation, the 2nd value for minimum number in the second last generation, and so on. E.g. if `(2,0,1)` is specified, it would require the simulated pedigree data to have at least 2 affected offspring in the last generation and 1 affected individual in the grandparental generation.
  - `ascertainment_quantitative` – (optional) ascertaining scheme to obtain conditional genotypes generated under LNR variable effects model or `rare_only=False` (represented by a list of conditions

and each condition applies on corresponded generation in the order as described in `ascertainment_qualitative`. In each condition two elements are required. The 1st one specifies for minimum number of individuals and the 2nd one for value range, where $\sim$ - any value, a$\sim$ - greater than or equal to a, $\sim$b - less than or equal to b, a$\sim$b - within a and b. E.g. `((2, 1.5~), (1, 2~))` requires the simulated data to carry at least 2 offspring in the last generation having trait value $>=$ 1.5 and also 1 individual or more with trait value $>=$ 2 found in the parental generation)

5.2 Rare Variant Sequence Data Simulation Options

Note

> Use provided template `SimulateRareVariant.conf` to update parameter values. Lines start with '[' and end with ']' are comments and will not be parsed by the program.

- `regRange` (default to `[1500, 2500]`)
  - *Gene Length (range)*
  - The length of gene region for each time of evolution is taken as a random number within the range of region.
  - If a fixed number of gene length N is required, this parameter should be set as [N, N].
- `fileName` (default to `MySimuRV`) – *output Files Name (prefix)*
- `numReps` (default to `3`) – *Number of Replicates (genes)*
- `N` (default to `[8100, 8100, 7900, 9000]`)
  - *Effective Population Sizes*
  - Assuming a n (n = array length - 1) stage demographic model, this parameter specifies population sizes at the beginning of evolution and at the end of each stage $N_0, ..., N_n$.
  - If $N_i < N_{i+1}$, an exponential population expansion model will be used to expand population from size N_i to N_{i+1}.
  - If $N_i > N_{i+1}$, an instant population reduction will reduce population size to N_{i+1}.
  - For example, `N=[5000, 5000, 800, 30000]`, which simulates a three stage demographic model where a population beginning with 5000 individuals first undergoes a burn-in stage with constant population size 5000, then goes through a bottleneck of 800 individuals, and after that expands exponentially to a size of 30000.
- `G` (default to `[500, 10, 370]`)
  - *Numbers of Generations per Stage*
  - Numbers of generations of each stage of a n stage demographic model.
  - This parameter should have `n` elements, in comparison to $n + 1$ elements for parameter `N` (Effective Population Sizes).
- `mutationModel` (default to `finite_sites`, choose between `finite_sites` and `infinite_sites`)
  - *Mutation Model*

  - The default mutation model is a finite-site model that allows mutations at any locus. If a mutant is mutated, it will be back-mutated to a wildtype allele.
  - Alternatively, an infinite-sites model can be simulated where new mutants must happen at loci without existing mutants, unless no vacant locus is available (a warning message will be printed in that case).

- `mu` (default to `1.8e-08`) – *Mutation Rate* per base pair

- `revertFixedSites` (default to `False`, boolean parameter, choose between `True` and `False`) – Whether or not to revert fixed mutant sites to wildtype sites

- `selModel` (default to `additive`, choose between `additive` and `multiplicative`) – *Multi-locus selection model*, namely how to obtain an overall individual fitness after obtaining fitness values at all loci

- `selDist` (default to `Boyko_2008_European`, choose from `constant`, `Eyre-Walker_2006`, `Boyko_2008_African`, `Boyko_2008_European`, and `Kyrukov_2009_European`)
  - *Selection Coefficient Distribution Model*
  - Each distribution specifies s (selection coefficient) and h (dominance coefficient, default to 0.5 for additivity) that assign fitness values 1, $1 - hs$ and $1 - s$ for genotypes *AA* (wildtype), *Aa* and ”aa’, respectively.

    > **Note**
    >
    > Positive s is used for negative selection so negative s is needed to specify positive selection.

