

[HOME](#) [MANUAL](#) [DOWNLOAD](#) [FAQ](#) [CONTACT](#)
**About**

- Programs
 - version checking
- Input, Output, Pipe
- Options
- Configure file

Input files

- In general
- Basic input files
 - . Genotypes
 - . Samples and covariates
 - . Pedigrees and probands
 - . Seed genes
 - . Structural variation (SV)
- Other files
 - . Chromosomal regions
 - . Liability classes
 - . LODs scores
 - . Damaging SV
 - . Variant Group File

Programs

- vMAF
- vDEL
- vGrp
- vSEG
- vAAA/vAAA2
- vSVA
- vFIN
- gnGBA
- vMerge

Tutorial

- Variant Classification
- Gene prioritization

1. About**1.1. Programs**
[\[Back to top\]](#)

PERCH is a software suite that includes a number of command-line programs for gene prioritization, variant prioritization, and variant classification:

Programs included in PERCH:

- gnGBA -- evaluates the biological relevance of each gene to the disease of interest (BayesGBA)
- vMAF -- calculates the allele frequency that is the closest to 0.5 across populations (MaxAF)
- vDEL -- calculates a combined deleteriousness score (BayesDel)
- vSEG -- performs co-segregation analysis (BayesSeg)
- vAAA -- performs rare-variant association analysis (BayesHLR)
- vSVA -- performs structural variation analysis (BayesHLR)
- vFIN -- combines all components (BayesSeg + BayesHLR + BayesGBA + others)
- vGrp -- prepare a [Variant Group File](#) for gene set analysis
- vMerge -- merge multiple gene lists for candidate gene selection

It is important to protect the privacy of participants in a research. Therefore, this software suite can run in your local computer. No information will be collected, stored, or sent out from your computer. Once installed, it does not retrieve any data from the Internet except for version checking.

1.1.1 Version Checking

The version of this software package is represented by a version number and a build date. If it is a beta version, the version number is followed by the word "beta". All programs have the same version number. Program version and data version are separated; sometimes you only need to upgrade the programs but not the data. The (--version) option of any program will check for new versions for both programs and data through the Internet. It will not send out any information.

1.2. Input, output and Unix pipe
[\[Back to top\]](#)

Most programs in this package read a [Genotype File](#) (see below for details about this file), do analysis, and then write a modified [Genotype File](#) that will be further analyzed and modified by downstream programs. At the end, the program vFIN reads the final version of the [Genotype File](#) and calculates an overall score for each gene or variant. Therefore, all these programs are linked by the [Genotype File](#).

The programs read a [Genotype File](#) from the standard input, analyze and modify it, then write to the standard output. Therefore, you can use the Unix pipe ("|") to run multiple programs sequentially. Most programs also accept a [Genotype File](#) filename in the command line (any string that is not recognized as a program option or argument), so that they provide the flexibility as to where to start a chain of analysis. At the end of the chain, you can redirect the standard output to a file ("| gzip -c > results.gz") or request the program to directly write

to a file (-o results.gz). Unix pipe can save time and disk space. However, if it is the first time you run these programs, it is better to run one program each time and redirect the intermediate output to a file. That way it is easier to locate a problem if there's any.

1.3. Options

[\[Back to top\]](#)

This User Manual tells the purpose of each program option. For more details about their usage, such as the number of arguments, data types, and default values, please use the (-h) or (--help) option to print the program help text.

In general, the usage of program options follows these rules:

- 1) Options with 2+ arguments must be used like this: "--option arg1 arg2";
- 2) Options with only 1 argument can be used like this: "--option=arg" or "--option arg";
- 3) You can omit a boolean argument that is positive, ie: "--option=yes" is equal to "--option";
- 4) It is better to always use "--option=arg" whenever possible, which is more readable;
- 5) If the argument is an array, you can replace the default by "--array-option=v1,v2";
- 6) You can clear the default by assignment to an empty array "--array-option=";
- 7) You can insert or delete items by "--array-option+=v1,v2" or "--array-option-=v1,v2", respectively;
- 8) The order of different options does not matter. For example, "-a -b" is equivalent to "-b -a";
- 9) The order of the same option matters. The last one overrides the others. E.g., "-a=yes -a=no" is "-a=no";
- 10) You cannot merge multiple single-letter options into one: "-a -b" cannot be written as "-ab".

Be careful about array-type arguments: the correct way to assign two values is "--array-option=v1,v2" but not "--array-option=v1 --array-option=v2". The latter will first assign v1 to the array, then assign v2 and get rid of v1, leaving only one element (v2) in the array.

All escape sequences in program options and arguments will be replaced. The following sequences are recognized: \ (backslash), \a (alert), \b (backspace), \f (form feed), \n (new line), \r (carriage return), \t (horizontal tab), \v (vertical tab), and \xHH (byte with 2-digits hexadecimal value).

The (-h) or (--help) option will print a short help text for the program. In the help text, the description of each program option follows the convention of "ProgramOption ArgumentDataType Function {CurrentValue}". Anything within a pair of square brackets is optional. Things within curly brackets are the current value after setting by factory defaults, configure files, and program options. To make the help text more readable, please set the terminal window size to at least 160 columns.

