MAGMA manual (version 1.06)

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 (optionally), 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].sets.out with competitive gene-set analysis results. To obtain self-contained gene-set p-values as well, use ‘--set-annot [SET_FILE] self-contained’ instead.

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 Including gender as a covariate Including a gene-covariate interaction term in the analysis	Yes
--pheno	Specifying an alternate phenotype from an external file	Yes
--meta	Performing a meta-analysis on gene analysis results Preparing a meta-analysis for gene-level analysis	No
--batch / --merge	Performing analysis in smaller parts, for distributed processing	No
--gene-settings	Specifying various settings affecting: - Permutation behaviour (if applicable) - Internal SNP quality control - SNP inclusion/exclusion lists	No
--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.

It should be noted that (at present) the specified sample size does not directly affect the gene analysis itself. It is however included for reference in output files, serves as default weight when performing meta-analysis, and will be used to correct for differences in sample size between genes in gene-level analysis (if any). The latter applies in particular to analyses based on results from a SNP meta-analysis, where the total sample size can vary considerably across SNPs. In this case, using the *ncol* modifier is strongly recommended, if such information is available.

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 -log(SNP p-value) as test statistic Equivalent to VEGAS, PLINK --set model Equivalent to SKAT model using inverse variance weights	Very 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	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

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 highly similar 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 25; 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 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 also 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).

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-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 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).

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. Note that SNPs 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 $H_0\colon \beta_s = 0$ against the one-sided alternative $H_A\colon \beta_s > 0$. 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.

Because of the general regression framework a wide range of extensions to the gene-level regression model is possible, and is presently being implemented. However, some models not yet officially implemented in MAGMA can be achieved by judicious coding of input variables, and a brief comparison of different models is given in Appendix B. Additional model options include the 'relative' model, testing whether the (conditional) mean association in one gene set is larger than that in another gene set; and the interaction model, testing whether the co-occurrence of two properties (as gene sets or gene covariates) interact to produce significant additional genetic association with a phenotype.

MAGMA also offers the option to perform self-contained gene-set analysis, although it is turned off by default. The self-contained analysis is implemented as a Z-test, testing whether the mean association with the phenotype of genes in the gene set is greater than zero (note that it does not condition on any variables). In effect this is an omnibus gene test, testing whether at least one of the genes in the gene set is associated with the phenotype. Importantly, this means that no biological or functional conclusions can be drawn about the property that defines the gene set. However, the test can be useful when considering a particular group of genes of which none are individually significant in the gene analysis. A significant result from a self-contained gene-set analysis would in this case indicate that this group does contain some associated genes, even if it cannot be determined which ones.

Gene-set analysis

Running a (competitive) gene-set analysis is done by using --gene-results flag to specify a gene analysis intermediate results file (ie. with .genes.raw extension), and the --set-annot flag to specify a file containing gene-set definitions. The basic gene-set analysis command would therefore look like this:

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

The gene-set definition file is expected to be in row-based format (with whitespace separated values), with each row corresponding to a gene set. The first value on each row is interpreted as the name of the gene set, and subsequent values as genes belonging to that gene set. Any gene ID or gene name nomenclature can be used (though of course it must match the one you used when running the annotation). The only restriction is that gene and gene-set names cannot start with a # character, and cannot contain whitespace. Note that some public gene-set databases use a row-based format, but with additional other values (such as a hyperlink) between the gene-set name and list of genes. In practice these can usually be left in; as long as the additional values cannot be interpreted as a gene ID or name from the nomenclature used, MAGMA will simply interpret the value as a gene not present in the data and discard it.

As an alternative MAGMA can also accept column-based gene-set annotation files, where each row corresponds to a gene - gene-set pair. Such annotation files can be used by adding the *col* modifier to the --set-annot flag. This modifier takes two values, the indices of respectively the gene ID column and the gene-set name column (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.

To request a self-contained gene-set analysis, add the *self-contained* modifier to the --set-annot flag. An additional column with self-contained p-values will be added to the output file. Note that the self-contained gene-set analysis is generally not affected by other settings for the gene-level analysis, and does not condition on any covariates.

If desired, the (competitive) gene-set analysis can be switched from a one-sided to a two-sided test, by adding the *twosided* modifier to --set-annot. It can also be switched to a one-sided test in the other direction by adding 'onesided=smaller' (the default setting is 'onesided=greater'). The latter is in effect a test of the complement of the gene set (ie. the gene set that contains all the genes not in the present gene set, and only those genes). Changing the testing direction is generally not recommended.

