snipar

Alexander Young

Apr 22, 2022

CONTENTS:

1 Tutorial **1**

1.1 Test data . . . 1
1.2 Inferring IBD between siblings . . . 1
1.3 Imputing missing parental genotypes . . . 2
1.4 Family based GWAS . . . 3
1.5 Correlations between effects . . . 4
1.6 Polygenic score analyses . . . 4

2 Guide **7**

3 Indices and tables **9**

CHAPTER

ONE

TUTORIAL

Tutorial on inferring IBD between siblings, imputing missing parental genotypes, and performing family based GWAS and polygenic score analyses. Before working through the tutorial, please first install the package and run the tests (see github).

1.1 Test data

To create a directory called 'example_data/' in the current directory and load the example data into it, use the command:

```
snipar_example_data.py --dest example_data
```

You can create the example data directory elsewhere by changing the –dest argument.

In the example_data/ directory, there is some example data. The file phenotype.txt is a simulated phenotype with direct, paternal, and maternal effects, where 80% of the phenotypic variance is explained by the combined direct, paternal and maternal effects of the SNPs; and the pairwise correlations between the direct, paternal, and maternal effects are 0.5. Please change your working directory to example_data/:

```
cd example_data
```

The genotype data has been simulated so that there are 3000 independent families, where 1000 have two siblings but no parents genotyped, 1000 have one parent genotyped and a 50% chance of having a genotyped sibling, and the final 1000 have both parents genotyped and a 50% chance of having a genotyped sibling. The example data includes genotype data formatted in both PLINK .bed format (chr_1.bed) and phased genotype data in .bgen format (chr_1.bgen with associated sample file chr_1.sample).

1.2 Inferring IBD between siblings

The first step is to infer the identity-by-descent segments shared between siblings. *snipar* contains a script, ibd.py, that employs a Hidden Markov Model (HMM) to infer the IBD segments for the sibling pairs. The per-SNP genotyping error probability will be inferred from parent-offspring pairs when available; alternatively, a genotyping error probability can be provided using the –p_error option. By default, SNPs with genotyping error rates greater than 0.01 will be filtered out, but this threshold can be changed with the –max_error argument. To infer the IBD segments from the genotype data in sample.bed,use the following command

```
ibd.py --bed chr_@ --king king.kin0 --agesex agesex.txt --out chr_@ --threads 4
--ld_out
```

This will output the IBD segments to a gzipped text file chr_1.ibd.segments.gz. Genotype files split over multiple chromosomes can be specified using '@' as a numerical wildcard character. In this example, –bed chr_@ instructs ibd.py to search for plink bed files chr_1.bed, chr_2.bed, ..., chr_22.bed, where each bed file contains SNPs from the numbered chromosome. In this case, only one bed file is in example_data/, chr_1.bed. If bed files for multiple

chromosomes are found, IBD will be inferred separately for each chromosome, with one output file per chromosome, with the chromosome number filling in the numerical wildcard in the –out argument. The *–king* argument requires the address of the relations (parent-offspring, sibling) inferred by KING by using the –related –degree 1 command, and the *–agesex* argument requires the address of a white-space separated text file with columns 'FID' (family ID), 'IID' (individual ID), 'age', 'sex' (coded as 'M' for male and 'F' for female).

The algorithm requires a genetic map to compute the probabilities of transitioning between different IBD states. If the genetic map positions (in cM) are provided in .bim file, the script will use these. Alternatively, the *–map* argument allows the user to specify a genetic map in the same format as used by SHAPEIT (https://mathgen.stats.ox.ac.uk/genetics_software/shapeit/shapeit.html#formats) an example of which is provided in genetic_map.txt.

If no genetic map is provided, then the deCODE sex-averaged map on Hg19 coordinates (Halldorsson, Bjarni V., et al. "Characterizing mutagenic effects of recombination through a sequence-level genetic map." Science 363.6425 (2019).), which is distributed as part of *snipar*, will be used.

The algorithm computes LD scores of SNPs in order to account for correlations between SNPs. The '–ld_out' argument writes the LD scores to file in the same format as LDSC (https://github.com/bulik/ldsc).