ArgumentDataType is represented by a letter or a string as shown below:

- D or DBL -- a floating point number
- I or INT -- an integer without commas for thousands (1234 but not 1,234)
- B or BOOL -- a case-insensitive boolean value (0/1/no/yes/n/y/false/true/f/t)
- C or CHAR -- a character
- S or STR -- a string, escape sequence allowed (e.g., "hi\n" = hi\n = \$'hi\n')
- F or FILE -- a filename, may include relative or absolute path, environment variables allowed
- P or PATH -- a path, relative or absolute, environment variables allowed, better ends with /
- FLD or FIELD -- a field number. The first field is 1, and so on. The last field is -1, second last is -2, and so on.
- FLDs or FIELDS -- an array of field numbers divided by a comma
- Ss or STRs -- an array of strings separated by a comma
- Fs or FILEs -- an array of filenames separated by a colon

If specified, there may be a strict requirement for the format and the number of arguments. For example, "D,D,D" refers to exactly 3 floating point numbers separated by a comma, such as the --penetrance arguments. ArgumentDataType could be followed by an integer to help the writing of the descriptions, such as S1,S2,S3 for the --xct-pfx option, where S1 is ExAC file prefix, S2 is cases' qc.log file prefix, S3 is cases' coverage file prefix.

1.4. Configure file

[\[Back to top\]](#)

A Configure File is another way to pass parameters to programs besides program options. Although being optional, using a Configure File is more convenient than program options, especially for the parameters that need to be the same for all programs. The programs read a Configure File at `./par.txt`, which should be used to specify parameters for the current analysis. Command line options override configure file options.

In this file, one row is one option. The format is "`--option=argument`". Space is not allowed in the arguments. Anything after the argument will be omitted. Multiple consecutive spaces or tabs are treated as one delimiter. Lines starting with a `#` are comments. Comments and blank lines will be omitted. The first row of the file has a special content (see the examples below), so that the programs will not accidentally read a wrong file.

Not all program options can be set in a Configure File. Below is a list of supported options:

```
--col-one Ss      Header of the first column {"#CHROM", "#Chrom", "CHROM", "Chrom", "#CHR", "#Chr", "#chr", "Chr", "CHR", "chr"}
--col-func S      Header of the column for functional consequence {Func_Type}
--col-gene S      Header of the column for gene symbol {Func_Gene}
--col-fdet S      Header of the column for functional details {Func_Detail}
--info-func S      The name of the INFO sub-field for functional consequence annotation {vAnnGene}
--gnuplot S       The command for gnuplot 4.6.0 or later {gnuplot}
--gdb S          The gene database (refGeneLite/refGeneFull/ensGeneLite/ensGeneFull) {refGeneLite}
--filt-vqs-nan B   Exclude variants if VQSLOD is missing {no}
--filt-vqs-snv D   Exclude variants if VQSLOD<D (-inf = no filtering) for SNVs {-5.368}
--filt-vqs-indel D Exclude variants if VQSLOD<D (-inf = no filtering) for Indels {-4.208}
--filt-miss-rate D Exclude variants if missing rate in cases or controls is > D (1 = no filtering) {0.01}
--filt-filter Ss   Exclude variants if FILTER is not one of the Ss {.,PASS}
--filt-QD D        Exclude variants if QD (Qual. By Depth.) < D (0 = no filtering, GATK default = 2) {0}
--filt-MQ D        Exclude variants if MQ (Mapping Quality) < D (0 = no filtering, GATK default = 40) {0}
--filt-FS D        Exclude variants if FS (Fisher Strand) > D (0 = no filtering, GATK default = 60) {0}
--filt-HS D        Exclude variants if HaplotypeScore > D (0 = no filtering, GATK default = 13) {0}
--filt-MR D        Exclude variants if MQRankSum < D (0 = no filtering, GATK default = -12.5) {0}
--filt-RP D        Exclude variants if ReadPosRankSum < D (0 = no filtering, GATK default = -8) {0}
--hard-filter B    Exclude variants by hard filters. Will set thresholds with GATK defaults if not set. {No}
--HardFilterIfNoVQ B Exclude variants by hard filters if there's no VQSLOD. Will set thresholds with GATK defaults. {yes}
--filt-DP I        Exclude genotypes if DP<I (0 = no filter) {10}
--filt-GQ I        Exclude genotypes if GQ<I (0 = no filter) {40}
--filt-GP D        Exclude genotypes if GP<D (0 = no filter) {0.8}
--filt-domGP B     Exclude genotypes by GP with a dominant model {false}
--filt-recGP B     Exclude genotypes by GP with a recessive model {false}
--filt-MaxAF D     Exclude variants if MaxAF > D (0 = no filter) {0.01}
--filt-SplAF D     Exclude variants if SplAF > D (0 = no filter) {0}
--filt-FdrAF D     Exclude variants if FdrAF > D (0 = no filter) {0.05}
--filt-del D       Exclude variants if BayesDel<D (-inf = no filtering) {-0.0592577}
--no-miss B        Treat missing genotype as homozygous REF allele {false}
--prevalence D     Prevalence {0.025}
--penetrance D,D,D Penetrance {0.02,0.1,0.5}
--include Fs       Analysis is restricted to regions in FILEs. Use 2+ files to define intersection. {}
--exclude Fs       Analysis is restricted to regions not in FILEs. Use 2+ files to define union. {}
--lof-only B       Analysis is restricted to loss-of-function (LoF) variants only, BayesDel still applies. {no}
--lof-no-del B     Analysis is restricted to loss-of-function (LoF) variants only, BayesDel not considered {no}
--lof-tol F        Analysis exclude LoF-tolerated genes in file F when --lof-only or --lof-no-del is yes {panel_LoFtol}
--no-mhc B         Analysis is restricted to non-MHC regions {}
--mhc-only B       Analysis is restricted to MHC regions {}
--rm-ind Fs        Remove individuals listed in file(s) Fs {}
--vc B            Run mode is Variant Classification {no}
--OUT S           Output prefix is S {""}
--no-web B        Do not check version from web {false}
```

Below is an example of the `./par.txt`:

```
VICTOR_parameters_1.0 << Do not delete or modify. This line prevents reading a wrong file.
--prevalence=0.025 << prevalence
--penetrance=0.02,0.1,0.5 << penetrance
```

2. Input Files

2.1. In general

[Back to top]

Unless otherwise stated in this Manual, input files for this software have columns divided by a space or a tab. Multiple successive delimiters are not treated as one. Lines starting with a ``#'` are comments and will be

ignored. The reading of input files is robust to Linux/Mac/Windows/mixed line breaks. It is also robust to the missing of a line break after the last line. Programs will stop reading a file at the first blank line. The input file does not need to be uncompressed if it is a .gz or .bz2 because the programs can read them directly. You don't even need to type ".gz" or ".bz2" in the command, as the programs will first look for the uncompressed file, then file.gz, followed by file.bz2. Below are the descriptions of each input file. You can also search for "[XXXXX File](#)" in this manual to find all related information about each file type.