Gene property analysis
Running a gene property analysis is done by again using the --gene-results flag to specify a gene analysis intermediate results file (ie. with .genes.raw extension), and the --gene-covar flag to specify a file with continuous gene-level predictor variables. The basic gene property analysis command would therefore look like this:

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

The gene covariate file is organised as a covariate file, with the gene ID in the first column. Missing values can be coded as NA. Each of the covariates in the file is analysed (though they are still analysed one at a time), unless the *include* or *exclude* modifier is added to specify which variables (not) to use. These take a comma-separated list of variable names (if a header is present) or indices, as well as 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.

To change the statistical test from the default two-sided test, the *onesided* modifier can be used. To perform a test for positive association, whether greater values on the gene-level predictor coincide with stronger association with the phenotype, add either 'onesided' or 'onesided=greater'. For the opposite test, add 'onesided=smaller'.

When performing a gene property analysis, the analysis will be restricted to only genes that are present in both the data as well as the gene covariate file (NOTE: other genes are removed altogether, this will also affect gene-set analyses being run at the same time). If the 'filter=0' option is set all genes present in the data but absent from the gene covariate file will be set to missing on all covariates instead. However, if this results in the maximum missingness threshold being exceeded, the analysis will be aborted.

The maximum allowed missingness proportion is set at 0.05 by default, but can be changed using the *max-miss* modifier for the --gene-covar flag, up to 'max-miss=0.2'. Gene covariates with missingness exceeding this threshold will be dropped from the analysis. For the remaining gene covariates, missing values handled using a simple imputation procedure, setting the missing values to a fixed value. By default the median of the non-missing values for that gene covariate will be used. This can be changed to the mean instead by adding 'impute-value=mean' to the --gene-covar flag, or to a specific value (eg. 'impute-value=0', to set all missing values to 0).

Conditional analysis
Any gene-level analysis can be conditioned on user-specified gene sets, gene covariates or both, by adding the *condition* modifier to the --set-annot and/or --gene-covar flags followed by a comma-separated list of gene set names (for --set-annot) or gene covariate names (if a header is present) or indices (for --gene-covar) to condition the analyses on. For --gene-covar the *condition* modifier also accepts ranges of variables, in the same way as the *include* and *exclude* modifiers.

It is not uncommon for gene set names to contain special characters like a quote character, an ampersand or a comma, which makes them difficult to specify directly on the command line. For this reason, for --set-annot the *condition-file* modifier can also be used to specify a file listing the gene sets to condition on (one gene set per line).

When the --set-annot and --gene-covar flag are both set, for --gene-covar the *condition-only* modifier can also be used instead of the *condition* modifier. This will suppress gene property analysis for gene covariates that are not conditioned on, only performing competitive gene-set analyses conditioned on the gene covariates specified by the *condition-only* modifier.

Empirical multiple testing correction

The p-values generated by MAGMA are not corrected for multiple testing, and a procedure such as Bonferroni correction will need to be applied manually by the user after the analysis. Since Bonferroni correction can be somewhat conservative when variables are correlated (eg. when gene sets are strongly overlapping), MAGMA provides an optional empirical multiple testing correction, which uses a permutation procedure to obtain p-values corrected for multiple testing. This can be turned on using the *fwer* modifier of the --model flag, using 'fwer=[NPERM]' to manually specify the number of permutations to use or simply 'fwer' to use the default 10,000 permutations.

When enabling the empirical correction, additional corrected p-values are added to the output files. The P_CORR is the p-value corrected for all gene sets (for .sets.out) or all gene covariates (for .gcov.out) in the analysis. If gene-set analysis and gene property analysis were performed, an additional P_CORR_GLOBAL column is added, containing p-values corrected for all gene sets and all gene covariates. The estimated significance thresholds at the α level (default of 0.05) are also shown in the log file.

NOTE: as with all permutation procedures, the corrected p-values and estimated significance threshold are subject to uncertainty, especially at lower numbers of permutations. It is therefore advised to use at least the default of 10,000 permutations, as well as to rerun the analysis with a larger number of permutations if any corrected p-value is very close to α.

