User manual for `SS-X12`

Alexandre M. Harris and Michael DeGiorgio

October 15, 2018

Contents

1 Introduction **2**

2 Operation **2**

2.1 Running `SS-X12` from the command line 3

2.2 Converting VCF files to `SS-X12` format with `vcf2ssx` 4

3 Default input file format **5**

4 Output format **6**

5 Examples **7**

1 Introduction

SS-X12 is a Python program that computes the expected haplotype homozygosity statistics $H12_{Tot}$, $H123_{Tot}$, $H2_{Tot}/H1_{Tot}$, SS-H12, and SS-H123, as well as their multilocus genotype (MLG) data equivalents ($G12_{Tot}$, $G123_{Tot}$, $G2_{Tot}/G1_{Tot}$, SS-G12, and SS-G123, respectively) from dense multilocus polymorphism data. These statistics are defined as in Harris and DeGiorgio [2018], and differ in name only to reflect the data type to which they are applied. Thus, SS-X12 is agnostic to the ploidy of input data (haploid or diploid), provided that it fits the input file criteria (see Section 3: *Input file format*). Additionally, SS-X12 may be applied to samples containing individuals from an arbitrary number of defined populations K, but we caution that analysis speed decreases as K increases.

Note that you will need to convert your data to fit the default the SS-X12 input format if your input file is not formatted exactly like the example (see Section 3: *Input file format* and Section 5: *Examples*). To assist with this conversion, we provide VCF to SS-X12 conversion software, vcf2ssx (also written in Python), with your download of this program, and describe its operation in Section 2.2: *Converting VCF files to SS-X12 format with vcf2ssx*.

Please contact Alex Harris (amh522@psu.edu) if you encounter any issues in your use of these programs. If necessary, then please also include an example converted data file (such as the first 1000 SNPs of your format-converted data).

If you use either SS-X12 or vcf2ssx software, please cite it as

> A M Harris and M DeGiorgio (2018). Identifying and classifying shared selective sweeps from multilocus data. *bioRxiv*.

2 Operation

SS-X12 and vcf2ssx are meant for use on UNIX systems and are formatted for Python 2.7 specifically. We do not guarantee compatibility with Python 3+. If you encounter error messages in your use of either program, then check your invoked version of Python. We distribute our software in compressed (.tar.gz) format. Your download includes the source codes for SS-X12 and vcf2ssx, this manual, and a directory containing example data and output. To unpack the SS-X12_program directory from the command line, enter

```
tar -xzvf SS-X12_program.tar.gz

cd ./SS-X12_program
```

These commands will create the directory SS-X12_program within the current directory, and switch the user to SS-X12_program. We recommend that all input files be in the same directory as the SS-X12 script. Additionally, output files are written to the local directory if the user does not specify a path within the outfile_name argument (see next section).

2.1 Running SS-X12 from the command line

Invoke SS-X12 with the command

```
python SS-X12.py <infile> <selected_populations> <population_header>
<outfile_name> <window_size> <window_shift> <first_group> <second_group>
```

Eight arguments in total must be defined. These are:

1. infile, the file name for an SS-X12-formatted plain text document (see Section 3: *Input file format*) containing the dense multilocus polymorphism data to be analyzed; for example, my_infile_hap.txt (be aware that you MUST name input files containing haplotype data as _hap.txt and input files containing diploid MLG data as _mlg.txt)
2. selected_populations, submitted as a comma-separated string (case sensitive, matching contents of the population header file, see next section) containing all populations to be analyzed; for example, pop1,pop2,pop3,pop4
3. population_header, the name for the file containing the population assignment for each individual in the infile; for example, my_population_IDs.txt
4. outfile_name, a string that defines what the output file will be called, and where it will print; for example, path/to/outfile_name (outfile format is .txt by default; no path defined causes output to print to local directory)
5. window_size, integer corresponding to the physical size of the analysis window, in nucleotides
6. window_shift, integer corresponding to the physical distance, in nucleotides, that the analysis window moves along the chromosome after each computation
7. first_group, comma-separated integers or single integer indicating the populations whose sampled individuals belong to the first of two groups; indexing of the groups begins from 1 and corresponds to the order defined in selected_populations; NOTE that this argument can be none, rather than an integer, to indicate that grouped analysis will not be performed
8. second_group, same as above; note that any permutation of the two groups is valid, such that 1,2 3,4 and 3,4 1,2 and 4,3 1,2 (for example) will yield identical results (see Section 4: *Output options* for details about grouped analysis)