2.2. Basic input files

[Back to top]

A **Genotype File** for PERCH is an annotated VCF file that fulfills the following requirements:

- 1) Variants with multiple alternative alleles are split into separate lines;
- 2) All variants are left-normalized after splitting alternative alleles;
- 3) It has deleteriousness score annotations, either in separate columns or in the INFO field;
- 4) It has a column named Func_Type for the most severe functional consequence of each variant;
- 5) It has a column named Func_Gene for the gene symbol associated with the above consequence;
- 6) Variants with functional consequences in multiple genes are split into separate lines;
- 7) Loss-of-function variants have a label "LoF" in the Func_Type column, e.g. LoF_StopGain;
- 8) Lines are sorted by #CHROM and Func_Gene.

The Genotype File is not required to be bgzip compressed and tabix indexed. It does not have to contain all chromosomes, so it's possible to run different chromosomes in multiple computer nodes simultaneously. It does need to contain all samples for analysis. However, if your study involves multiple cohorts from different populations, you can make one VCF file for each population and do the analysis separately.

Sample File

A Sample File contains ID, sex, outcome and covariates for all samples, including unrelated cases and controls for association tests, duplicated samples for quality control, individuals from pedigrees for linkage analysis or de novo mutation detection, and any other samples that have been sequenced and could be used for quality control. This file has a header row. The first column should be named "SeqID", which contains the Sequence ID for each sequenced sample in a Genotype File, i.e., the ID in a VCF header row. The second column is "Sex" with contents of Female or Male or UnknSex; other strings represent unknown sex. If you don't know the sex of some samples, set them to unknown; **don't set unknown sex as males or females because this will lead to wrong allele frequency calculations for chromosome X or Y**. The third column is the outcome variable, which can be an affection status (Unaffected or Affected or UnknAff; other strings are unknown affection status) or a quantitative trait (if there is any numbers not equal to 0/1/2). Quantitative trait values must be greater than 0. Sex and Affection Status are case-insensitive. Samples not to be used in a case-control association test should have an unknown affection status. They are in this file just to provide the sex information. Columns starting from the fourth are optional. They are covariates for association tests and quality control. Missing values can be . or unknown. Covariates can be string-type or numeric-type but not mixed, i.e., a covariate cannot contain both numbers and non-missing strings. The name of a covariate cannot start with "victorPC" or "victorPop", which are reserved by VICTOR to store the principle component analysis results. If you want to provide the population origin information instead of performing a PCA, you can add a string-type covariate with the header "Pop". A Sample File may contain more or fewer people than the Genotype File. Only the overlapping samples will be analyzed. In an association test, samples with a missing value for the outcome or any covariate will be omitted. Lines in this file need not be sorted nor in the same order as the Genotype File. Below is an example, where the columns are separated by multiple spaces for web display purpose only. In a real file they should be divided by one tab.

SeqID	Sex	Disease	Pop
ped1_i1	Male	UnknAff	NFE
ped1_i2	Female	UnknAff	NFE
ped1_i3	Male	Affected	NFE
ped1_i4	Female	UnknAff	NFE
random	UnknSex	UnknAff	SAS
HG00096	Male	Unaffected	NFE
HG00097	Female	Unaffected	NFE

HG00099 Female Unaffected NFE
 HG00100 Female Unaffected NFE

Pedigree File

A Pedigree File contains all pedigrees for linkage analysis by vSEG or de novo mutation analysis by vAAA. This file is similar to PLINK .fam, but is more robust, flexible, and informative. If you already have a PLINK .fam file you can directly use it as a Pedigree File. It has 6 columns corresponding to PedigreeID, IndividualID, FatherID, MotherID, Sex, and AffectionStatus. It has a header line as shown in the example below. Optionally, it can also have the 7th column for LiabilityClass. Pedigrees should not have any consanguinity loop or marriage loop. If your pedigree has loops, you can manually break the loops by founderizing a minimum number of individuals while maintaining the relationships among affected individuals as much as possible. My program "[pedpro](#)" has the function to do so automatically. You can use that program to break the loops before analysis by vSEG. There is no other restriction on pedigree structure. For variant classification, Pedigree File should indicate who is the proband in each pedigree, which can be done by adding the phrase "(proband)" at the end of the IndividualID. Beware that the word "proband" may mean different concepts to different people -- to a genetic counselor it may be the first person requested for a genetic test, who may or may not be a carrier of the mutation; while to a statistician it is the first person in the pedigree who tested positive for the mutation. If you don't know how to set a proband, then just leave it to the program; vSEG will automatically choose a proband in a conservative way. You can use the traditional coding of Sex (1 for male, 2 female, 0 for unknown) and AffectionStatus (1 for unaffected, 2 for affected, 0 for unknown) or use the same coding as in a Sample File. Liability Class is an integer starting from 1. The IndividualID should match with the header of the VCF file. Below is an example. Here columns are separated by multiple spaces for web display purpose. In a real file they should be divided by exactly one tab.