Automatic corrections, filtering, truncation and other settings

Both competitive gene-set analysis and gene property analysis are corrected using a conditional model for gene size, gene density, inverse gene MAC and (if applicable) differential sample size (as well as the log of those values). These corrections are intended to protect against confounding due to these variables. Although strongly discouraged, they can be turned off using the *correct* modifier of the --model flag.

Adding 'correct=all' (the default setting) or 'correct=none' turns all corrections on or off. The *correct* modifier can also be set to either 'include' or 'exclude' followed by a list of internal variables. When set to 'include', only the listed variables (and their log values) are corrected for; when set to 'exclude' all but the listed variables (and their log values) are corrected for. The available internal variables are designated as `size`, `density`, `mac` and `sampsize`.

Gene-level analysis can be restricted to a subset of genes by using the *filter* modifier of the --settings flag. This takes a single value specifying the name of a filter file, and will include only genes listed in the first column of this file in the analysis.

To protect against outliers unduly influencing the results, the gene association Z-values are truncated. The minimum and maximum values are specified as a number of standard deviations below and above the mean, and all Z-values exceeding those boundaries are set to the exceeded boundary value. The boundary values can be changed using the *outlier* modifier for the --settings flag, which takes either one or two values specifying the number of standard deviations. If only one value is given the bounds are symmetrical, if two values are given the first is used for the lower bound and the second for the upper bound. The default is equivalent to '--settings outlier=3,6'.

To change the α level for determining significance from its default value of 0.05, setting the *alpha* modifier on the --model flag to the desired value. The α level does not affect the results directly, but in some cases additional output may be generated for significant gene sets or gene covariates.

The --seed [SEED] flag can again 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.

Output

A .setgenes.out file is created when there are significant gene sets (after multiple testing correction) in the competitive gene-set analysis, containing output for the individual genes in the significant gene sets. The columns are a subset of those also found in the .genes.out from a gene analysis, plus an additional tag for each gene set (eg. '_SET1_') such that information for individual sets can easily be extracted using utilities like grep.

The .sets.out and .gcov.out files will include the following output. General properties of the analysis are included at the top of the file:

- **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
- **CONDITIONED_INTERNAL/SETS/COVAR**: the internal covariates, gene sets and gene covariates the analysis was conditioned on
- When using empirical multiple testing correction:
    - **ALPHA**: the alpha level set for significance
    - **FWER_PERM**: the number of permutations used for the empirical multiple testing correction
    - **FWER_THRESH_ALPHA**: the significance threshold for the chosen alpha corrected for multiple testing for the gene sets/gene covariates in the output file
    - **FWER_THRESH_ALPHA_GLOBAL**: the significance threshold for the chosen alpha corrected for multiple testing for all gene sets and gene covariates analysed (if applicable)

Output per gene set / gene covariate:

- **SET/COVAR**: name of the gene set / gene covariate; gene-set names in this column are capped at 25 characters to keep the output file more readable
- **NGENES**: the number of genes in the data that are in the gene set (.sets.out only)
- **OBS_GENES**: the number of genes in the data that have a non-missing value on the gene covariate (.gcov.out only)
- **BETA**: the regression coefficient of the gene set / gene covariate
- **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
- **SE**: the standard error of the regression coefficient
- **P**: the competitive gene-set p-value / gene covariate p-value
- **P_CORR**: the p-value empirically corrected for multiple testing for all the gene sets / gene covariates in the output file (if enabled)
- **P_CORR_GLOBAL**: the p-value empirically corrected for multiple testing for all the gene sets and gene covariates analysed (if enabled and applicable)
- **SELF_P**: the self-contained gene-set p-value (.sets.out only, if enabled)
- **FULL_NAME**: the full gene set name (.sets.out only, and only if any gene set names in the SET column were abbreviated)

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 specifiying 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 allowed.
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
```

--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
```

--set-annot *condition-file*
Requires a **list file**, containing gene-set names.

--settings *filter*
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.

Appendix B: Overview of competitive gene-set analysis models

This appendix contains an overview of different variations on the competitive gene-set analysis models, as well as details on how to run them. Not all these models are directly implemented in MAGMA, but due to the flexible linear regression structure can straightforwardly be run regardless by choosing the right transformations of input variables. See also the Overview section of *Gene-level Analysis* for additional for more information on the general structure and implementation of gene-set analysis in MAGMA.