A complete command line invocation of SS-X12, using the above nomenclature (including grouped analysis), is thus:

```
python SS-X12.py my_infile_hap.txt pop1,pop2,pop3,pop4 my_population_IDs.txt
path/to/outfile_name 40000 4000 1,2 3,4
```

Here, we are running SS-X12 on four populations from my_infile_hap.txt (containing phased haplotype data), using population header file my_population_IDs.txt, outputting outfile_name.txt to the directory path/to/, scanning with a 40 kb window that advances by 4 kb increments, and additionally requesting a grouped analysis wherein group 1 consists of the first and second populations (pop1 and pop2), and group 2 consists of the third and fourth populations (pop3 and pop4).

2.2 Converting VCF files to SS-X12 format with vcf2ssx

Here, we describe the operation of vcf2ssx, a Python script which converts VCF files to compressed files in SS-X12 format (as .txt.gz). SS-X12 is compatible with either compressed (must contain ".gz" in the file name) or uncompressed input data files. Invoke vcf2ssx with the command

```
python vcf2ssx.py <vcf_infile> <formatting> <nonbi_removed>
[<chromosome_index>] [<position_index>] [<rsID_index>] [<refAllele_index>]
[<altAllele_index>] [<data_index>]
```

A minimum of three arguments is required, with six additional arguments depending on the choice of formatting:

1. vcf_infile, the name of the VCF file to be converted to SS-X12 format; for example my_data.vcf.gz (note that uncompressed VCFs are also compatible with vcf2ssx)
2. formatting, defines whether the order of columns in the VCF follows the standard VCFv4.2 format or not; answering yes indicates formatting following VCFv4.2, meaning that vcf2ssx requires no further arguments; answering no subsequently requires the user to define the index positions of important columns in the infile (see items 4-9)
3. nonbi_removed, defines whether the user has removed all sites from their input VCF file that are NOT biallelic SNPs; answering yes indicates that such sites are removed, while answering no indicates that such sites are retained in the input (requiring vcf2ssx to filter these out)
4. chromosome_index, optional argument if formatting is no, integer specifying the index of the chromosome number column in the VCF input file (indexing begins from zero for this and subsequent arguments)
5. position_index, optional argument if formatting is no, integer specifying the index of the physical position column in the VCF input file
6. rsID_index, optional argument if formatting is no, integer specifying the index of the SNP rsID column in the VCF input file
7. refAllele_index, optional argument if formatting is no, integer specifying the index of the reference allele column in the VCF input file
8. altAllele_index, optional argument if formatting is no, integer specifying the index of the alternate allele column in the VCF input file
9. data_index, optional argument if formatting is no, integer specifying the first data column's index in the VCF input file

A complete command line invocation of vcf2ssx, with standard formatting of the VCF input file and sites that are not biallelic SNPs removed, is:

```
python vcf2ssx.py my_data.vcf.gz yes yes
```

3 Default input file format

Here, we discuss the default format of the data file which `SS-X12` takes as input, as well as the `population header` file that must accompany the input data. The example data files included within the `SS-X12_program` directory are formatted according to our requirements.

The `SS-X12` input data file is individual-delimited and contains data for all individuals from all study populations, as only one data file is analyzed per run. Additionally, each input data file must contain data from only one chromosome. Remember to include `_hap` at the end of the filename to indicate phased haplotype data, or `_mlg` to indicate unphased MLG data. We recommend running analyses on multiple chromosomes in parallel with a multithreading or multiprocessing module (such as `parallel` in R or `multiprocessing` in Python).