PedID	IndID	Father	Mother	Sex	Aff
ped1	ped1_i1	0	0	1	2
ped1	ped1_i2	0	0	2	1
ped1	ped1_i3	ped1_i1	ped1_i2	1	2
ped1	ped1_i4	ped1_i1	ped1_i2	2	2

Sometimes it is inconvenient to manually create a pedigree file with IndividualIDs matching with the header of a VCF file. [VICTOR](#) provides a program named vConvertPed to help you create this file from another format. Please see the vConvertPed section below.

Sample Weight File

A Sample Weight File contains weights for each individual to be used in a regression analysis to control for relationship among pedigree members. This file has two columns: SeqID and Weight.

Seed File

A Seed File lists the seed genes for guilt-by-association analysis by gnGBA. The first column of this file is a disease name and the second column gene symbols. Space is not allowed in the disease name or the gene symbol. Make sure that gene symbols are NCBI Official Symbols, not full names or synonyms or GeneIDs. This file does not have a header row. Below is an example:

Disease Gene1
 Disease Gene2

Structural Variation File

A Structural Variation File contains genotypes of structural variations. It is in one of the 6 formats: 1) the original VCF format by the 1000 Genome Project; 2) XHMM output format, old or new version; 3) WHAM output format; 4) CLAMMS output format; 5) PennCNV output format; and 6) long format. Among these, the first 3 format is called the VCF-like format, they could be automatically detected by vSVA. Long format is a 10-column text file with a header line. The columns are sv_calling_program, sample_ID, affection_status, chr, start, end,

sv_type, kb, quality, and read_count. The vSVA program will read the sample_ID, chr, start, end, sv_type, and quality column.

2.3. Other input files or intermediate result files

[\[Back to top\]](#)

A **Region File** defines chromosomal regions to be included in or excluded from an analysis. It is tab-delimited (space is not a delimiter to separate columns). This file may be a BED file, from which only the first 3 columns (Chr, Start, and End) will be read. The Start position is 0-based, while End is 1-based. There should be only one track in a BED file. Track title lines that start with "track name=" will be ignored. Alternatively, if the file contains only 1 column, the content should be Chr:Start-End, where both Start and End are 1-based. If the file has exactly 2 columns, then the first column is Chr and the second is Start, assuming Start equals to End, which are both 1-based.

A **Liability Class File** contains penetrance for all liability classes that will be used in a linkage analysis. The first non-comment data line of this file is the number of liability classes. Starting from the second line are penetrance for each liability class, which are three numbers separated by a space, for 0, 1 and 2 copies of a risk allele, respectively. All these together is deemed a liability class model for autosomal chromosomes. After this, there should be a blank line and then another liability class model for chromosome X.

A **Score File** contains log10 Bayes factor scores for all genes, which could be biological relevance scores calculated by the gnGBA program or other scores on the same scale, such as LOD scores from a previous linkage scan, LOD scores calculated by SEQLinkage, structural variant analysis results output from the vSVA program, or Bayes factors calculated from previous expression profiling studies. In this file, the first row contains gene symbols; the second row are log10 Bayes factors. Alternatively, this file can also be transposed: column 1 are gene symbols, and column 2 scores. This file may have a column header row. Below are two examples:

example 1 (the first column will be ignored):

```
Trait\Gene ATAD3A HLA-A C1orf53 ..
Disease1 -0.002 1.3990 0.8266 ..
```

example 2 (the first row will be ignored):

```
Symbol Disease1
ATAD3A -0.002
A3GALT2 1.3990
C1orf53 0.8266
..
```

A **Damaging Structural Variation Log File** is output from the vSVA program by the (--detail) option. This file contains seven columns: #Chrom, StargePos, EndPos, VariantType, GeneSymbol, SeqID, CopyNumber. Header line is optional. This file can be read by vAAA for an joint analysis with SNVs and InDels.

A **Variant Group File** is a file created by vGrp. It contains variant information for group-wise analyses by vAAA2. A group of variant can be those within a protein domain, a gene, or a gene set. This file does not contain genotypes, but should have an INFO column that contains the necessary annotations for variant filtering, such as BayesDel score and MaxAF. The file should have the first 5 columns of a VCF file and a group column. Lines should be sorted by the group value.

3. Programs

3.1. vMAF

[\[Back to top\]](#)

vMAF identifies allele frequency annotations in a **Genotype File** (--col-af and --pop), calculates the allele frequency that is the closest to 0.5 across populations (MaxAF), and appends a new column of MaxAF to the output **Genotype File**. Allele frequency annotations can be in the INFO field or in separate columns. Therefore, this program can directly read G1K, ExAC, gnomAD or similar VCF files containing allele frequency annotations,

and/or files annotated by ANNOVAR.

3.2. vDEL [\[Back to top\]](#)

vDEL calculates a combined deleteriousness score name BayesDel from a set of individual scores. This program has built-in weight functions for different sets of component scores (--ensemble). The input Genotype File should have been annotated with the deleteriousness scores that match your choice of ensemble. These scores should be in separate columns with the expected header names, or in the INFO column with the expected field names. Please read the help text for a list of the names. The default ensemble in PERCH v1.2 is dbNSFP v3.3a plus MaxAF. Accordingly, the BayesDel integrates FATHMM, GERP++_RS, LRT, MutationAssessor, MutationTaster, Polyphen2_HDIV, Polyphen2_HVAR, PROVEAN, SIFT, SiPhy_29way, VEST3, fitCons, fathmm-MKL_coding, phastCons100way, phastCons20way, phyloP100way, phyloP20way, and MaxAF.

vDEL not only work for missense variants. It calculate BayesDel for in-frame InDels from PROVEAN, and for non-coding SNVs and InDels from the combination of fathmm-MKL_noncoding, CADD and MaxAF. For loss-of-function variants (stop-gain, splice-altering, frame-shifting except those in the last 5% of a coding sequence), BayesDel is equal to the highest score among all possible missense variants within the gene (--wst-for). Therefore, the input Genotype File should have been annotated functional consequences (--col-func), where loss-of-function variants should be labeled "LoF", for example, StopGain(LoF).