All analyses are structured as a linear regression model on gene-level data (ie. with genes serving as data points, as the rows in the data matrix) with outcome variable Z, the Z-score (level of association) for each gene. Gene sets G_i are binary indicator variables, coded 1 for genes in the gene set and 0 otherwise. The target set is always G_1 unless otherwise specified.

The residual vector ε is assumed to have a multivariate normal distribution with variance parameter σ_e^2. The residual correlation matrix is set to the gene-gene correlations estimated during the gene analysis step, to account for LD between genes. All models are also automatically conditioned on a number of additional covariates (gene size, density, inverse MAC, per-gene sample size, plus the log value of each), but for clarity of notation these are omitted from the formulas below.

Note that the models below can all also be generalized to continuous covariates (eg. tissue-specific gene expression levels). Implementation and interpretation is largely the same. This generalization is omitted here for clarity of exposition.

Basic competitive gene-set analysis
Model: $Z = \beta_0 + \beta_1 G_1 + \varepsilon$
Aim: Test whether β_1 is greater than 0.

The intercept β_0 represents the baseline level of association, and β_1 the additional association specific to the gene set. As such, the value of β_1 reflects the degree to which the genes in the gene set exhibit stronger associations with the phenotype than other genes. The analysis thus tests whether the mean association of genes in the gene set is greater than that of other genes.

Implementation: this is the standard analysis run when setting the --set-annot flag.

Conditional gene-set analysis
Model: $Z = \beta_0 + \beta_1 G_1 + \beta_2 G_2 + \varepsilon$
Aim: Test whether β_1 is greater than 0.

β_1 represents the association specific to gene set 1, conditional on any effects specific to gene set 2 (and vice versa). If there is little or no overlap between the two gene sets this will be almost identical to the basic competitive gene-set analysis. If the gene sets do overlap, the (implicitly estimated) effect shared by the two gene sets is divided over the two gene sets.

If gene set 1 was significant in the basic analysis but not after condition on another gene set (or a covariate), this suggests that the effect originally observed in gene set 1 was probably (largely) due to the other gene set (or covariate); ie. the other gene set (or covariate) is a confounding variable.

A special cases of this is when gene set 1 is (almost entirely) a subset of gene set 2. For example, gene set 1 reflects a particular brain pathway and gene set 2 a general set of brain-expressed genes. If gene set 1 now ceases to be significant when conditioned on gene set 2, this suggests that that particular brain pathway likely does not play a role in the phenotype after all; it was only significant initially because the phenotype has a lot of brain involvement, and the gene set contains a lot of brain-expressed genes.

Conversely, gene set 2 can also be (almost entirely) a subset of gene set 1. For example, gene set 1 now reflects a general set of brain-expressed genes, and gene set 2 a particular brain pathway. If gene set 1 ceases to be significant in this case when conditioned on gene set 2, this would suggest that there isn't any general brain involvement for that phenotype. Rather, it is that particular brain pathway that is actually involved in the phenotype, because gene set 2 contains that pathway it becomes significant as well.

Implementation: adding the *condition* or *condition-file* modifier to --set-annot.

Gene-set interaction analysis

Model: $Z = \beta_0 + \beta_1 G_1 + \beta_2 G_2 + \beta_{12} G_1 G_2 + \varepsilon$

Aim: Test whether β_{12} is greater than 0.

The $G_1 G_2$ term is simply the product of the two gene set indicators, ie. it is 1 for genes present in both gene sets and 0 otherwise. This analysis allows testing for specificity. For example, if gene set 1 represents "cell metabolism genes" and gene set 2 represents "liver-expressed genes", then the interaction is the subset of "liver-expressed cell metabolism genes", and the analysis tests whether such genes have associations with the phenotype above and beyond the main effects of "cell metabolism" and "liver-expressed".

Compare the basic model $Z = \beta_0 + \beta_{12} G_1 G_2 + \varepsilon$ (eg. when the overlapping part is manually entered as a gene set). Although similar, this only tests whether "liver-expressed cell metabolism genes" are more strongly associated with the phenotype than other genes; if for example the "liver-expressed genes" set G_2 has a significant effect itself, this subset is likely to become significant as well even though it is not specific to that subset of cell metabolism genes. By using the full interaction analysis however, this specificity can be achieved.

Note that the analysis is only meaningful if there is incomplete overlap between the two gene sets. If there is no overlap the interaction term is always 0 and collinear with the intercept; on the other hand, if one gene set is a subset of the other the interaction term is identical to (and collinear with) that gene set. In both cases the model is unstable because of the collinearity, and conceptually identical to a conditional analysis with the interaction term removed.

