MAGMA manual (version 1.09)

TABLE OF CONTENTS

OVERVIEW

A basic analysis in MAGMA consists of two or three steps: first, an **annotation step** to map SNPs onto genes; second, a **gene analysis step** to compute gene p-values; and three, a **gene-level analysis step**: either a generalized gene-set analysis, a gene property analysis, or both. The gene-level analyses are all based on the gene analysis, and any of the possible gene analysis models can serve as input for a gene-level analysis. A detailed description of each of these three steps can be found in the corresponding sections below.

MAGMA analysis can be performed on raw GWAS data, or on SNP p-values. In the latter case, a reference data set (eg. 1,000 Genomes European panel) is used to account for linkage disequilibrium between SNPs. The input GWAS data or p-values are assumed to have undergone appropriate quality control and filtering prior to running the MAGMA analysis. When using imputed data, it is advisable to only use SNPs for which the quality of imputation is high. It is also strongly recommended to screen the data for population outliers, and to use principal components computed from the GWAS data (eg. using Eigenstrat) as covariates in the gene analysis to correct for possible population stratification.

MAGMA is a stand-alone program that is run form the command line. Input arguments for MAGMA take the form of flags (prefixed by --) followed by the relevant values needed for that flag (if any). Many flags accept additional optional modifiers. These are keywords specified after the value(s) for that flag that modify the behaviour of that flag. Some modifiers consist of only the keyword itself, other modifiers take further parameters specified by the = sign and a comma-separated list of parameter values. For example, the --annotate flag has no required values and has modifiers *nonhuman* (no parameter) and *filter* (one parameter). Thus, adding the flag "--annotate filter=filterfile.snps nonhuman" would tell MAGMA to perform annotation, that only the SNPs listed in the file 'filterfile.snps' should be included, and that the annotation is not for human genomes.

MAGMA is undergoing active development, and updates to the program and auxiliary files can be obtained from http://ctglab.nl/software/magma. Further questions, error reports, suggestions for improvements and feature requests can be sent to c.a.de.leeuw@vu.nl.

Quickstart

The following commands can be used to perform the basic analysis steps. Replace the all-caps values in square brackets (eg. [VALUE]) with the appropriate value for your analysis. See below for detailed explanation and additional options. It is assumed throughout that the MAGMA executable can be called from anywhere by typing 'magma' on the command line. If not, it will be necessary to add the full path to the MAGMA executable (eg. './magma' if it is in the current directory).

   When running any command MAGMA will write a log with details of what it is doing to the screen as well as to a .log file. **You are strongly advised to always inspect this log** to ensure that the program is doing what you are expecting and to make yourself aware of warnings issued by MAGMA (unlike errors, these do not stop the program). In addition, it allows you to determine if reported input and output values make sense.

   For example, a mismatch in SNP ID nomenclature between reference data and a SNP p-value would result in a large percentage of SNPs being dropped from the analysis. This will not stop the analysis unless no matched SNPs remain, nor will it trigger a warning, but it will be clear from the reported percentage of valid SNP p-values read from the p-value file.

Annotation
Annotation can be performed with the command:

```
magma --annotate --snp-loc [SNPLOC_FILE] --gene-loc [GENELOC_FILE] --out [ANNOT_PREFIX]
```

This will read SNP locations from the file [SNPLOC_FILE] and gene locations from the file [GENELOC_FILE], and produces the file [ANNOT_PREFIX].genes.annot containing the mapping of SNPs to genes. The [GENELOC_FILE] can be downloaded from the MAGMA website for different builds, the [SNPLOC_FILE] must be provided by the user. The .bim file of the genotype (reference) data can be used for this (make sure that the gene and SNP locations are based on the same human genome reference build).

Gene analysis - raw data
To perform gene analysis on raw GWAS data:

```
magma --bfile [DATA] --gene-annot [ANNOT_PREFIX].genes.annot --out [GENE_PREFIX]
```

This will perform gene analysis on GWAS data in binary PLINK format ([DATA].bed/.bim/.fam files) using the previously generated annotation file. It will output two files: [GENE_PREFIX].genes.out and [GENE_PREFIX].genes.raw. The .genes.out file contains the gene analysis results in human-readable format. The .genes.raw file is the intermediary file that serves as the input for subsequent gene-level analyses. To perform only a gene analysis, with no subsequent gene-set analysis, the --genes-only flag can be added. This suppresses the creation of the .genes.raw file, and significantly reduces the running time and memory required.

Gene analysis - SNP p-values
To perform gene analysis on SNP p-values:

```
magma --bfile [REFDATA] --pval [PVAL_FILE] N=[N] --gene-annot [ANNOT_PREFIX].genes.annot \
      --out [GENE_PREFIX]
```

This is similar to the raw data analysis, but replaces the raw data with a reference data set such as the 1,000 Genomes European panel (available on the MAGMA site) and a file with previously computed SNP p-values (in columns 'SNP' and 'P' in the file [PVAL_FILE]). The sample size [N] of the data the SNP p-values were obtained from also needs to be specified. If a column of sample sizes per SNP is available

in [PVAL_FILE], this can be used as well by using 'ncol=[N_COLUMN_NAME]' for the --pval flag (instead of 'N=[N]'). The --genes-only flag can again be set if no subsequent gene-set analysis will be performed.

Gene-set analysis
To perform a gene-set analysis:

```
magma --gene-results [GENE_PREFIX].genes.raw --set-annot [SET_FILE] --out [GS_PREFIX]
```

The results from the previously performed gene analysis are read in, as is the mapping of genes to gene sets specified in the file [SET_FILE] (with each row corresponding to a gene set: name of the gene set followed by the gene IDs, separated by whitespace). This will produce the file [GS_PREFIX].gsa.out with (competitive) gene-set analysis results. Analysis of continuous gene-level variables can be performed using:

```
magma --gene-results [GENE_PREFIX].genes.raw --gene-covar [COVAR_FILE] --out [GC_PREFIX]
```

The file [COVAR_FILE] is a data file with rows corresponding to genes, and columns corresponding to variables to be analysed (with the first column containing gene IDs). This command will produce the file [GC_PREFIX].gsa.out with analysis results for each of the variables in the file.

Annotation

Overview

The annotation step is a pre-processing step prior to the actual analysis, in which SNPs are mapped to genes. The mapping is based on genomic location, assigning a SNP to a gene if the SNP's location falls inside the region provided for each gene; typically this region is defined by the transcription start and stop sites of that gene. Because genomic location is relative to a particular human genome reference build, it is crucial to make sure that the SNP locations you are using as input are based on the same human genome build as the gene locations. Gene locations for protein-coding genes (using Entrez gene IDs) are provided on the MAGMA website for builds 36 (hg18), 37 (hg19) and 38. For the SNP locations the .bim file of the input genotype data you are using is often the best choice. Though SNP locations can be taken from an external source as well, this may not contain locations for all the SNPs in your data, which would result in those SNPs being excluded from the analysis.

Optionally, SNPs near a gene but outside its transcription region can be mapped to that gene as well by specifying an annotation window around the genes. SNPs in that window will be annotated to that gene as well. In this way nearby SNPs still relevant to that gene (for example because they are in the gene's promoter region) can be included in that gene. However, SNPs not relevant to that gene may be included in this way as well, so it is advisable to keep this window small.

The annotation step will produce an output file with the .genes.annot suffix, with each row corresponding to a gene, containing the gene ID, a specification of the gene's location, and a list of SNP IDs of SNPs mapped to that gene.

Note: analysis of chromosome Y (or chromosomes W and Z, for non-human data) is at present not supported. Please contact us if you have data for these chromosomes that wish to include in your analyses.

Running the annotation

The annotation is performed using the --annotate flag, and further requires the --snp-loc and --gene-loc flags specifying the files containing the SNP and gene locations respectively. The --out flag can further be added to specify the file prefix for the output files. The basic annotation command will therefore look like this:

```
magma --annotate --snp-loc [SNPLOC_FILE] --gene-loc [GENELOC_FILE] --out [OUTPUT_PREFIX]
```

The SNP location file should contain three columns: SNP ID, chromosome, and base pair position. These should be the first three columns in that file (additional columns are ignored). The only exception is if you use a .bim file from binary PLINK data, this can be provided to MAGMA as a SNP location file without modification.

The gene location file must contain at least four columns, in this order: gene ID, chromosome, start site, stop site. Optionally the strand of the gene can be included in a fifth column as well, though this is only used when annotating with an asymmetrical gene window (see below).

For both the SNP and gene IDs, MAGMA will accept any nomenclature. The only requirement is that the IDs do not contain whitespace characters, and do not begin with a # character. Autosomal chromosomes are coded numerically from 1 through 22, the sex chromosomes can be coded either as X and Y or as 23 and 24. If included, gene strands must be coded as + for the sense/positive strand and - for the antisense/negative strand.

Adding an annotation window around genes

To include SNPs in a window around genes, you can set the *window* modifier on the --annotate flag. This modifier takes either one or two values, specifying the size of the window in kilobases. If only one

value is provided the window is symmetrical, extending the annotation region by the specified number of kilobases in both directions. If two values are provided the first value is the size of the upstream (towards 5' end) window, and the second value the size of the downstream (towards 3' end) window. For example, '--annotate window=5,1.5' would set a 5kb upstream and 1.5kb downstream window.

If you want to specify an asymmetrical gene window that is strand-insensitive, the additional modifier *ignore-strand* is added. In this case all genes are assumed to be on the positive strand, and the strand column in the gene location file (if any) is ignored.