3.3. vGrp [\[Back to top\]](#)

This program reads annotated VCF files for all chromosomes, and writes a Variant Group File for group-wise analysis by vAAA2. A group of variant can be those within a protein domain, a gene, or a gene set.

For gene set analysis, it also reads a gene set file in GMT format (--gene-set) and the output file from vAAA gene-wise --collapse analysis (--collapse). The genes that have no observed variants in the previous --collapse analysis will removed from a gene set, thus reducing the original gene set size to an effective gene set size. The output gene sets are restricted by the effective gene set size (--min-set, --max-set); gene sets that are too big or too small will be discarded. Gene sets can also be restricted by the biological function of a gene set that is indicated by the guilt-by-association (GBA) score of each gene in the gene set. You need to input a GBA score file (--invo-gba-file) and a GBA score cutoff (--gba-cut). vGrp will output only the gene sets that have a gene with a GBA score greater than or equal to the cutoff value. Alternatively, if you use the option (--incl-gba-file), vGrp will output only the gene sets that *all* genes in the gene set have a GBA score greater than or equal to the cutoff value. It is recommended to use the (--invo-gba-file) option with the default (--gba-cut) argument. This strategy will decrease the total number of gene sets, reducing multiple testing penalty, while still maintains sufficient gene sets for hypothesis-generating research.

3.4. vSEG/vSEG2 [\[Back to top\]](#)

vSEG performs group-wise (--group) or variant-wise (--detail, --vc) co-segregation analysis. It reads two files: a Pedigree File (--ped) describing pedigree structures and phenotypes and a Genotype File containing genotypes of all sequenced samples. If the Pedigree File contains a column of Liability Classes, then a Liability Class File (--liab) is mandatory. vSEG requires that Genotype File already has the MaxAF annotation, so that the program can use it in the linkage analysis. If MaxAF is 0, a default allele frequency will be used (--default-maf). By default, variants are grouped by values of the user-defined columns (--group), which may contain gene symbols and/or protein domain names. The output of this program is a modified Genotype File with an appended column of group-wise LOD scores. In this column, variants with Mendelian errors will show a "nan" instead of a LOD score, while other variants in the same gene will still have a LOD score. The vAAA program reads the vSEG output and excludes "nan" variants from subsequent analysis. Thus, vSEG also serves as a quality control method. The option (--detail) shows the details for each variant in an analysis. It appends a new column of single-variant-based analysis results, which are LOD scores or the filters that the variant didn't pass.

Penetrance

Using a correct penetrance parameter is vital for linkage analysis. Considerations in penetrance should include inheritance model (dominant / recessive / additive), age, sex, pleiotropy, race, birth year, and other risk factors for the disease of interest such as environmental exposures and pathogenic variants in known disease genes. Each distinct penetrance is called a liability class. If you need only one penetrance for all individuals (i.e., no liability class), then you only need to set the penetrance option (`--penetrance`); if you need different penetrance for different group of individuals, then the Pedigree File should contain a liability class column and a Liability Class File (`--liab`) should be provided.

vSEG2

vSEG2 is the next version of vSEG. I will gradually modify all my scripts to use vSEG2 only, and discard vSEG in the future. For now I still keep the vSEG in the package for a while. vSEG2 is different from vSEG in that it reads a Variant Group File and VCF Files (`--gtp`) instead of just a Genotype File. Because of this, the VCF File needs not be annotated and sorted by gene symbol, thus it can be the tabix-indexed VCF file right after quality control. The Variant Group File is the default input file. The VCF Files can a file that contains all chromosomes, or a set of files separated by chromosomes or chromosome arms. For the former, you can set the (`--gtp`) argument to the actual VCF File name. For the latter, the (`--gtp`) argument is a filename template. For example, the filename template can be `PERCH.chr@.qc.vcf.gz`. vSEG2 will replace the `@` character with a chromosome code (1-22,X,Y,M) for each variant and look for the corresponding VCF File. Because vSEG2 needs to know the VCF File format before it reads the Variant Group File, it requires an example (`--example-chr`) if the (`--gtp`) argument is a filename template.

3.5. vAAA/vAAA2

[\[Back to top\]](#)

Input files

This program performs a variety of rare-variant association analyses. This program reads two files. The first one is a Sample File (`--spl`). It has three or more columns, corresponding to SeqID, Sex, Outcome, and covariates, respectively. All covariates will be included in the association analysis, so don't list any extra columns in the file. Samples with a missing value for the outcome variable or any covariate variable will be omitted. But samples with a missing value for sex will still be used for the analysis of autosomal chromosomes.

The second file is Genotype File. This file should have been analyzed by vDEL and should contain the INFO column from the original VCF file. This INFO will be used to calculate a weight for each variant. In the analysis, variants are grouped by values of the user-defined columns (`--group`), which may contain protein domain IDs or gene symbols. Therefore, it can perform domain-based, gene-based, or region-based association tests at user's discretion. The rows of the input file should be sorted by the grouping columns.

It would be nice to jointly consider structural variations and SNVs and InDels. To do so, you need to specify a Damaging Structural Variation Log File (`--sv`) and an exclusion criteria (`--filt-sv`).