The user can also input a phased .bgen file. For example, to infer IBD from chr_1.bgen using the genetic map in sample.genetic_map.txt, use this command:

```
ibd.py --bgen chr_@ --king king.kin0 --agesex agesex.txt --out chr_@ --threads
4 --ld_out --map genetic_map.txt
```

If the user has a pedigree file with columns FID (family ID), IID (individual ID), FATHER_ID (father ID), MOTHER_ID (mother ID), they can input that instead of the *–king* and *–agesex* arguments. Missing IDs in the pedigree are denoted by 0. Siblings are inferred as individuals in the pedigree that share both parents. (Warning: MZ twins should be removed from the pedigree to avoid confusing them with full-siblings.) Using the example pedigree in sample.ped, you can infer IBD using this command:

```
ibd.py --bed chr_@ --pedigree pedigree.txt --map genetic_map.txt --out chr_@
--threads 4 --ld_out
```

The IBD inference can be performed on a smaller set of SNPs than will be imputed to save computation time. For example, IBD inference could be performed using SNPs from a genotyping array, and the imputation performed using all SNPs that have been imputed from a reference panel. For imputation from siblings, SNPs that fall outside of regions covered by the IBD segments will be imputed as missing values.

1.3 Imputing missing parental genotypes

To impute the missing parental genotypes without using phase information, type:

```
impute.py --ibd chr_@.ibd --bed chr_@ --king king.kin0 --agesex agesex.txt
--out chr_@ --threads 4
```

The script constructs a pedigree from the output of KING's relatedness inference (sample.king), and age and sex information (sample.agesex). The pedigree along with the IBD segments shared between siblings recorded in chr_1.ibd.segments.gz are used to impute missing parental genotypes from the observed sibling and parental genotypes in sample.bed. The imputed parental genotypes are in a HDF5 file sample.hdf5. The *–snipar_ibd* flag indicates the IBD segments are formatted as output by *snipar*.

If phased haplotypes are available in .bgen format, the imputation can use these as input, which improves the information gained by imputation in certain situations. To perform imputation from the phased .bgen file in example_data/, use the following command:

```
impute.py --ibd chr_@.ibd --bgen chr_@ --king king.kin0 --agesex agesex.txt
--out chr_@ --threads 4
```

It is necessary to provide the *–from_chr* and *–to_chr* arguments when imputing from a single .bgen file (not multiple .bgen files input with the numerical wildcard for chromosome number) since they often do not contain information on which chromosome the SNPs are located on, and it's necessary to match the IBD segments to the SNPs on the same chromosome.

To use IBD segments output by KING with the –ibdseg argument (sample.king.segments.gz), use the following command:

```
impute.py --ibd king --bgen chr_@ --king king.kin0 --agesex agesex.txt --out
chr_@ --threads 4 --ibd_is_king
```

As with the ibd.py script, the impute_runner.py script can use a user input pedigree (with the *–pedigree* argument) rather than the *–king* and *–agesex* arguments.

Note that if memory issues are encountered running the imputation, the –chunks argument can be used to read the SNPs into memory in smaller batches (of number equal to the argument given to –chunks).

1.4 Family based GWAS

To compute summary statistics for direct, paternal, and maternal effects for all SNPs in the .bed file, type:

```
gwas.py phenotype.txt --bed chr_@ --imp chr_@ --threads 4
```

This takes the observed genotypes in sample.bed and the imputed parental genotypes in sample.hdf5 and uses them to perform, for each SNP, a joint regression onto the proband's genotype, the father's (imputed) genotype, and the mother's (imputed) genotype. This is done using a linear mixedl model that models phenotypic correlations between siblings, where sibling relations in the pedigree are stored in the output of the imputation script: chr_1.hdf5. The 'family variance estimate' output is the phenotypic variance explained by mean differences between sibships, and the residual variance is the remaining phenotypic variance.