Restricting the annotation to a subset of SNPs

If you want the annotation to be restricted to a subset of SNPs, the *filter* modifier can be added specifying a file containing the IDs of SNPs you want to include (eg. '--annotate filter=example.bim'). SNP IDs will be read from the first column of the file only, or from the second column if it has a .bim file extension. This option can be useful for example to obtain an annotation restricted to only SNPs in a particular data set, or to restrict a subsequent gene analysis to SNPs of a certain (functional or otherwise) type.

Annotation of non-human genomes

MAGMA supports annotation and analysis of (diploid) nonhuman genomes, by adding the *nonhuman* modifier to the --annotate flag. This will suppress the translation of chromosome codes 23 and 24 to X and Y, and will allow numeric chromosome/scaffold codes up to 2 million. Sex chromosomes are coded as X, Y, W and Z. The nonhuman modifier will be encoded in the .genes.annot file, so that subsequent analyses using this annotation will automatically be recognised as pertaining to non-human data.

Gene Analysis

Overview

In the gene analysis step the gene p-values and other gene-level metrics are computed. Correlations between neighbouring genes are computed as well, in preparation for the gene-level analysis. The gene analysis results are output into a formatted output file with .genes.out suffix. The same results plus gene correlations are also stored in a .genes.raw file, which serves as input for subsequent gene-level analysis. The latter can be suppressed by adding the --genes-only flag, to reduce computing time when only a gene analysis is needed.

Gene analysis can be performed on raw genotype and phenotype input data, as well as on SNP p-values from an existing (meta-)analysis. In the latter case raw genotype data is still needed to serve as a reference for LD. Publically available data sets such as the 1,000 Genomes data set can be used for this purpose. The two main requirements for the reference data is that 1) there is a strong overlap between it and the SNP p-value input file in the SNPs it contains, since only SNPs that occur in both files will be used in the analysis; and 2) that the general ancestry (ie. European, African, Asian, et cetera), of the reference data matches that of the data the SNP p-values were computed from. For most analysis a European ancestry reference data set is appropriate, and the European panel of the 1,000 Genomes phase 3 data is available for download from the MAGMA website.

A range of additional options is available to change the behaviour of the gene analysis. These are briefly summarized in the table below, and discussed in more detail in the remainder of this chapter. As indicated, some options are only available when analysing raw genotype data, and cannot be used when the --pval flag is set.

Flag	Used for	Requires raw data input?
--gene-model	Specifying the gene analysis model	Depends on model
--burden	Specifying burden score settings for rare variants	Yes
--covar	Including covariates from an external file	Yes
	Including gender as a covariate	
	Including a gene-covariate interaction term in the analysis	
--pheno	Specifying an alternate phenotype from an external file	Yes
--meta	Performing a meta-analysis on gene analysis results	No
	Preparing a meta-analysis for gene-level analysis	
--batch / --merge	Performing analysis in smaller parts, for distributed processing	No
--gene-settings	Specifying various settings affecting:	No
	- Permutation behaviour (if applicable)	
	- Internal SNP quality control	
	- SNP inclusion/exclusion lists	
--big-data	Special mode for reducing running time and memory usage when analysing very large data sets	No
--genes-only	Suppressing computation of the gene-gene correlations used for subsequent gene-level analysis	No
--out	Specifies the output file prefix	No

Gene analysis on raw genotype data

To perform a raw data gene analysis at default settings, two flags must be set: the --bfile flag to specify the genotype data, and the --gene-annot flag to specify the annotation file you created in the annotation step. The basic gene analysis command will therefore look like this:

```
magma --bfile [DATA] --gene-annot [ANNOT].genes.annot --out [OUTPUT_PREFIX]
```

The genotype data used in MAGMA must be in binary PLINK format, and only the file prefix needs to be specified. Given the above command, MAGMA will look for the [DATA].bed, [DATA].bim and [DATA].fam files. At its default settings, MAGMA will use the principal components regression gene analysis model. It will also use low level burden scoring to deal with very rare variants, and will add gender as a covariate when analysing the X chromosome. The sections following this describe how to change these defaults if desired.

Gene analysis on SNP p-value data

When analysing SNP p-value data, the --pval flag must be added to specify the file containing the SNP p-values. In this case the genotype data specified by --bfile is used to specify the reference data used to estimate LD between SNPs; any phenotype contained in that data is ignored. The basic gene analysis command will thus change to one of the two following (the --out flag is omitted here for reasons of space). In this case, MAGMA will default to the *snp-wise=mean* gene analysis model.

```
magma --bfile [DATA] --gene-annot [ANNOT].genes.annot --pval [PVAL_FILE] N=[N]
magma --bfile [DATA] --gene-annot [ANNOT].genes.annot --pval [PVAL_FILE] ncol=[N_COL]
```

The p-value file must be a plain text data file with each row corresponding to a SNP. If MAGMA detects a header in the file it will look for SNP IDs and p-values in the SNP and P column respectively. If no header is found it will use the first column for SNP IDs and the second column for p-values. If you want to use different columns instead, you can add the *use* modifier (with two values: SNP ID column, then p-value column) to the --pval flag to specify both, or the *snp-id* and *pval* modifiers (with one value each) to specify which. Columns can be specified by name (if there is a header) or index, so for example adding 'use=rsid,6' will look for SNP IDs in the column named 'rsid', and for p-values in the sixth column.

The *N* and *ncol* modifiers are used to specify the sample size. The *N* modifier is used to specify the sample size directly (the total sample size, also when using case-control analysis results). Optionally, a different sample size can be specified for the X and Y chromosomes. When using *N* with two values, the second is used for both X and Y; with three values, the second is used for X and the third for Y. Alternatively, sample size can also be set by using the *ncol* modifier to specify a column in the p-value file that contains the sample size used per SNP. This option is recommended if such a column is present, especially when analysing SNP meta-analysis results where sample size per SNP can vary considerably.

The *duplicate* modifier can be used to specify the desired behaviour for dealing with duplicate SNPs in the file, and can be set to one of four values: 'drop', 'first', 'last', and 'error'. When set to 'drop', the corresponding SNP is removed from the analysis entirely. When set to 'first' or 'last', either the first or the last entry for that SNPs in the file is used. When set to 'error', the program terminates if encountering any duplicate SNPs. The default mode is 'duplicate=drop'. Note that SNPs are only checked for duplication if they are present in the genotype data, and if they have a non-missing p-value (and sample size, if *ncol* is set). When synonymous SNP IDs have been loaded, different SNP IDs referring to the same SNP are considered duplicates as well. Unless *duplicate* is set to 'error', a list of duplicate SNPs will be written to the supplementary log file.

When running an analysis using SNP p-value data it is advisable to include a SNP synonym file, as the SNP IDs used in the reference data and SNP p-value file may contain different IDs referring to the same SNP. See the section *Accounting for synonymous SNP IDs* below.

Accounting for synonymous SNP IDs

Over the course of successive dbSNP releases, some distinct rs IDs have been merged into a single ID when they were found to refer to the same SNP. If input files are used that were created using different dbSNP versions, loss of data may therefore occur due to some SNPs being designated with different rs

IDs in different files. This is a concern in particular when using the --pval option, as any SNP p-value for which no corresponding genotype data can be found in the reference data is discarded.

To address this issue, an additional input file can be supplied to MAGMA specifying SNP IDs that are synonymous with each other (see Appendix A for details on the file format). For the dbSNP rs IDs, a complete synonym file that can be used for this purpose is available from the MAGMA website. The synonym file can be specified by adding 'synonyms=[FILE]' to the --bfile flag. In addition, if a file '[DATA].synonyms' is found (where [DATA] is the PLINK data prefix specified using --bfile) this will automatically be used as a synonym file unless the *synonyms* modifier is set. The 1,000 Genomes reference data provided on the MAGMA website include such a .synonyms file. Set 'synonyms=0' on the --bfile flag to suppress automatically loading such synonym files.

When loading SNP ID synonyms, MAGMA may detect SNP IDs in the genotype data that are synonyms of each other. The *synonym-dup* modifier for the --bfile flag can be used to specify the desired behaviour for dealing with such SNPs. This modifier can be set to one of four values: 'drop', 'drop-dup', 'skip', 'skip-dup' and 'error'. When set to 'drop', SNPs that have multiple synonyms in the data are removed from the analysis. Conversely, when set to 'skip' the SNPs are left in the data and the synonym entry in the synonym file is skipped. When set to 'drop-dup', for each synonym entry only the first listed in the synonym file is retained; for subsequent SNP IDs in the same entry that are found in the data are removed, and their IDs are mapped as synonyms to the first SNP. When set to 'skip-dup' the genotype data for all synonymous SNPs is retained; SNP IDs not found in the data are mapped to the first SNP in the synonym entry that is. Finally, when set to 'error', the program will simply terminate when encountering synonymous SNPs in the data. The default mode is 'synonym-dup=skip'. Unless *synonym-dup* is set to error, a list of synonymous SNPs in the data will be written to the supplementary log file.

Note that in the synonym files provided on the MAGMA website, the SNP IDs in each synonym entry are ordered, which is relevant when using the 'drop-dup' and 'skip-dup' options. The first ID listed is the current ID of the underlying SNP, according to the most recent version of dbSNP at the time the synonym file was created. The subsequent IDs in the entry are listed in increasing numerical order of their rs ID, since when two rs IDs are merged in dbSNP the lowest of the two is (almost) always retained.