For de novo mutation analysis (`--de-novo`), it reads a pedigree file (`--ped`) and finds the trios that all three persons were sequenced, and neither one of the parents was affected. The child in a trio will be used as cases or controls depending on the affection status. Optionally, you can also restrict the analysis to de novo mutations only (`--dn-only`). Please see the Tutorial for more details.

If you use multiple affected individuals from a family in a linear regression analysis instead of one index case per family, an Individual Weight File is required (`--weight`). This file has two columns: sequence ID and individual weight. Weight can be calculated from a kinship coefficient matrix or by simulation. Weight values should be between zero (non-inclusive) and one (inclusive). However, when multiple cases are used in an association test, you cannot do co-segregation analysis by vSEG because otherwise the same information would be counted twice. Alternatively, you can select an index case per family based on age of onset, severity of disease, and family history of disease.

Filters

The options (--include) and (--exclude) read chromosomal regions for inclusion and exclusion from a [Region File](#). These options can be instantiated for an unlimited number of times. Analysis can also be restricted to SNVs (--snv-only) or loss-of-function variants (--lof-only). Sometimes it is desired to include only rare or low-frequency variants in an analysis. For this purpose, this program allows the setting of a minor allele frequency cutoff and filter variants by the MaxAF (--filt-MaxAF). This should be valid if the population in MaxAF is unselected (i.e., they are not healthy controls) or the disease of interest is rare. This program can also filter variants by the observed allele frequency among founders (--filt-FdrAF). Because this filter is sensitive to small sample size, it will be applied only when the sample size is large enough. The program calculates the minimum sample size so that the probability of observing an MAF greater than the --filt-FdrAF cutoff given the --filt-MaxAF cutoff is low (--SpISzP), therefore it needs that the --filt-FdrAF cutoff is higher than the --filt-MaxAF cutoff. It is noteworthy that the observed allele frequency is calculated from the combined samples of cases and controls and other samples with unknown affection status, which will not inflate the type-I-error rate, as has been proven previously (Nature Genetics, Vol.43 No.5).

Variant weight

Variant weight includes deleteriousness and variant call quality that is available in the INFO column of the [Genotype File](#). If you provide a [Score File](#) (--biol), the weight function will also include biological relevance scores. Whether to weight a variant in an analysis is an option (--var-wt). Users can also choose to remove only deleteriousness (--wt-del) or call quality (--wt-vqs) from the weight function.

Analysis

- 1) The default analysis method is Fisher's exact test (--FET). It does allow for categorical covariates. The program will divide the samples into groups based on the covariates, perform tests in each group, then combine the p-values by Fisher's method weighted by the effective sample size. It works for small sample size. That's why it is the default. If you have a moderate to large sample size and may have quantitative covariates, please use the regression methods.
- 2) Firth's penalized linear regression with collapsed genotype scores (--linear). It allows categorical or quantitative covariates. This method accepts multiple affected individuals from a family, in which case an [Individual Weight File](#) is required (--weight). Therefore, with this method you don't have to choose one index case per pedigree; you can use all of them. But you also do linkage analysis and want to combine the results, then you still have to choose one index case per pedigree for the association analysis.
- 3) Firth's penalized logistic regression with collapsed genotype scores (--logistic). It allows categorical or quantitative covariates. This method accepts multiple affected individuals from a family, in which case an [Individual Weight File](#) is required (--weight). Therefore, with this method you don't have to choose one index case per pedigree; you can use all of them. But if you also do linkage analysis and want to combine the results, then you still have to choose one index case per pedigree for the association analysis.
- 4) Haplotype Likelihood Ratio score (--HLR), which requires a penetrance model like that in a linkage analysis (--pen).
- 5) The sum of weighted squared score test (--SSUw). It weights variants inversely by the sample standard deviation of the genotypes.
- 6) Variant classification (--vc), which performs a single variant test and doesn't weight variant by deleteriousness score.
- 7) User provided R script (--my-R). Data is read into a data frame named 'x' by the program. The R script needs to do analysis and write the p-value to MyR_pval. Other information should be written to MyR_info. Use "OUTCOME_VARIABLE" to represent outcome variable, and "+COVARIATE_NAMES" for covariates in a regression analysis. The main predictor is "gene". Below is an example script:

```
# Firth's penalized logistic regression
options(warn=-1)
suppressMessages(library(logistf))
res <- logistf(OUTCOME_VARIABLE~gene+COVARIATE_NAMES, data=x, weights=weight)
cat("MyR_pval ", res$prob[2], "\n", sep = "")
cat("MyR_info OR=", exp(res$coefficients[2]), "[",
exp(res$ci.lower[2]), ", ", exp(res$ci.upper[2]), "]", "\n", sep = "")
```

Output

By default the program will add a column to the input Genotype File. The content is Bayes factor in log10 scale, which will be used for subsequent integration with co-segregation analysis and biological knowledge of genes. You can also request to output a -log10 p-value (--out-mlp) when the analysis is --linear or --SSUw.