One further consideration specific to this model is that depending on the research question it may be of interest to change the direction of the significance test, or perform a two-sided test. A scenario with β_{12} smaller than 0 would occur if genes present in both gene sets are in fact significantly less associated with the phenotype than genes that occur in just one or the other gene set. Although probably less likely to happen, detecting this scenario may be of interest as well.

Implementation: Define a new gene set corresponding to the interaction term $G_1 G_2$, consisting of genes present in both gene sets. Now run a conditional gene set analysis on this interaction gene set, conditioning on the original two gene sets G_1 and G_2. The parameter estimates and p-value in the output file are those for the β_{12} parameter and corresponding significance test.

Relative gene-set analysis
Model: $Z = \beta_0 + \beta_1 G_1 + \beta_2 G_2 + \varepsilon$
Aim: Test whether β_1 is greater than β_2.

The aim of this analysis is to test whether the level of association of genes in one gene set is stronger than that in another gene set. This is generally only meaningful if both gene sets are individually significant, but can in that case provide more insight in their relative level of association. It is also primarily of interest for gene sets that do not strongly overlap, since considerably overlapping gene sets would be expected to have similar levels of association simply due to containing many of the same genes. In that case the conditional gene-set analysis will tend to be more informative.

Some care should be taken when interpreting the results of this kind of analysis. Although the difference may represent an underlying genetic and/or biological difference, it may also have been caused simply by differences in the way the two sets are constructed. For example, when comparing a very well characterized known biological pathway to a noisy Gene Ontology category, the pathway showing significantly stronger effect may well be due to the GO category simply containing a lot of irrelevant, misclassified genes (and omitting other, relevant genes not yet annotated to that category).

Statistically speaking, testing $\beta_1 > \beta_2$ is equivalent to testing $\beta_{\text{diff}} = \beta_1 - \beta_2 > 0$. The core of the model can be rewritten to accommodate this as follows: $\beta_1 G_1 + \beta_2 G_2 = \beta_1 G_1 - \beta_2 G_1 + \beta_2 G_1 + \beta_2 G_2 = (\beta_1 - \beta_2) G_1 + \beta_2 (G_1 + G_2) = \beta_{\text{diff}} G_1 + \beta_2 G_{1+2}$, with $G_{1+2} = G_1 + G_2$. Consequently, the desired test can be performed by creating the variable G_{1+2} and running a conditional gene-set analysis of G_1 conditioned on G_{1+2}. In practice it will usually be of interest to perform a two-sided test in this analysis rather than the default, performing a symmetrical test of whether $\beta_1 \neq \beta_2$.

Note that for genes present in both gene sets, G_{1+2} will have a value of 2. As such, those genes will have a stronger effect on the genes present in only one of the gene sets. This does make sense as they affect both β_1 and β_2, but an alternative would be to define a gene set $G_{1\text{or}2}$ containing all genes that are present in G_1 and G_2, then separately testing G_1 and G_2 conditioned on $G_{1\text{or}2}$. This does change the interpretation somewhat, however. The coefficient of $G_{1\text{or}2}$ reflects the mean association of genes in the two gene sets, and the coefficients for G_1 or G_2 will reflect how much the association in that gene set exceeds that shared mean.

Implementation: For testing the first variant of this model, G_{1+2} must be defined as a gene covariate rather than a gene set, since overlapping genes are scored as 2. To do so, read the .genes.out file corresponding to the .genes.raw file you will be using into a data processing program like R, and extract the GENEID variable. Add a new variable of the same length to that GENEID variable (ie. as a data frame in R). Set this to 2 for all genes present in both gene sets, to 1 for all genes present in only one of the gene sets, and to 0 for all other genes. Write these two variables as columns to a data file. Now perform a gene-set analysis of G_1, using the --gene-covar option to include the data file you just created and conditioning on the G_{1+2} variable in that file using the *condition* modifier of --gene-covar. Set the *twosided* modifier on --set-annot to make it a symmetrical test. The estimates and p-value in the output file will be those for the β_{diff} parameter and corresponding significance test.

For testing the second variant, create a new gene set to represent $G_{1\text{or}2}$, containing all genes in G_1 and G_2. Now perform a conditional gene-set analysis of G_1 and G_2, conditioned on the $G_{1\text{or}2}$ gene set.