Specifying the gene analysis model

A number of different gene analysis models is available in MAGMA, which can be specified by the --gene-model flag. Statistically, the different models are sensitive to different genetic architectures, which can vary by gene. More practically, they also vary in the time it takes to run them, although in practice they should all complete fairly quickly.

At present three base analysis models are available: the principal components regression (*linreg*) model, the SNP-wise Mean (*snp-wise=mean*) model and the SNP-wise Top 1 (*snp-wise=top*) model. The *snp-wise=top* model is most sensitive when only a small proportion of SNPs in a gene show association. The *snp-wise=mean* model is more attuned to the mean SNP association, though it skews towards associations in areas of higher LD in a gene. The *linreg* is also designed to be sensitive to the mean level of association and has better power to detect them in low LD areas, though is less sensitive when only a small proportion of SNPs is associated.

In addition to the base analysis models MAGMA also provides the class of multi-models. For each gene these run several of the base analysis models and combine the resulting gene p-values into an aggregate p-value. This has the advantage of yielding a more even distribution of statistical power and sensitivity for a wider range of different genetic architectures, although at the cost of a somewhat increased running time. The p-values for the individual models are also included in the output, though subsequent gene-level analyses will always be based on the aggregate p-value.

A brief overview of the different models is provided in the table below, along with the modifier for the --gene-model flag used to specify it.

Modifier	Description	Speed
linreg **(raw data default)**	Principal components regression Regresses the phenotype on principal components derived from the SNPs in a gene **Requires raw genotype data**	Very fast
snp-wise snp-wise=mean **(default if using --pval)**	Test of mean SNP association Uses sum of squared SNP Z-statistics as test statistic Uses simulation-based empirical p-values as backstop for genes where numerical integration fails	Fast
snp-wise=top snp-wise=top,1	Test of top SNP association Uses lowest SNP p-value as test statistic	Moderate to fast
snp-wise=top,P *with P a number between 0 and 1 or greater than 1*	Test of top SNP associations Uses sum of -log(SNP p-value) for top SNPs as statistic - The top number P of SNPs if K is greater than 1 - The top proportion P of SNPs if P is between 0 and 1 As this model requires permutation, use of the *snp-wise=top* or *multi=snp-wise* model is recommended instead over this one	Slow
multi=all	Aggregate of *linreg*, *snp-wise=mean* and *snp-wise=top* Computes all three models for each gene, then combines the three p-values into an aggregate p-value **Requires raw genotype data**	Moderate
multi=mean	Same as *multi=all*, but using only the *linreg* and *snp-wise=mean* models **Requires raw genotype data**	Fast
multi=snp-wise snp-wise=multi	Same as *multi=all*, but using only the *snp-wise=mean* and *snp-wise=top* models	Moderate
multi	Equivalent to multi=all if analysing raw data, equivalent to multi=snp-wise when --pval is set	Moderate

When using a multi-model in combination with an interaction term or the --partition flag, p-values for the individual models are suppressed in the output by default (as the total number of computed p-values can become quite large). Add the *multi-show-all* modifier to override this behaviour.

NOTE: the SNP-wise Mean model was changed in v1.08 to correct an issue with inflation of type 1 error rates that could occur in dense SNP data, particularly in very large genes (in number of SNPs). Compared to the original version, the test statistic has been slightly changed, which also results in small changes in the NPARAM values. A numerical integration procedure is now used to obtain the p-value. As this process can sometimes fail, when this happens a simulation-based p-value is computed instead. This uses an adaptive simulation procedure with up to 1 billion simulations. It returns a p-value of 5e-10 if no simulated value exceeds the observed test statistic. More details on this model change are provided in a separate document that can be found on the MAGMA website.

Rare variant data
Rare variant data (as well as mixed rare and common variant data) can be analysed without modification in MAGMA, gene analysis proceeds in the same way regardless of allele frequency. The SNP-wise Mean model is equivalent to the commonly used SKAT model with inverse variance weights. However, power to detect associations can deteriorate as the minor allele count approaches zero,

especially in the presence of (more) common variants. To help address this a burden scoring mechanism is available, which can be set using the --burden flag (only available for raw data analysis).

In MAGMA, burden scores are implemented as a pre-processing step. A threshold is first specified on either MAF or MAC to designate SNPs as 'rare'. For each gene, one or more burden scores are then created by summing the rare variants in that gene. The individual rare variants are then removed from the gene and replaced by these burden scores. Analysis of that gene then proceeds as normal, using the chosen gene analysis model (specified by --gene-model) to analyse the burden scores and remaining common variants (if any).

The 'rare' threshold is set by adding the --burden flag followed by a single numeric value. If this value is between 0 and 0.5 it is interpreted as a MAF threshold, if 1 or greater it is interpreted as a MAC threshold. If the --burden flag is not set an automatic burden score mechanism is activated at a MAF threshold of 1% and a MAC threshold of 100; in this case, only SNPs below both thresholds are designated as 'rare'. Setting "--burden 0" or adding the *no-autoburden* modifier to the --gene-model flag will disable burden scores altogether, although this is not recommended.

When computing the burden scores the included variants by default are weighted by their allele frequency, with weights for each variant computed as $w_j = 1/\sqrt{f_j(1-f_j)}$, where f_j is the minor allele frequency for SNP j. In effect this amounts to standardizing the variants prior to summing them. To use an unweighted sum instead, add the modifier 'freq-weighted=0' to the --burden flag.

In addition, a maximum is set on the number of variants that are included in a burden score. If the number of rare variants in a gene exceeds this maximum, multiple burden scores are computed for that gene. Rare variants are distributed as equally as possible over the burden scores, in the order in which they appear in the data. By default the maximum is set at 25, and can be adjusted using the 'multi-count=[MAX]' modifier. The maximum can be turned off by setting 'multi-count=0', in which case rare variants are always aggregated into a single burden score.

Analysis can be restricted to rare variants only. The *rare-only* modifier can be added to the --burden flag to remove all variants not designated as 'rare' from the analysis. Note that additional options for filtering based on minimum and maximum MAF or MAC are available through the --gene-settings flag as well, as is filtering based on inclusion or exclusion lists (see details below). Alternatively, the modifiers *rare-sum* or *rare-plain* can be used instead. These turn on the *rare-only* option and set the multi-count value to 0, with *rare-plain* also turning off the frequency weighting.

As further shorthands, the value for the --burden flag can be set to either 'all' (ie. "--burden all") or 'all-plain'. This will include all SNPs into a single burden score (ie. setting multi-count to 0) regardless of their MAF or MAC, and with 'all-plain' the frequency weighting is additionally turned off.

Weighted burden scores

User-defined weights can also be specified for the burden score analysis, using "--burden-weights [FILENAME]" with a file containing weights for each SNP. By default MAGMA will use the columns SNP and WEIGHT if the file contains a header, or the first and second column if it doesn't. As with the --pval flag different columns can be specified using the *use* modifier (or the individual *snp-id* and *weight* columns; both must be set when these are used), and the *duplicate* modifier can be used to determine what to do when duplicate SNP IDs are encountered in the file (by default these SNPs are removed from the analysis; see subsection **Gene analysis on SNP p-value data** above for more details).

The weight values specified in the file cannot be negative, and SNPs with weights set to zero are filtered from the data prior to analysis. SNPs not included in the weights file will be set to a weight of zero and be removed as well; however, adding 'default=[VALUE]' to the --burden-weights flag can be used to set a different default SNP weight. Note that although the weights are only used for the SNPs included in the burden scores, and are disregarded for common SNPs in the analysis, this filtering is applied to all SNPs. When no explicit weights are included for common variants in the weights file, the default weight must therefore be set to a non-zero value for those common variants to remain in the analysis.

Weights can also be specified by SNP type rather than individually per SNP, to simplify assigning weights to SNPs that have been annotated to a discrete set of categories. This can be specified by adding 'type-weights=[TYPE_WEIGHT_FILE]' to the --burden-weights flag. This file should contain two columns, with the SNP type names in the first column and the corresponding weight in the second. When this option is used, the values in the main SNP weight file are interpreted as type names rather than numeric weights (note that the default column name used is still WEIGHT), where the type names can be any mixture of numeric codes and character strings. Since SNPs with weights of zero are filtered from the analysis, setting the weight of a particular type to zero in the type weight file can be used to omit an entire category of SNPs from analysis.

The *type-default* modifier determines what happens when SNPs are assigned a type that isn't defined in the type weight file. By default this is set to 'drop', which results in such SNPs are removed from the analysis with a warning in the output log listing the unknown type. Alternatively this modifier can be set to 'default' to assign such SNPs the default weight instead (though that will still remove those SNPs if that default weight is left at zero), or to 'error' to terminate the analysis if an unknown type is encountered.

It should be noted that the settings specified using the --burden-weights flag do not change the default settings of the --burden flag. As such, under default settings a variant with a user-defined weight of u_j will be assigned an actual weight of $w_j = u_j/\sqrt{f_j(1 - f_j)}$, ie. the user-defined weights are applied to the standardized genotypes. The allele frequency weighting must be turned off under the --burden flag to use weighted sums of raw genotypes instead. Similarly, multiple weighted burden scores will still be created per gene if it contains larger number of SNPs, unless this option is explicitly turned off.

Adding covariates

When analysing raw genotype data, the --covar flag can be used to include covariates in the analysis to condition the gene associations on. Covariates can be included from a file by adding the modifier 'file=[COVAR_FILE]'. This file should contain family IDs and individual IDs as its first two columns, corresponding to the IDs in the genotype data .fam file.