After the analysis, you may want to look at the details of each gene and see what variants are driving the high scores of the top genes. The option (--collapse) shows the collapsed genotype of each gene. It writes one line per gene, showing the number of samples with no, at least one heterozygous, and at last one homozygous variant in cases and controls, respectively. Note that this is an over-simplified view of the data and may not reflect the true data in an analysis, as the probability of damaging is not shown. The option (--detail) shows the details of each variant in an analysis. It will append a new column named vAAA_details to the Genotype File. For the variants that were included in the analysis, the content of this column is the number of samples with genotype CC,RC,RR (R=rare, C=common) in cases and controls, respectively. If the option (--show-id) is used, the content of this column will be the IDs of samples with an RC or CC genotype. For the variants that were excluded from the analysis, the content is the reason they were excluded.

vAAA2

vAAA2 is the next version of vAAA. I will gradually modify all my scripts to use vAAA2 only, and discard vAAA in the future. For now I still keep the vAAA in the package for a while. vAAA2 is different from vAAA in that it reads a Variant Group File and VCF Files (--gtp) instead of just a Genotype File. Because of this, the VCF File needs not be annotated and sorted by gene symbol, thus it can be the tabix-indexed VCF file right after quality control. The Variant Group File is the default input file. The VCF Files can a file that contains all chromosomes, or a set of files separated by chromosomes or chromosome arms. For the former, you can set the (--gtp) argument to the actual VCF File name. For the latter, the (--gtp) argument is a filename template. For example, the filename template can be PERCH.chr@.qc.vcf.gz. vAAA2 will replace the @ character with a chromosome code (1-22,X,Y,M) for each variant and look for the corresponding VCF File. Because vAAA2 needs to know the VCF File format before it reads the Variant Group File, it requires an example (--example-chr) if the (--gtp) argument is a filename template.

3.6. vSVA

[\[Back to top\]](#)

vSVA analyzes structural variations (SV), which include large insertions, inversions, deletions, duplications, and copy number variations. Like vAAA, this program reads a Sample File (--spl) and a Structural Variation File. Unlike vAAA, it does functional consequence annotations internally, so the Structural Variation File is a raw VCF file without any annotations. By default the Structural Variation File is in VCF-like format (VCF,XHMM,WHAM), which can be automatically detected by vSVA. If the file is in PennCNV or CLAMMs or long format, please use the (--penn) or (--clamms) or (--long) option to specify.

An inversion damages a gene if it covers any exon or flanking regions but not the entire gene including flanking regions. A deletion or insertion is damaging if it overlaps any exon or flanking regions. A duplication is not damaging if it covers the whole gene. The upstream flanking region is defined as 1 to 35 basepairs upstream of the transcription start site (--up). Downstream flanking region is 0 by default (--dn). Intronic padding is 12 basepairs (--intron). These assignments are rather conservative. You can record the damaged genes in a Damaging Structural Variant Log File (--detail), then jointly analyze structural variations with SNVs and Indels by vAAA (--sv option of vAAA). You can also test for association with the structural variants only (--test). This is necessary as vAAA will not analyze a gene if there's no SNV or InDel observed in the gene. So it is better to perform two analyses: 1. joint analysis of small variants and structural variations by vAAA; and 2. analysis of structural variations by vSVA. Then see whether some genes are analyzed by vSVA but not vAAA.

Like SNVs and InDel, it is highly recommended to jointly call SVs for all samples together. Please note that the default values for DP (--filt-DP) and GQ (--filt-GQ) is different from those for small variants.

3.7. vFIN

[Back to top]

Inputs

vFIN reads a Genotype File, sums up co-segregation and association scores, then writes the final scores to the standard output. It uses a group column to identify variant groups (--group), which could be gene symbols, protein domains, or gene sets. It identifies score columns by the header line. You can omit some of the scores by the options (--add-seg=no) or (--add-hlr=no). To include biological relevance, you need to specify a Score File (--biol). To include other scores, you need to specify other Score Files (--lbf). The differences between (--biol) and (--lbf) are:

1. Biological relevance assessment is gene-based, so the IDs in a (--biol) file are always gene symbols. Whereas IDs in a (--lbf) file are required to match with the group columns in vAAA and vSEG analyses. For example, if you did a domain-based analysis, then the IDs in a (--lbf) file should be something like "geneA:domain1". It contains both a gene symbol and a domain name, separated by a colon.
2. Because of the matching, (--lbf) scores can be directly added with those from vAAA and vSEG, whereas for (--biol) it may be less straightforward. For domain-based or gene-based analysis, (--biol) scores are directly added. For pathway-based analysis, the biological relevance of a pathway is the highest GBA score among all the genes within the pathway.
3. Negative (--biol) scores are converted to zeros by default, whereas (--lbf) scores are not. This special treatment for (--biol) is based on the idea that we know only very little about the genetics of human diseases and about gene networks. Therefore, genes that are in strong connections with known disease genes are certainly interesting candidates, but a weak or no connection does not necessarily mean that the gene is less likely a disease gene. Hence, it may make sense not to down-score the latter. Nevertheless, this feature can be turned off by (--neg-biol).
4. When a gene is not involved in the biological relevance assessment due to the incompleteness of a gene network, it is possible to assign an estimated relevance score if the gene is part of a group of duplicated genes (--dgd).

Outputs

The output of vFIN has these columns from left to right: group, cytoband, basepair position of the middle point of the observed variants in the group, overall score, and then the component scores. This output can also be the input file for vFIN again.

Given the hypothesis of the disease, you can remove non-protein coding genes (--no-ncRNA) or the genes with high mutation tolerance (--no-rvis) from the output.

vFIN can draw a Manhattan plot (--plot-to) by calling the GnuPlot program version 4.6 or later. If you have performed rare-variant association tests (Fisher's exact test, logistic or linear regression, SSU, permutation), vFIN will plot the -log₁₀ p-values. Otherwise, it will plot the log₁₀ Bayes factors. It will automatically detect what you have done from the header row of the input file, so no need for any program options. This plot is robust to the grouping of variants in the analyses, i.e., no matter whether the analysis is variant-based, domain-based, gene-based, region-based or even pathway-based, each data point in the plot is always the maximum score of a gene. vFIN calls the GnuPlot program to plot to a file. Currently, only PDF format is supported. Thus, the argument for (--plot-to) must end with ".pdf". If the required version of GnuPlot is not in the default \$PATH, the option (--gnuplot) can be used to set the correct path to GnuPlot.