  - The following distributions are supported
    - `constant`: A single selection coefficient that gives each mutant *a* constant value s. The default parameter for this model is [0.01, 0.5]. You can set `selCoef` to [0, 0] to simulate neutral cases or a negative value for positive selection.
    - `Eyre-Walker_2006`: A basic gamma distribution assuming a constant population size model (Eyre-Walker et. al., 2006). The default parameters for this model is Pr(s=x)=Gamma(0.23, 0.185*2), with h=0.5. A scaling parameter 0.185*2 is used because s in our simulation accounts for 2s in Eyre-Walker et. al.
    - `Boyko_2008_African`: A gamma distribution assuming a two-epoch population size change model for African population (Boyko et. al., 2008). The default parameters for this model is Pr(s=x)=Gamma(0.184, 0.160*2), with h=0.5.
    - `Boyko_2008_European`: A gamma distribution (for *s*) assuming a complex bottleneck model for European population (Boyko et. al., 2008). The default parameters for this model is Pr(s=x)=Gamma(0.206, 0.146*2) with h=0.5.
    - `Kyrukov_2009_European`: A gamma distribution (for *s*) assuming a complex bottleneck model for European population (Kyrukov et. al., 2009). The default parameters for this model is Pr(s=x)=Gamma(0.562341, 0.01) with h=0.5

  > **Note**
  >
  > If you would like to define your own selection model, use the option below `selCoef`.

- `selCoef` (default to `[]`)

- *Customized Selection Coefficient Distribution Model*
- If `[]` is given, the default value of distribution selected in `selDist` will be used.
- Note that length of this parameter determines which type of model to use and values specify distribution coefficients
  - `[]` - no customized input
  - `[s, h]` - constant selection coefficient (s) and dominance coefficient (h)
  - `[k, d, h]` - gamma distributed selection coefficient, where k, d are shape, scale parameters of gamma distribution and h is dominance coefficient
  - `[p, s, k, d, h]` - mixed-gamma distributed selection coefficient, where p is the probability of having the selection coefficient equal to s; k, d are shape and scale parameters of gamma distribution; h is the dominance coefficient
  - `[p,s,k,d,h,l,u]` - truncated mixed gamma, where l,u are lower and upper bounds
  - `[p,s,q,k1,d1,k2,d2,h]` - complex mixed gamma distribution that can generate a mix of constant, positive gamma and negative gamma distributions. The negative distribution represents protective variant sites with negative selection coefficients, where q is the probability of having the selection coefficient following a positive gamma distribution with shape/scale parameters k1/d1. Thus, the probability of having selection coefficient following a negative/opposite gamma distribution is 1-p-q. The negative gamma distribution takes parameters k2 and d2. The positive distribution will be truncated at [0.00001, 0.1], while the negative one at [-0.1, -0.00001]
  - `[p,s,q,k1,d1,k2,d2,h,l1,u1,l2,u2]` – truncated complex mixed gamma distribution, where [l1,u1], [l2,u2] are lower/upper bounds or [k1,d1] and [k2,d2] gamma distributions, respectively
- For example, Parameter [0.001, 0] for a constant model defines a recessive model with fixed s. Recommended parameter for mixed-gamma is [0.37, 0.0, 0.184, 0.160*2, 0.5] for Prob(s=0.0)=0.37 (neutral or synonymous) and Prob(s=x)=(1-0.37)*Gamma(0.184,0.160*2)

- `selRange` (default to `[1e-05, 0.01]`) – Generated selection coefficient is truncated by range limits.

- `recRate` (default to `0`) – *Recombination rate per base pair*, if r times loci distance is greater than 0.5, a rate of 0.5 will be used

- `verbose` (default to `1`) – ”Screen Output Mode (-1, 0, 1), -1 - quiet, no screen output; 0 - minimum output of simulation progress and time spent for each replicate; 1 - regular screen output of statistics, simulation progress and time spent.

- `step` (default to `[100]`) – *Detailed Screen Output Interval per Stage*
  - Calculate and output statistics at intervals of specified number of generations.
  - A single number or a list of numbers for each stage can be specified.
  - If left blank (`[]`), statistics from the beginning to the end of every generation of each stage will be printed out.