The covariate file can contain a header with variable names, but this is not required. The file may contain individuals not in the genotype data, and need not contain all individuals in the genotype data; in this case, analysis is restricted to only those individuals present in both the genotype data and the covariate file. Missing values in the covariate file can be coded as NA; any individuals with missing values on covariates used in the analysis are also excluded from the analysis.

By default all variables in the file are included as covariates, but this can be modified by adding either the *include* or the *exclude* modifier specifying which variables to include or exclude. These take a comma-separated list of variable names (if a header is present) or indices, as well as ranges of either. For example, '--covar file=example.covar include=age-iq,3-5,ses' will include the variables from 'age' up to and including 'iq', variables 3 through 5, and 'ses'. Note that variables are numbered from the first variable after the leading FID and IID columns, ie. the first variable corresponds to the third column in the covariate file.

To use the gender variable encoded in the .fam file of the genotype data, add the modifier *use-sex* to the --covar flag. Note that by default, for chromosome X the gender value is automatically included. To suppress this behaviour, add the modifier *chrX-use-sex=0*.

To condition the gene analysis on SNPs in the data, add the modifier *include-snps* followed by the IDs of SNPs to be conditioned on (eg. '--covar include-snps=rs7517363,rs519723,rs12118262'). SNPs will be excluded if the missingness is too high (using twice the normal missingness threshold, to a maximum of 0.25; see also Internal QC section below). Otherwise, missing values are set to the mean of the observed values for that SNP.

Adding a gene by covariate interaction component

When analysing raw genotype data, a gene-covariate interaction component can be added to the analysis via the --covar flag. To do so, set a covariate file using the *file* modifier, and use the 'interact=[INTERACTOR]' modifier to specify which variable (by name or index) to use as the interactor. When the *interactor* modifier is used the other covariates in the file are not automatically included in the analysis. Use the *include* or *exclude* modifiers to explicitly add sets of covariates, or add the *include-all* modifier to include them all. To use the gender variable encoded in the .fam file of the genotype data as interactor, you can add the *interact-sex* modifier to --covar instead. Similarly, to use a specific SNP as interactor, add the *interact-snp* modifier specifying the ID of the interactor SNP. Note that only a single interactor can be included at a time.

When running an interaction analysis, a SNP-covariate interaction term is included for each SNP in a gene. Three gene p-values will then be computed: one for the joint main genotype effects (same as a regular gene analysis conditioned on that covariate), one for the joint interaction effects, and one for the full model (main and interaction effects combined). All three p-values are provided in the gene analysis output (except when using the 'snp-wise=top,P' gene analysis model), but subsequent gene-level analysis will always be based on the full model p-values.

Alternate phenotypes

When analysing raw genotype data, to use a different phenotype than the one in the .fam file use '--pheno file=[PHENO_FILE]'. As with the covariate file, the first two columns should be family ID and individual ID. The first variable after the FID/IID columns is selected by default. The 'use=[VAR]' modifier can be used to select a different variable. This can again be done by name (if a header is present) or by index (counting the first column after the FID/IID columns as variable 1). Missing values can be coded as NA. If the phenotype is binary, missing values can also be coded as -9 or 0, following the PLINK convention. A phenotype is considered binary if only the values NA, -9, 0, 1 and 2 are encountered.

Gene meta-analysis

The meta-analysis option can be used to perform gene meta-analysis on MAGMA results from multiple cohorts, as well as preparing for gene-level meta-analysis. The weighted Stouffer's Z method is used to combine the Z-scores for each gene across cohorts, with weights set to the square root of the sample size each Z-score is based on.

When performing the meta-analysis it is not required that a gene is present in each cohort; for each gene meta-analysis is performed using all cohorts that are available for that gene. MAGMA does check whether a gene's chromosome and location is the same across cohorts however. It is therefore important to use the same annotation in the analysis of the individual cohorts.

When performing gene meta-analysis on .genes.out files, this new Z-score and corresponding p-values is written to a new .genes.out file. Other values such as the number of SNPs in a gene are averaged over the corresponding values of the input files. When preparing gene-level meta-analysis for .genes.raw files, Z-scores and other values are combined in the same way, and gene correlations are merged as well. These are all written to a new .genes.raw file, which can be used in subsequent gene-level analyses as normal.

The meta-analysis is run using the --meta flag, using either the *genes* modifier to specify a list of .genes.out files, or the *raw* modifier to specify a list of .genes.raw files. Optionally, the *prefix* modifier can be added to the --meta flag. If this is set, the .genes.out and .genes.raw suffixes can be omitted. The meta-analysis commands would for example look something like this:

```
magma --meta genes=[COHORT1].genes.out, [COHORT2].genes.out --out [OUTPUT_PREFIX]
magma --meta raw=[COHORT1], [COHORT2], [COHORT3] prefix --out [OUTPUT_PREFIX]
```

As an alternative to the *genes* and *raw* modifiers, the *genes-file* and *raw-file* can be used instead, specifying a settings file. Each row in this settings file corresponds to a cohort to be used in the meta-analysis, and contains from one to four values. The first value is the name of the .genes.out or .genes.raw file; if the *prefix* modifier is set, the .genes.out and .genes.raw suffixes can again be omitted. The cohort file name can be followed by up to three sample size override values. If specified, these override the sample size values encoded in the input files for that cohort. If only one value is specified this is used for all genes; with two values, the second value is used for the X and Y chromosome, and with three values the second is used for the X chromosome and the third for the Y chromosome (see Appendix A for an example).

The *unweighted* modifier can be added to the --meta flag to override the default weighting and use the unweighted Stouffer's Z method instead. Additionally, the *correlations* modifier can be used to specify the name of a file containing a covariance matrix specifying cross-cohort correlations. This can be used to perform meta-analysis in case of sample overlap; for example the bivariate LD Score regression intercept for each pair of cohorts can be used to obtain these correlations. Note that the input covariance matrix is automatically standardized to a correlation matrix. The number of rows and columns of the matrix should be the same as the number of input files for the meta-analysis, and in the same order.

Analysing very large data sets

When analysing very large data sets, computational efficiency can become a major concern. Three options are available in MAGMA to facilitate analysing such data sets. The first is the batch mode, which allows analysis of the data in smaller chunks. Although this does not decrease the required memory or computation time as such, it does allow for analysis of the data to be performed in parallel, and distributed over multiple CPUs/computers. A full description of batch mode is provided below.

The second option is the chromosome batch mode. This is similar to normal batch mode, but splits the data in batches by chromosome. This can significantly reduce the amount of memory required to run the analysis per batch, and also allows for genotype data to be input per chromosome rather than in one big file.

The third option is the 'big data' mode, enabled by setting the --big-data flag. This mode is available for data with sample sizes greater than 10,000 individuals (in the analysis, ie. after removing individuals with missing values on phenotype and any covariates). The primary aim of this mode is to reduce the memory required to run the analysis, although it can significantly reduce the running time as well. This is achieved by computing the correlations between genes (and gene blocks, for very large genes) from only a subset of the individuals in the data, which reduces the amount of computation required but still yields sufficiently accurate correlations. 'Big data' mode is turned on automatically if the number of individuals in the data is at least 25,000 (set '--big-data off' to turn it off).

If only a gene analysis is needed, a further option to reduce memory demands and running time is to turn off computation of gene-gene correlations altogether with the --genes-only flag.

Distributed gene analysis in batch mode

When working with large data sets it can be convenient to process the data in smaller chunks which are analysed separately, and in parallel. For gene analysis in MAGMA this can be achieved by using the --batch flag, specifying the batch to be analysed as well as the total number of batches to use; the --batch flag is simply added to the command for running the gene analysis. For example, using '--batch 7 20' will split the data into 20 batches and analyse batch number 7. This should then be repeated for each of the 20 batches, which can then be merged using the --merge flag. It is crucial that each batch uses the same input data, and that all flags and modifiers are set to the same value. This includes the --out flag, as the output files all get an additional suffix (eg. .batch7_20.genes.out) to identify them.

Because MAGMA needs to compute correlations between genes (up to a certain distance from each other), there is a limit to the number of batches that can be used. Chromosomes are treated as

independent from each other and some larger chromosomes can often be divided into multiple independent batches, so in practice the maximum number of batches will be around 25 to 30. If the requested total number of chunks exceeds this maximum, it will be automatically reduced to the maximum. For commands intended to analyse batches beyond the maximum, no output other than a log file will be generated. However, when the --genes-only flag is set and gene correlations do not need to be computed, the maximum number of batches is simply the number of genes in the analysis.

When using the --batch flag, the number of batches can also be set to 'chr' to enable chromosome batch mode. In this case the batch number becomes the chromosome code; for chromosome X both 23 and X can be used (eg. '--batch X chr'). Using chromosome batch mode gives the additional option of using PLINK input data that has been split by chromosome (although it can be used with data in a single PLINK data set as well). Any occurrence of the string '#CHR#' in the prefix set by the --bfile flag will automatically be replaced by the current chromosome. For example, when using '--bfile chr#CHR#_data --batch 13 chr' MAGMA will look for PLINK data files with prefix 'chr13_data'.

When running chromosome batch mode, MAGMA will filter out SNPs not on the specified chromosome when loading the data in order to reduce memory usage. This filtering is based on the .bim file specified with the --bfile flag, and assumes that the chromosome column in that file is accurate. If this assumption is not met, this feature can be turned off by adding 'chr-filter=0' to the --batch flag (it is also turned off automatically if input files are specified by chromosome, using the #CHR# placeholder).