Gene set analysis

If you do a gene set analysis, vFIN has some special treatments in the output. If you specify pathway files in GMT format (--gene-set), it will find out whether each gene set belongs to any pathways, and add the pathway names at the end of the gene set name in the output. The #CHROM column is the chromosome number for each gene in the gene set, separated by a comma. The POS column is no longer chromosomal positions, because it doesn't make sense for gene sets. It is replaced with a string that shows whether the association of the gene set is more significant than each gene in the gene set, if you specify a result file from gene-based

analysis (--per-gene). This is a useful feature for result interpretation and candidate gene selection. The hypothesis and advantage of gene set analysis is to aggregate the association of individual genes so that the power of the analysis is increased. So a true disease-causing gene set should have a stronger association with the disease than each gene in the gene set. The content of the POS column can be separated into two parts by a colon. The first part is a 5-character string representing the percentage of genes in the gene set that has a p-value higher than the gene set p-value. For example, +++++ means 100%, ++++- means >80% <100%, and so on. The second part is the actual number of genes in the calculation of this percentage. You can use the Linux command grep to select interesting gene sets: `grep +++++ vFin.output`.

3.8. gnGBA

[\[Back to top\]](#)

gnGBA is a program to compute guilt-by-association within the GeneMania gene-gene association network. It reads a Seed File (-s) for a list of seed genes that are related to the disease etiology, which could be found by genome-wide association studies, linkage analyses and positional cloning, expression profiling, epigenetic assays, loss of heterozygosity studies, functional research on cell lines or animal models, or other study methods. It is better that the seed genes are drivers of the disease, not passengers at the downstream stage of the disease development. Please note that although it is perfectly fine to use a gene found by animal research, this program only accepts human genes, in this case, a human homolog. For undiagnosed diseases, they could be genes related to each of the sub-phenotypes of the disease, or genes related to similar diseases.

To make the list, you don't need the raw data from each study. Just read the literature about the disease, and find out what genes are significantly related to the disease by whatever studies. There are databases about disease-gene correlations, such as the DisGeNET. You can use it as a starting point, but it is always better to do the manual curation for higher accuracy of seed genes. The DisGeNET database has a "CURATED" subset. It is recommended to use that instead of the whole database.

This program writes a Score File to the standard output, which can be read by other programs of the framework, such as vAAA and vFIN. In this file, the first row contains gene symbols. The second row are log10 Bayes factors. Below is an example:

```
Trait\Gene ATAD3A HLA-A C1orf53 ..
Disease1 -0.002 1.3990 0.8266 ..
```

3.9. vMerge

[\[Back to top\]](#)

This program merges multiple gene lists from different analysis strategies or research groups to create a final candidate gene list for confirmation by NGS in additional samples. It simply takes the top gene from each list until a number of genes is reached (--num-genes). You can also specify a set of genes that the final list must contain (--must-have). It is better not to select genes by averaged rank number or even only the overlapping genes, because it is likely that a true gene may be found by one method but not the others due to different hypotheses, and the research strategies could be very different.

4. Tutorial

4.1. Variant Classification

[\[Back to top\]](#)

Please note that PERCH was not designed to follow the ACMG guideline, but rather similar to the IARC guideline, which is a quantitative integration of multiple lines of evidence. PERCH is not yet a one-stop variant classification tool; there are still many other functions that are not implemented. And it doesn't provide a graphical-user-interface (GUI). However, you are welcome to use PERCH as a component of a more comprehensive application to serve the end users.

To do variant classification by linkage analysis, association analysis, and deleteriousness prediction, the input files are slightly different from those in a gene discovery research. To correct for the fact that a proband is known to carry the variant of interest, vSEG needs to know who is the proband in each pedigree. This can be done by adding the string "(proband)" at the end of their SeqID in the Pedigree File. Because different genes

have a different penetrance model and prior probability, and different variants have a different set of pedigrees and probands, normally the Genotype File should contain only one variant.

In the command line, vSEG, vAAA and vFIN need an additional option to command that the analysis is for variant classification (--vc). In vSEG, this option makes the program to check whether probands are correctly specified. In vFIN, it lets the program output a posterior probability instead of a log10(Bayes factor). The default prior probability is non-informative, i.e., $\pi=0.5$. You can use the option (--prior) to set a specific prior probability for the gene in question. In all three programs, the option (--vc) demands the analysis to be variant-wise, not group-wise.

This is the command to do variant classification without association analysis:

```
vMAF Annotated_Genotype_File.tsv | vDEL | vSEG --ped=pedigree.txt --vc | vFIN --vc > MyStudy.out
```

4.2. Gene prioritization

[\[Back to top\]](#)

Suppose you have prepared the following files: a Genotype File with deleteriousness score, allele frequency and functional consequence annotations (ex1.tsv), a Structural Variations File (sv.vcf), a Pedigree File describing pedigree structures (ex1.ped), a Sample File listing all independent cases and controls (ex1.spl), and a Seed File containing seed genes (ex1.seeds). Below is the command to conduct gene prioritization. The first line computes biological relevance scores and saves to ex1.bio. The last line computes MaxAF, calculates BayesDel, performs co-segregation and association analyses, and then combines the scores.

Below are the commands to rank genes. If you don't want to do gnGBA or vSVA, just ignore the command and remove the corresponding program option in the last line.

```
gnGBA -s ex1.seeds > ex1.bio
vSVA sv.vcf --spl=ex1.spl --detail=ex1.sv --test=no
vMAF ex1.tsv | vDEL | vSEG --ped=ex1.ped | vAAA --spl=ex1.spl --sv=ex1.sv | vFIN --biol=ex1.bio > ex1.out
```