To use the .bgen file instead, type:

```
gwas.py phenotype.txt --bgen chr_@ --imp chr_@ --threads 4
```

The script outputs summary statistics in a gzipped text file: h2_quad_0.8.sumstats.gz. This file gives the chromosome, SNP id, position, alleles (A1, the allele that effects are given with respect to; and A2, the alternative allele), the frequency of the A1 allele, then summary statistics for each type of effect. For each effect, we give the effective N for each SNP; this differs from the actual N due to the fact that there are differing amounts of information for each type of effect, and due to relatedness in the sample. We give the effect estimate in the first column for each effect, the column 'effect_Beta', where 'effect' can be direct, paternal, etc; this is followed by the standard error, the Z-score, and the negative log10 P-value for a non-zero effect. Even if not directly estimated in the regression, we also output the average non-transmitted coefficient (NTC) estimate (estimate of the average of maternal NTC and paternal NTC), and the population effect estimate, which is equivalent to what is estimated by standard GWAS methods that regress phenotype onto genotype without control for parental genotypes. The final columns give the sampling correlations between the different effect estimates at that SNP.

In addition to the plain text output, the effects and their sampling variance-covariance matrices are output in h2_quad_0.8.sumstats.hdf5. The contents of the HDF5 file can be read into Python (using h5py) and R (using rhdf5) easily. The output contains different datasets:

1. *estimate*, the estimated SNP effect, where each row gives a SNP, and each column gives an effect
2. *bim*, equivalent to the bim file for plink, recording the information on each SNP
3. *estimate_cols*, gives the names of the effects estimate for each SNP: direct, paternal, maternal, etc.
4. *estimate_ses*, the standard errors for the effect estimates in *estimate*
5. *estimate_covariance*, 3 dimensional array with sampling variance-covariance matrices for each SNP's estimated effects, with SNPs indexed by the first axis

6. *freqs*, frequencies of the effect alleles
7. *sigma2*, maximum likelihood estimate of the residual variance in the null model
8. *tau*, maximum likelihood estimate of the ratio between the residual variance and family variance

Now we have estimated SNP specific summary statistics. To compare to the true effects, run

```
python estimate_sim_effects.py chr_1.sumstats.hdf5 phenotype.effects.txt
```

This should print estimates of the bias of the effect estimates.

The bias estimates for direct, paternal NTCs, maternal NTCs, and average NTCs should not be statistically significantly different from zero (with high probability). Population effects (which are estimated by univariate regression of individuals' phenotypes onto their genotypes – as in standard GWAS) here are biased estimates of direct effects, since population effects include both direct and indirect parental effects.

If the imputation has been performed from siblings alone, then the regression onto proband (focal, phenotyped individual), imputed paternal, and imputed maternal becomes collinear. This is because the imputation is the same for paternal and maternal genotypes. In this case, the regression should be performed onto proband and sum of imputed paternal and maternal genotypes. This can be achieved by providing the *–parsum* option to the script. The script can also estimate indirect sibling effects for each SNP by providing the *–fit_sib* option; however, this will reduce power for estimating other effects.

GWAS can also be performed without imputed parental genotypes. In this case, only probands with genotypes for both parents available will be used. In order to do this, one must provide a pedigree to gwas.py, as in:

```
gwas.py phenotype.txt --out trios_ --bgen chr_@ --pedigree pedigree.txt
--threads 4
```

1.5 Correlations between effects

snipar provides a script to compute correlations between direct and population effects and between direct effects and average NTCs. To compute these correlations from the effects estimated in this tutorial (output by gwas.py to h2_quad_0.8.sumstats.gz) using the LD scores computed by ibd.py (and output to 1.l2.ldscore.gz), use the following command:

```
correlate.py chr_@ effect --ldscores chr_@
```

This should give a correlation between direct effects and average NTCs of close to 0.5. The estimated correlations and their standard errors, estimated by block-jacknife, are output to effect_corrs.txt.

The method is similar to LDSC, but correlates the marginal effects, adjusting for the known sampling variance-covariance matrix of the effects. The LD scores are used for weighting. LD scores output by LDSC can be input. If LD scores are not available, they can be computed from .bed files by providing them through the –bed argument.

1.6 Polygenic score analyses

In addition to family based GWAS, *snipar* provides a script (pgs.py) for computing polygenic scores (PGS) based on observed/imputed genotypes, and for performing family based polygenic score analyses. Here, we give some examples of how to use this script. The script computes a PGS from weights provided in LD-pred format . The true direct genetic effects for the simulated trait are given as PGS weights in this format in direct_weights.txt. This is a tab-delimited text file with a header and columns 'chrom' (chromosome), 'pos' (position), 'sid' (SNP ID), 'nt1' (allele 1), 'nt2' (allele 2), 'raw_beta' (raw effect estimates), 'ldpred_beta' (LD-pred adjusted weight). The script uses as weights the 'ldpred_beta' column.