Merging of data generated in batch mode is done using '--merge [BATCH_OUT]', where [BATCH_OUT] is the value used for the --out flag when analysing the batches. MAGMA will automatically try to detect the number of batches from the available files. If this is not possible an error will be generated, and the number of batches that were used must be specified explicitly (eg. '--merge [BATCH_OUT] 20'). If normal batch mode was used, MAGMA will verify whether output for all batches is present, stopping if any are missing. However, if chromosome batch mode was used it will only give a warning for missing autosomal chromosomes and no warning for missing allosomal chromosomes, and will continue regardless. Note that when merging, the output files for the individual batches are not removed.

The commands for a batch analysis would for example look something like the following. The first command would be run 10 times, varying [P] from 1 to 10. This is then followed by the second command to merge. In this example the --out when merging is set to the same value as in the batch analysis, though this is not required (it also will not override output files from the individual batches, since those have an additional batch-specific suffix.

```
magma --bfile [DATA] --gene-annot [ANNOT].genes.annot --batch [P] 10 --out [BATCH_PREFIX]
magma --merge [BATCH_PREFIX] --out [BATCH_PREFIX]
```

Internal QC, filtering, permutation and other settings
Prior to gene analysis, MAGMA applies a number of internal SNP quality control steps, removing SNPs that do not pass from the analysis. Settings for some of these steps can be adjusted using the --gene-settings flag with the appropriate modifiers. Minimum SNP minor allele frequency and minor allele count can be specified using *snp-min-maf* and *snp-min-mac* respectively, set to the desired values (default is 0, though SNPs with no minor alleles are always removed). Similarly, maximum SNP minor allele frequency and minor allele count can be specified using *snp-max-maf* and *snp-max-mac* respectively, to filter out SNPs exceeding those values. The maximum allowed SNP missingness proportion can be set using *snp-max-miss* (default is 0.05).

A test for differential missingness is also performed, by evaluating the correlation between the phenotype and missingness status. SNPs which this correlation is significant at a given threshold are filtered out, with the threshold set by the *snp-diff* modifier (default is 1e-6).

Manual additional SNP filtering can also be performed by using the *snp-include* and *snp-exclude* modifiers, which should specify a file with SNP IDs to be included or excluded. Similarly,

individuals can also be filtered using the *indiv-include* and *indiv-exclude* modifiers; these filtering files should contain an FID-IID pair on every line. Note that SNPs and individuals in the include file may still be excluded based on the other filters.

For the principal components regression model, pruning is also performed on the principal components. This removes components that account for very little variance in the genotype data for a gene. These are very unlikely to contain relevant association with the phenotype but would otherwise still use degrees of freedom in the regression model reducing power. Pruning behaviour is primarily set using the *prune* modifier to set the proportion of genotypic variance to be retained (default is 0.95). In addition, the *prune-prop* modifier can be used to set the maximum proportion of PCs to retain (default is 1), and the *prune-count* modifier to set the maximum number of PCs to retain (default is all). PCs are pruned until all conditions are met.

The --gene-settings flag also controls the settings for permutation-based empirical gene p-values, using the *fixed-permp* or *adap-permp* modifiers to enable computation of an empirical p-value using a fixed number of permutations or an adaptive permutation procedure respectively. This empirical p-value is computed in addition to the normal p-value, except when using the 'snp-wise=top,P' model (with P not equal to 1). This model already uses adaptive permutation at its default settings, and therefore setting the permutation control modifiers only changes the mode and settings for the existing permutation.

When using *fixed-permp* the same number of permutations is computed for each gene. This which defaults at 1,000 permutations and can be changed by specifying the number of permutations for *fixed-permp* (eg. '--gene-settings fixed-permp=5000'). This simple approach is generally not very efficient however, since a large number of permutations only has added value if the p-value is very low.

Adaptive permutation solves this problem by varying the number of permutations per gene, depending on its p-value. A minimum and maximum number of permutations are set, and permutations are performed in batches of a predefined size. After completing a batching of permutations is completed the stopping criterion is evaluated. To do so, the number of permutations is counted for which the test statistic is more extreme than the test statistic for the observed data. If this count is at least as high as a specified threshold permutation is stopped, otherwise it continues until either the stopping criterion is met or the maximum number of permutations is reached.

The *adap-permp* modifier with no additional values will enable adaptive permutation with a maximum number of permutations of 1,000,000 and a stopping criterion of 10. Setting one value changes the maximum number of permutations, adding a second value changes the stopping criterion. For example, 'adap-permp=1e7,25' will run up to 10 million permutations per gene, using a stopping criterion of 25. The minimum number of permutations defaults at 1,000, and can be changed using the *min-perm* modifier set to the desired number of permutations.

The --seed [SEED] flag can be used to set the seed of the random number generator, where [SEED] is a positive number. This ensures that an analysis can be repeated with identical results, even if random numbers are being generated during the analysis. Note that not all gene analysis models use random numbers.

Output

The **.genes.out** file will contain the following output:

- **GENE**: the gene ID as specified in the annotation file
- **CHR**: the chromosome the gene is on
- **START/STOP**: the annotation boundaries of the gene on that chromosome (this includes any window around the gene applied during annotation)
- **NSNPS**: the number of SNPs annotated to that gene that were found in the data and were not excluded based on internal SNP QC
- **NRARE**: the number of those SNPs classified as rare (when using the --burden option)

- **NPARAM**: the number of relevant parameters used in the model. For the SNP-wise models this is an approximate value; for the principal components regression (raw data default) this is set to the number of principal components retained after pruning; for the multi-models this is the mean NPARAM value of the component base models
- **N**: the sample size used when analysing that gene; can differ for allosomal chromosomes or when analysing SNP p-value input with variable sample size by SNP (due to missingness or differences in coverage in meta-analysis)
- **DATASETS**: the number of cohorts the p-value is based on (when performing gene meta-analysis)
- **ZSTAT**: the Z-value for the gene, based on its (permutation) p-value; this is what is used as the measure of gene association in the gene-level analyses
- **P**: the gene p-value, using asymptotic sampling distribution (if available)
  - When using an interaction model, three p-value columns **P_FULL**, **P_INTERACT** and **P_MAIN** are included instead
  - When using the multi-model, multiple p-value columns named **P_JOINT** and one **P_[MODEL]** for each individual model are included instead; unless an interaction term is included as well, in which case only **P_FULL_JOINT**, **P_INTERACT_JOINT** and **P_MAIN_JOINT** are included
- **PERMP**: the gene p-value, using permutation-based sampling distribution (if enabled)
- **NPERM**: the number of permutations PERMP is based on (if applicable)
- **RSQ/RSQ_ADJ**: the R^2 and adjusted R^2 values for the model (for principal components regression only); when using covariates, this is relative to the phenotypic variance after removing the variance accounted for by those phenotypes (only when using principal components regression). Note that the adjusted R^2 values are capped at 0

Gene-level Analysis

Overview

Gene-set analysis in MAGMA is implemented as a special case of a more general gene-level linear regression model. Internally, a gene-level data matrix is constructed in which genes (rather than people) are the data points / rows in the data matrix. Gene sets are represented as binary indicator variables, coded 1 for genes in that gene set and 0 otherwise. Other properties of genes such as the number of SNPs they contain are represented as continuous variables. The association that a gene has with the phenotype is quantified as a Z-score, a probit transformation of the gene p-value computed during the gene analysis step ($Z_g = \Phi(1 - P_g)$), mapping low p-values onto high positive Z-scores; $Z_g = 0$ corresponds to $P_g = 0.5$). Some outlier truncation is subsequently applied to the Z-scores to reduce the risk of individual genes unduly affecting the results of the analysis.

The competitive gene-set analysis is implemented as a linear regression model on this gene-level data matrix, $Z = \beta_0 + S\beta_s + C\beta_c + \varepsilon$, with S the gene-set indicator variable and C a matrix of covariates (such as gene size) to correct for. The residuals ε are modelled as multivariate normal with correlations set to the gene-gene correlations computed during the gene analysis. This is to account for the LD between genes in close proximity to each other, which would otherwise invalidate any statistical tests on the model.

The gene-set p-value is the p-value resulting from a test on the coefficient β_s, testing the null hypothesis $H_0: \beta_s = 0$ against the one-sided alternative $H_A: \beta_s > 0$. In effect, this tests whether the (conditional) mean association with the phenotype of genes in the gene set is greater than that of genes not in the gene set. By default the gene set variable is conditioned on the gene size, gene density (representing the relative level of LD between SNPs in that gene) and the inverse of the mean MAC in the gene (to correct for potential power loss in very low MAC SNPs), as well the log value of these three variables. In addition, the per-gene sample size as well as the log thereof are condition on, if this varies between genes (for example because not all genes were present in all cohorts in a meta-analysis). Additional covariates to condition on can be added to the model using the conditional gene-set / gene-property analysis option.

Also available as a form of gene-level analysis is the ==gene property== analysis. This is essentially the same model as the competitive gene-set analysis, but using a continuous variable as predictor rather than a binary indicator like a gene set. A difference is that by default the test of the coefficient is two-sided in the gene property analysis. This can be used to test for example whether tissue-specific differential expression levels are predictive of associations with a phenotype (which could point to the relevance of particular tissues to the aetiology of that phenotype).

When running a gene property analysis it is important to consider the choice of variable, and how it is scaled. For example, when using variables based on the p-values of differential expression tests, it may be more informative to use the -log values of those p-values rather than corresponding Z-scores, as the log transformation more effectively compresses the higher p-values into a smaller range (these will likely reflect mostly noise rather than genuine differential expression, and are of less interest). It is also advisable to check the continuous predictors for outliers and truncate them to more reasonable values if present, as the results might otherwise be unduly influenced by an individual gene.

A range of different options is available to perform conditional and joint analysis of multiple gene sets and/or gene properties simultaneously. These can be used to analyse the joint effects of gene sets and properties, as well to detect or rule out confounding between them in their relation the genetic associations. In addition, interaction analysis models are implemented to analyse dependency of gene set and gene property effects on each other. Such effects can occur for example when specifically the combination of two gene sets plays a role in the genetics of the phenotype, rather than either gene set individually; similarly, it could arise when the involvement of a particular pathway is tissue-specific, primarily restricted to genes in that pathway that are strongly expressed in a given tissue.

A detailed guideline and workflow for performing gene-set and gene property analysis, with a particular emphasis on the conditional, joint and interaction analysis models, can be found in our paper *Conditional and interaction gene-set analysis reveals novel functional pathways for blood pressure* (De Leeuw, et al., Nature Communications (2018)). Supplementary R scripts for assisting in post-hoc inspection of results are also provided in the MAGMA website.

Providing input

Input for the gene analysis can be supplied in two forms, which can be used simultaneously. The first form of input is binary gene sets, which are specified using the --set-annot flag:

```
magma --gene-results [GENE_RESULTS].genes.raw --set-annot [SET_FILE] --out [OUTPUT_PREFIX]
```

By default MAGMA expects row-based input, where each row in the input file corresponds to a gene set definition. The first value on each row is the name of the gene set, followed by the IDs of genes in that gene set. Any values that do not match IDs of genes in the .genes.raw file will be ignored.

Gene-set definitions can also be supplied in a column-based form, by adding the *col* modifier to the --set-annot flag. This modifier should specify two values, indicating the index for the gene ID column and gene-set name column respectively (eg. '--set-annot sets.col col=3,1' will read gene - gene-set pairs from the sets.col file, looking for gene IDs in the third column and gene-set names in the first column). The two columns can also be specified separately, using the *gene-col* and *set-col* modifiers instead of the *col* modifier.

The second form of input is continuous gene properties, specified using the --gene-covar flag:

```
magma --gene-results [GENE_RESULTS].genes.raw --gene-covar [COVAR_FILE] --out [OUTPUT_PREFIX]
```

The specified [COVAR_FILE] must be a column-based input file, with each column corresponding to an input variable and the rows corresponding to genes. The first column must contain the gene IDs, and the file must also have a header. Only numeric values can be used, and missing values must be coded as NA. Genes in the input file that are not present in the .genes.raw file will be ignored.

When using the --gene-covar input format, there are two types of missingness to consider and two options that govern how they are dealt with. The first type of missingness that can occur is when a gene is present in the file, but some of its values are set to NA. The *missing-values* modifier for the --gene-covar flag specifies how to deal with these missing values, and can be set to either 'drop', 'median', 'mean', or a specific numeric value. When set to 'drop' all genes with any missing values on them are removed from the data, with the other three settings the missing values are replaced by either the median or mean of the observed values for that same variable, or by the specified value. The default setting for this option is 'median'. However, for any variable, if the missingness rate is greater than 0.05 it is removed from the data instead. The *max-miss* modifier can be used to change this exclusion threshold (to a maximum of 0.2).

The second type of missingness occurs when genes are completely missing from the file. By default, such genes are removed from the data entirely. By adding 'missing-genes=fill' to the --gene-covar flag these genes are treated as having all missing values, which are dealt with as specified by the *missing-values* modifier. This option should be used with caution (especially when the number of missing genes is large), but can be useful for example when performing a joint analysis of gene properties and gene sets when it might be undesirable to remove all those genes from the analysis.

When using the --gene-covar flag, all variables in the file will be read and processed, even if only not all of them are to be used in the subsequent analyses. To read only a subset of variables from the file the *filter-read* or *filter-skip* can be used, specifying a list of variables to either read in and keep or conversely to skip over. These modifiers take a comma-separated list of variable names or variable indices, or ranges of either. For example, '--gene-covar file=example.covar include=brain-liver,3-5,kidney' will include the variables from 'brain' up to and including 'liver', variables 3 through 5, and

'kidney'. Note that variables are numbered from the first variable after the leading gene ID column, ie. the first variable corresponds to the second column in the covariate file.

Basic analysis

If no additional settings are specified, when the --set-annot and/or --gene-covar flags are set the variables will be analysed one at a time, with the analysis results written to a .gsa.out file. To analyse only a subset of input variables, the *analyse* modifier on the --model flag can be used. This can be set to 'sets' or 'cov'/'covar'/'covariates' to analyse only gene sets or only gene properties. It can also be set to 'list' followed by a comma-separated list of variable names, or to 'file' followed by the name of a file containing a list of variables names. In both cases, only the listed variables are analysed (names of variables not in the input files will be skipped).

When using --set-annot and --gene-covar simultaneously, it is possible for the same variable name to occur in both input files. In this case, to disambiguate which variable is intended, append :set: or :SET: for gene sets and :cov: or :COV: for gene properties to the front of the variable name when specifying it (for example for the *analyse* modifier, but also for all other --model options for which specific variables are specified).

The --model flag can also be used to control the direction of testing using the *direction*, *direction-sets* and *direction-covar* modifiers. By default, the testing direction is set to one-sided, positive for gene sets and to two-sided for gene properties. The *direction* modifier sets the testing direction for gene sets and gene properties simultaneously. For all three modifiers, accepted values are 'pos'/'positive'/'greater', 'neg'/'negative'/'smaller' and 'both'/'two'/'twosided'/'two-sided'.

For analysis of gene sets, two additional modifiers are available for the --model flag. The first is the *self-contained* modifier, which will perform an additional self-contained gene-set analysis for all gene sets, writing the output to a separate .gsa.self.out file. The second is the *alpha* modifier, which sets the significance level. When performing the standard single-variable analysis, for each significant gene set (after multiple testing correction) additional per-gene information is provided in a .gsa.sets.genes.out file as well as for all genes in the analysis in a .gsa.genes.out file (see also the *Output* section below).

To control the automatic correction for technical data-level properties the *correct* modifier is used. Setting 'correct=all' (default) or 'correct=none' turns all corrections on or off. It can also be set to 'include' or 'exclude' followed by a list of internal variables to correct for, or not correct for. Valid internal variable names are 'size'/'genesize', 'density', 'mac', and 'sampsize'/'N', which designate the gene size in number of SNPs, the gene density (a measure of within-gene LD), the inverse mean minor allele count, and the sample size (if this does not vary across genes, it is disabled automatically).

The --settings flag controls some additional parameters for the basic analysis. Modifiers *gene-include* and *gene-exclude* can be used to specify a file of gene IDs to either include in the analysis (discarding the rest) or to exclude from the analysis.

The *outlier* and *covar-outlier* modifiers can be used to specify the upper and lower thresholds on the Z-scores and gene properties for considering a value an outlier; such outlier values are truncated to the threshold value when performing the analysis. These thresholds are defined in terms of the standard deviation of the variable. For the Z-scores (using *outlier*), two values must be provided, for the lower and upper bound respectively. The lower bound is defined as the specified number of standard deviations below zero, whereas the upper bound is defined as the specified number of standard deviations above the mean score. For gene properties (using *covar-outlier*) only a single value is specified, setting the thresholds to that number of standard deviations above and below the mean. Default values for these settings are 3 and 6 for *outlier*, and 5 for *covar-outlier*.

The final two modifiers for --settings are *gene-info* and *abbreviate*. The *gene-info* modifier turns on the output of the .gsa.genes.out file with per-gene information (see above). The *abbreviate* modifier controls the variable name abbreviation for the .gsa.out file, which abbreviates long variable names to make the file more readable. By default the maximum length is set to 30 characters, and if

any variable names are abbreviated the full names are provided in an additional, final column. The *abbreviate* modifier can be set to 0 to disable abbreviation, to a positive value (at least 20) to change the maximum length, or to 'file' to write all abbreviated names and corresponding full names to a separate file (abbreviations are all unique, so different variable names will always map to different abbreviated names). To change both the maximum length and write abbreviations to file, specify the length followed by 'file'.

Conditional and joint analysis

The gene-level regression model can accommodate joint analysis of any number of input variables, testing their association conditional on the other variables in the model. The models to be analysed can be specified via two sets of modifiers for the --model flag, specifying conditional and joint analyses.

For conditional analysis, there are three main types of modifier: *condition*, *condition-hide*, and *condition-residualize* (a fourth type, *condition-interaction*, is discussed in the next section). All three function in largely the same way. The variables specified are included as covariates in all models that will be analysed, and they can therefore be combined each other and with each of the joint and interaction analysis options, as well as the standard single variable analysis. The variables can be specified in one of two ways: either directly as a comma-separated list of variable names, or as a list in a file (whitespace separated, can be spread over any number of lines). In the latter case, the first value for the modifier must be 'file', followed by the name of the file to be used.

For variables specified with the *condition* variable, for each model analysed the .gsa.out output file will also contain results for those variables. An extra MODEL column is added to the output file to indicate the model ID for each variable in the file, to indicate which results belong to the same analysis (all results for a joint analysis will be printed consecutively in the output file). For variables specified using the *condition-hide* modifier the analysis is performed in the same way, with the only difference being that no results are printed in the output file.