To compute the PGS from the true direct effects, use the following command:

```
pgs.py direct --bed chr_@ --imp chr_@ --weights direct_weights.txt
```

This uses the weights in the weights file to compute the polygenic scores for each genotyped individual for whom observed or imputed parental genotypes are available. It outputs the PGS to direct.pgs.txt, which is a white-space delimited text file with columns FID (family ID, shared between siblings), IID (individual ID), proband (PGS of individual with given IID), maternal (observed or imputed PGS of that individual's mother), paternal (observed or imputed PGS of that individual's father). To use .bgen input, replace the *–bed* argument with *–bgen*.

To estimate direct, paternal, and maternal effects of the PGS, use the following command:

```
pgs.py direct --pgs direct.pgs.txt --phenofile phenotype.txt
```

This uses a linear mixed model that has a random effect for mean differences between families (defined as sibships here) and fixed effects for the direct, paternal, and maternal effects of the PGS. It also estimates the 'population' effect of the PGS: the effect from regression of individuals' phenotypes onto their PGS values. The estimated effects and their standard errors are output to direct.effects.txt, with the effect names (direct, paternal, maternal, population) in the first column, their estimates in the second column, and their standard errors in the final column. The sampling variance-covariance matrix of direct, paternal, and maternal effects is output in direct.vcov.txt.

Estimates of the direct effect of the PGS should be equal to 1 in expectation since we are using the true direct effects as the weights, so the PGS corresponds to the true direct effect component of the trait. The parental effect estimates capture the correlation between the direct and indirect parental effects. The population effect estimate should be greater than 1, since this captures both the direct effect of the PGS, and the correlation between direct and indirect parental effects.

If parental genotypes have been imputed from sibling data alone, then imputed paternal and maternal PGS are perfectly correlated, and the above regression on proband, paternal, and maternal PGS becomes co-linear. To deal with this, add the –parsum option to the above command, which will estimate the average parental effect rather than separate maternal and paternal effects of the PGS.

It is also possible to estimate indirect effects from siblings. We can compute the PGS for genotyped individuals with genotyped siblings and estimate direct, indirect sibling, paternal and maternal effects in one command with the addition of the –fit_sib option:

```
pgs.py direct_sib --bed chr_1 --imp chr_1 --weights direct_weights.txt
--phenofile phenotype.txt --fit_sib
```

This outputs the PGS values for each individual along with the PGS value of their sibling, and imputed/observed paternal and maternal PGS to direct_sib.pgs.txt. (If an individual has multiple genotyped siblings, the average of the siblings' PGS is used for the PGS of the sibling.) It outputs estimates of direct, indirect sibling, paternal, and maternal effects of the PGS to direct_sib.effects.txt and their sampling variance-covariance matrix to direct_sib.vcov.txt. Since indirect effects from siblings were zero in this simulation, the estimated sibling effect should be close to zero.

Note that the standard error for the direct effect estimate increases: this is due both to a drop in sample size since only those probands with genotyped siblings are included, and due to the fact that adding the sibling effect to the regression decreases the independent information on the direct effect.

CHAPTER

TWO

GUIDE

Introduction

snipar (single nucleotide imputation of parents) is a python library for imputing missing parental genotypes from observed genotypes in a nuclear family, and for performing family based genome-wide association and polygenic score analyses using the resulting imputed parental genotypes.

***snipar* contains command line scripts:**

1. for inferring IBD segments shared between siblings (ibd.py)
2. imputing missing parental genotypes from observed parent/offspring genotypes and IBD segments (impute.py),
3. performing genome-wide estimation of direct genetic effects, non-transmitted coefficients, and population effects of SNPs (gwas.py),
4. for estimating direct effects and non-transmitted coefficients of polygenic scores (pgs.py),
5. and for estimating genome-wide correlations between direct and population effects and direct effects and non-transmtitted coefficients (correlate.py)

CHAPTER

THREE

INDICES AND TABLES

- genindex
- modindex
- search