Variables specified via the *condition-residualize* modifier are not directly included in the analyses, but rather are processed in the same way as the internal variables automatically included for correction (see *Basic Analysis* section above). A regression model is fitted using these variables, and the residuals from this model are used as the outcome variable in all subsequent analyses. Any collinearity between variables included in this model is pruned away. No output is generated for these variables.

Whereas in a normal conditional association any association shared between two variables is distributed over their coefficients, when residualizing based on a set of variables any overlap in association is assigned to those variables. The estimated effects of variables that use these residuals as their outcome variable will therefore only cover association with the outcome not already explained by the residualizing variables.

This option can therefore be used to ensure that any such overlapping effects are not counted towards the tested variables (for example when the residualizing variables represent potential technical artifacts in the data that will not have genuine, biological meaning). This type of analysis can be overly conservative if the overlapping association can be considered biologically relevant however, so this option should be used with care.

Variables can only be specified for one of the three 'condition' modifiers. If a variable is specified for more than one of these modifiers, *condition* will take precedence over *condition-hide*, and *condition-residualize* will take precedence over both other modifiers.

Two modifiers are available to perform more general joint analysis models. As with the *condition* modifier, a MODEL column is added to the output file to indicate which rows of results belong together in a single model. The *joint* modifier is used to specify a file containing model definitions, with each line in the file corresponding to a separate model. Variables on each line are then analysed together in a single model, together with any variables specified via the 'condition' modifiers. These models can contain any number of variables, though they cannot contain variables also used to condition on; models containing such variables are skipped. Alternatively, the *joint-pairs* modifier can

be used, which will analyse all possible pairs of available variables. It can be combined with the *analyse* modifier to define a more specific selection of variables that should be used for this.

Interaction analysis

The gene-level regression model can also be used to perform interaction analyses, either between two gene sets or between a gene set and a continuous gene properties (interaction analysis between two gene properties is not currently available). In the context of gene-level analysis, an interaction effect can be interpreted as the combined involvement of two gene sets or gene properties, rather than such gene sets or properties having an effect on their own.

For the interaction between two gene sets, the interaction term is defined as the set of genes shared by the two individual sets. If there is a positive interaction, this would suggest that it is specifically the combination of the two properties that plays a role in the genetics of the phenotype. In some cases this will be in addition to main effects of those gene sets though in other cases there might only be an interaction, with any marginal effects of those gene sets caused by this interaction effect. If there a significant negative interaction between gene sets, this suggests that one of the gene sets is involved, but only or primarily in the absence of the second gene set. In this case the main effect in the interaction model of one of the two gene sets will likely show a stronger association compared to its marginal association.

Interactions between a gene set and a continuous gene property are interpreted in largely the same way. If there is a positive interaction, this indicates that the effect of the gene set is stronger for genes that have a higher score on the gene property (for example, the role of a particular pathway may be specific to genes in that pathway that are more strongly brain-expressed). If negative, the gene set effect is stronger for genes that have a lower score. Conversely, interactions can also be interpreted as showing that the effect of the gene property is stronger (if positive) or weaker (if negative) for genes in the gene set, compared to other genes. For the interaction analysis, the interaction term is defined in terms of the gene property centred on its mean value for the gene set. As such, the main effect of the gene set in the interaction model corresponds to the difference in genetic association between genes inside and outside the gene set at that set-specific mean for the gene property.

Four different options are available to run interaction analysis. These cannot be combined with each other or with either of the 'joint' modifiers, but can be combined with all the 'conditional' modifiers (however, interaction models cannot contain any variables also specified to be conditioned on). The most general option is the *interaction* modifier, which is used to specify a file defining the interactions to be analysed. Each line in the file should contain two variable names, for which the interaction will be analysed. A second option is *interaction-pairs*. This will take all possible (and valid) pairs of available variables and analyse their interactions. This can be combined with the *analyse* modifier to define a more specific selection of variables between which the interactions should be computed.

The two other options are *interaction-each* and *interaction-all*. For both of these a list of variables should be specified, either directly or using a file in the same way as the 'condition' modifiers (see above). Interactions with all other available variables will then be computed; the *analyse* modifier can again be used to limit which variables are used for this. With the *interaction-each* option, the interactions will be analysed separately: for each remaining available variable, the interactions with the variables specified for the modifier will be analysed in turn. By contrast, with the *interaction-all* modifier, for each remaining available variable the interactions with all be analysed simultaneously in a single model. This can therefore be used to analyse models containing multiple interactions.

In addition to the main interaction modifiers, an additional *condition-interaction* option is also available. This can be used to condition all analyses on specific interactions (and corresponding main effects), and like the other condition modifiers can be combined with any other model specification options. The variables to be used are also specified in the same way, either as a comma-separated list or via a file. This list is broken down into pairs of two consecutive variable names, for which interactions are conditioned on. As such, the number of variables specified for this modifier must always be even.

For the interaction model, analysis can only performed if a valid interaction term can be defined. For interactions between gene sets the interaction term is the set of genes shared by both sets, and as such there must be at least some overlap between the sets. Moreover, neither gene set can be completely contained by the other, as in that case the contained gene set would be identical to the interaction term and therefore completely collinear with it. To deal with this, for set by set interactions a minimum size requirement is imposed on both the overlap between the gene sets as well as the portion of each gene set not overlapping with each other. The minimum size is defined both in an absolute number of genes, as well as a proportion of the total size of each gene set. This can be set using the *interaction-ss-size* modifier, which must be provided two values: the minimum number of genes, followed by the minimum proportion of the gene set. The default setting for these parameters is 25, 0.1.

For gene set by continuous gene property interactions the interaction term is in principle always defined, but in practice the interactions can be very sensitive to outlier effects if the gene set is small. As such, a minimum and maximum size are imposed via the *interaction-sc-size* modifier. This takes a single value setting the minimum size a gene set must have for interactions with gene properties to be considered valid. The maximum size is then set to the total number of genes used in the analysis minus this number. By default, this parameter is set to 100.

For the interaction analysis, the testing direction for the interaction terms can be set independently of the testing direction for marginal and main effects. This can be done using the *direction-interaction* modifier, or the *direction-interaction-ss* and *direction-interaction-sc* modifiers, to set the direction for both types at ones or separately for set by set and set by covariate interactions (see above for valid values that can be used). The default testing directions are one-sided, positive for set by set interactions, and two-sided for set by covariate interactions.

When analysing interactions, if a significant interaction is detected (after multiple testing correction, for the number of models analysed), additional per-gene output for those interactions is written to a .gsa.inter.genes.out file. In addition, per-gene output for all genes in the analysis will then be provided in a .gsa.genes.out file.

Output

Primary output for the gene-level analysis is written to a file with the extension **.gsa.out**. Depending on settings and analysis results, more output may be provided in additional files. If self-contained gene-set analysis is enabled, results for this analysis are written to a separate **.gsa.self.out** file. When the abbreviate option is set to file mode and any of the variable names exceeds the maximum length, a **.gsa.abbr.out** file is created containing an index of all the abbreviations and the corresponding full variable names.

When performing analysis of single-variable models (not counting variables specified via any of the --model *condition* modifiers), per-gene output is written to a **.gsa.sets.genes.out** file for each significant gene set (after multiple testing correction). Similarly, when analysing interactions, per-gene output for significant interactions is written to a **.gsa.inter.genes.out** file. When either of these files is created or when '--settings gene-info' is set, an additional **.gsa.genes.out** file with per-gene output for all genes used in the analyses is also created.

The **.gsa.out** file will include the following output. General parameters and settings for the analysis are included at the top of the file for reference:
- **MEAN_SAMPLE_SIZE**: the sample size of the data the gene associations were based on (averaged across genes)
- **TOTAL_GENES**: the total number of genes included in the analysis
- **TEST_DIRECTION:** the testing direction used for the different types of parameters

- **CONDITIONED_INTERNAL/RESIDUALIZED/HIDDEN/VARIABLES**: the internal covariates and external variables which the analysis was conditioned on. Variables listed under CONDITIONED_VARIABLES will also have individual output for each analysed model in the file.

Output columns per variable:
- **VARIABLE**: name of the gene set, gene covariate or interaction. Names in this column are capped at 30 (by default) characters to keep the output file more readable
- **MODEL**: for models containing multiple variables, indicates which model the results on that row is part of; all output for each analysed model is always on consecutive rows in the file
- **TYPE:** denotes the type of variable, either SET or COVAR for normal gene sets and gene covariates provided in the input files; or INTER-SS or INTER-SC, for internally created interaction terms (set by set and set by covariate, respectively)
- **TERM:** for models containing interactions, provides a short-hand indication of which model term the row correspond to. Main effects are each assigned a letter, interactions effects are denoted as the product of the main effects (by their assigned letters) they are based on
- **NGENES**: the number of genes in the data that are in the set (for gene sets and set by covariate interactions), that are in the interaction set (for set by set interactions), or for which non-missing values were available (for gene covariates)
- **BETA**: the regression coefficient of the variable
- **BETA_STD**: the semi-standardized regression coefficient, corresponding to the predicted change in Z-value given a change of one standard deviation in the predictor gene set / gene covariate (ie. BETA divided by the variable's standard deviation)
- **SE**: the standard error of the regression coefficient
- **P**: p-value for the parameter / variable
- **FULL_NAME**: the full variable name; only included if the variable names exceed the maximum length for the **VARIABLE** column (and the *abbreviate* option is not set to file mode)

The **.gsa.self.out** file with results for self-contained gene-set analysis will contained the same **MEAN_SAMPLE_SIZE** parameter as the .gsa.out file (see above), and contains the following output columns per gene set:
- **GENE_SET:** name of the gene set. Names in this column are capped at 30 (by default) characters to keep the output file more readable
- **NGENES:** number of genes in the gene set
- **MU:** the estimated mean association for the gene set
- **P:** the self-contained gene-set p-value
- **FULL_NAME**: the full variable name; only included if the variable names exceed the maximum length for the **VARIABLE** column (and the *abbreviate* option is not set to file mode)

The **.gsa.genes.out**, **.gsa.sets.genes.out** and **.gsa.inter.genes.out** files contain largely the same columns as the normal **.genes.out** file created by the gene analysis (see description of columns there). Note that the Z-scores shown in these files are the values used in the analysis, with the truncation applied for the gene-level analysis (the corresponding p-values have not been truncated) and with internal covariates and residualizing covariates (specified using the *correct* and *condition-residualize* modifiers of --model) regressed out. Only additional columns not present in .genes.out are listed below.

General parameters included in the files are **ALPHA** and **NUMBER_OF_TESTS** for **.gsa.sets.genes.out** and **.gsa.inter.genes.out** denoting the nominal significance threshold and number of tests used to determine significance for the results shown in the files. For **.gsa.genes.out** the same

CONDITIONED_INTERNAL/RESIDUALIZED/HIDDEN/VARIABLES parameters as printed in the **.gsa.out** file is included.

In the **.gsa.sets.genes.out** and **.gsa.inter.genes.out** files, for each result the name, number of genes and p-value are also provided for reference. Output for all genes in that gene set or interaction term are then provided. For set by set interactions, output is given for all genes in either gene set, with the SET1 and SET2 columns indicating which gene belongs in which gene set. An additional unique tag (eg. '_SET1_') is added such that information for a single result can be more easily extracted using for example the grep utility.

Additional output columns per gene:
- **ZFITTED_BASE** and **ZRESID_BASE**: the fitted values and residual values for the base model, ie. the model including only the intercept and all conditioned-on variables. These are included in **.gsa.genes.out**, and in **.gsa.inter.genes.out** for set by covariate interactions, to be used for post-hoc outlier checks
- **SET1** and **SET2**: in **.gsa.inter.genes.out**, for set by set interactions. Indicate which genes belong to which of the two sets involved in the interaction
- **COVAR:** in **.gsa.inter.genes.out**, for set by covariate interactions. These are centred and scaled values for the gene covariate involved in the interaction, to be used for post-hoc outlier checks

Appendix A: Input files

This appendix describes the requirements for files used as input in MAGMA, ordered by the flag and modifier where they are specified. Only external input files are listed here, MAGMA output files later used as input are already described in the sections above (as needed).

Two commonly used types of files are referred to as the **formatted data file** and the **list file**. The **formatted data file** is a structured plain text file with the same number of values on each row in the file, separated by whitespace. These are usually allowed to have an optional header row with column names. The **list file** is a simple text file containing a list of values. The file is allowed to contain multiple values per row, but only the first value in each row is actually used. A **.bim** file always refers to the .bim file of a binary PLINK data set.

Unless otherwise specified, files are expected to be plain text with values separated by whitespace (ie. space or tab characters, with multiple consecutive whitespace characters counting as a single delimiter). Non-numeric values (eg. IDs, column names, etc.) should not start with a # character.

A small example of a **formatted data file** is shown below, in this case a file containing SNP p-values from a GWAS, for use with the --pval flag.

```
SNP       CHR     BP        P        NOBS
rs1345    2       100123    0.01     1087
rs18667   3       30566921  0.5611   1100
rs145     16      9992021   0.173    1100
```

--annotate *filter*
Requires a **list file**, containing SNP IDs.
Alternatively, a **.bim** file can be used as well.

--bfile
Requires a binary PLINK format data set, consisting of a .bed, .bim and .fam trio of files. Please consult the PLINK or PLINK 2 website for more details (eg. https://www.cog-genomics.org/plink2/input#bed)

--bfile *synonyms*
Requires a plain text file, with rows corresponding to a full set of synonymous SNP IDs.
If a SNP ID is synonymous with multiple other IDs, all these IDs should be listed on the same row. No ID is allowed to occur more than once in the file. For using the 'drop-dup' and 'skip-dup' options for the *synonym-dup* modifier, it is recommended that the IDs in each row have a meaningful order, listing them in decreasing order of preference in case of clashes. An optional tag specifying a common prefix for the SNP IDs can be added at the top of the file, such as:

```
# PREFIX = rs
```

If this tag is included, the specified prefix will be appended to all IDs as they are read from the file.

--covar *file*
Requires a **formatted data file**, with rows corresponding to individuals; a header is allowed.
The file does not need to include all individuals in the genotype data. The first two columns must contain the FID and IID values, corresponding to those in the genotype data. Variables must be numeric, with missing values coded as NA.

--gene-covar *file*
Requires a **formatted data file**, with rows corresponding to genes; a header is required.
The file does not need to include all genes in the genotype data. The first column must contain the gene ID values, using the same nomenclature as was used in the gene annotation. Variables must be numeric, with missing values coded as NA.

--gene-loc
Requires a **formatted data file**, with rows corresponding to genes; a header is not allowed.
The file must have four columns containing the gene ID, chromosome, start position and stop position, in that order. It may have a fifth column containing the strand, coded as + for the positive/sense strand and - for the negative/antisense strand.

--gene-settings *snp-exclude* and *snp-include*
Requires a **list file**, containing SNP IDs.
Alternatively, a **.bim** file can be used as well.

--meta *genes-file* and *raw-file*
Requires a plain text file, with rows corresponding to data files.
The first value should be the name of the input file. The .genes.out (for *genes-file*) and .genes.raw (for *raw-file*) suffixes can be omitted if the *prefix* modifier on --meta is set. Three additional sample size override values can be included, to be used instead of the sample sizes embedded in the input files themselves during the meta-analysis. The first override value (if set) is used for all genes; the second (if set) for X and Y chromosome genes, instead of the first override value; and the third (if set) for the Y chromosome, instead of the first or second override value.
      An example of a settings file is provided below for illustration. In this example the sample size for cohort 1 is set to 1000 for all X and Y chromosome genes and 2000 for all others; the original sample sizes specified in the .genes.out file are used for cohort 2; and for cohort 3 the sample size is set to 1500 for all genes.

```
cohort1.genes.out    2000      1000
cohort2.genes.out
cohort3.genes.out    1500
```

--model *condition*, *condition-hide*, *condition-residualize*, *condition-interaction*, *interaction-each*, *interaction-all* and *analyse* (for *file* mode)
Requires a **list file**, can contain multiple variable names per row (each will be used)

--model *joint* and *interaction*
Requires a **list file** which will be read in per row, with each row corresponding to a single model specification. For the *interaction* modifier, each row must contain exactly two variable names.

--pheno *file*
Requires a **formatted data file**, with rows corresponding to individuals; a header is allowed.
The file does not need to include all individuals in the genotype data. The first two columns must contain the FID and IID values, corresponding to those in the genotype data. Variables must be numeric, with missing values coded as NA. For binary phenotypes, missing values can also be coded as -9 or 0 (a phenotype is considered binary if it only contains the values -9, 0, NA, 1 and 2).

--pval
Requires a **formatted data file**, with rows corresponding to SNPs; a header is allowed.
The file does not need to include all SNPs in the data, and cannot contain duplicate SNP IDs. The file must contain a column containing SNP IDs and a column containing p-values; missing p-values can be

coded as NA. The file is allowed to contain any number of additional columns. By default the SNP and P columns will be used if a header is present, or the first and second columns if it is not (other columns can be specified when using --pval). Optionally, a column containing the sample size used per SNP can be included in the analysis. This has no default value, and is only used when explicitly specified while running the analysis.

--set-annot (row-based format)
Requires a plain text file, with rows corresponding to gene sets.
The first value on each row is the gene-set name, subsequent values are gene IDs, using the same nomenclature as was used in the gene annotation. Gene IDs that cannot be matched to those present in the data are discarded, therefore extraneous values (eg. the second value being used for a hyperlink) can be left in, as long they cannot be confused for a legitimate gene ID.
An example of a small row-based gene-set file is given below.

```
SET1  1546   90122   386    6001   23019
SET2  12345  6001    45109
```

--set-annot (column-based format)
Requires a **formatted data file**, with rows corresponding to gene ID - gene-set pairs; a header is not allowed.
Gene IDs must use the same nomenclature as was used in the gene annotation. The file is allowed to contain any number of other columns aside from the gene ID and gene-set name columns.
An example of a small column-based gene-set file is given below. It would be used by adding 'col=2,1' to the --set-annot flag. It is equivalent to the above example of a row-based gene-set file.

```
SET1   1546
SET1   90122
SET1   386
SET1   6001
SET1   23019
SET2   12345
SET2   6001
SET2   45109
```

--settings *gene-include* and *gene-exclude*
Requires a **list file**, containing gene names.

--snp-loc
Requires a **formatted data file**, with rows corresponding to SNPs; a header is not allowed.
The file must have three columns containing the SNP ID, chromosome and base pair position, in that order. Allowed chromosome codes are 1-24, X and Y (where 23 and 24 correspond to X and Y respectively).
Alternatively, a **.bim** file can be used as well.