For S analyzed SNPs and n sampled diploid individuals, the input data file contains S lines of data. Each line consists of summary information on the SNP—chromosome ID, SNP rsID, physical position, reference allele, and alternate allele—followed by n entries corresponding to the allelic state of each sampled individual at that SNP. Thus, data lines contain $n+5$ entries. The data line should look as follows:

12 rs536857393 93597 C T 1 1 3 1 1 4 4 2 4 2

For phased haplotype data, an individual's allelic state is coded as 1, 2, 3, or 4. States 1 and 4 correspond to homozygous genotypes for the reference (VCF format: 0|0) and alternate alleles (1|1), respectively. States 2 and 3 correspond to heterozygous genotypes, where the alternate allele is located on the second haplotype in state 2 (0|1) and on the first haplotype in state 3 (1|0). For unphased diploid MLGs, states 2 and 3 are not included. Instead, an individual's heterozygous allelic state is 5 (1/0 or 0/1). Missing sites are encoded as the letter "N", corresponding to each instance of "." in the VCF.

The `population header` file must contain a single line of n entries. Each entry is the name of a sampled population, and this name must be invoked exactly within the `selected_populations` argument (see above). Population assignments are indexed to match the position of individuals in the input data file, such that an entry at position p in the `population header` file is the population assignment of the individual at position $p+5$ in the input data file. Note that least two populations are required for the operation of SS-X12. The header line should look as follows (two populations, CEU and CHB are included here; same example as above):

CEU CEU CEU CEU CEU CHB CHB CHB CHB CHB

Although our example here features only 10 individuals, with 5 from either population, we caution that `SS-X12` performs best for larger sample sizes. That is, the sample size needs to be sufficient for the captured variation to properly distinguish between neutral and selected regions of the genome. To make this determination, we recommend running `SS-X12` on a few megabases (Mb) of your data and observing the contrast in value between signal peaks and signal valleys. Larger sample sizes are required in order to classify sweeps as hard or soft, as the difference between these signals is more

subtle than between selection and neutrality. We have tested samples of 25 diploids per population in simulated data based on human-inspired parameters and find that this works well for detecting shared sweeps, but have not tried smaller sizes. We also find that we have more power to detect sweeps from phased haplotype data than from MLG data (for the same number of individuals, the MLG sample size is half of the haploid sample size). Moreover, we could not classify shared sweeps as hard or soft from samples consisting of fewer than 100 diploids. Accordingly, analyses in Harris and DeGiorgio [2018] feature samples of 100 diploids per population. For users seeking to classify shared sweeps as hard or soft, we recommend large sample sizes.

4 Output format

The output of `SS-X12` consists of a single, space-separated, uncompressed plain text (.txt) file in which every line is the result from an analysis window. Thus, for W analysis windows from the data, the output will consist of W lines. Accordingly, selecting a smaller window size will yield an output file of larger size. As an example, the output for human chromosome 1 using a 40 kilobase (kb) analysis window advancing by 4 kb increments is 7.7 Mb in size, while using a 20 kb window advancing by 2 kb produces a 15.3 Mb file.

An `SS-X12` output line consists of the following:

- H12_{Tot} (alternatively G12_{Tot} for MLGs)
- H123_{Tot} (alternatively G123_{Tot} for MLGs)
- $\text{H2}_{\text{Tot}}/\text{H1}_{\text{Tot}}$ (alternatively $\text{G2}_{\text{Tot}}/\text{G1}_{\text{Tot}}$ for MLGs)
- max(SS-H12) (alternatively max(SS-G12) for MLGs), the maximum value of SS-H12 (SS-G12) among all absolute maximum values of SS-H12 (SS-G12) across all pairs of populations in the analysis
- max(SS-H123) (alternatively max(SS-G123) for MLGs), the maximum value of SS-H123 (SS-G123) among all absolute maximum values of SS-H123 (SS-G123) across all pairs of populations in the analysis
- min(SS-H12) (alternatively min(SS-G12) for MLGs), the minimum value of SS-H12 (SS-G12) among all absolute maximum values of SS-H12 (SS-G12) across all pairs of populations in the analysis
- min(SS-H123) (alternatively min(SS-G123) for MLGs), the minimum value of SS-H123 (SS-G123) among all absolute maximum values of SS-H123 (SS-G123) across all pairs of populations in the analysis
- grouped SS-H12 (alternatively grouped SS-G12 for MLGs), computed only if `first_group` and `second_group` are not `none`, and following the grouping specified in `first_group` and `second_group`
- grouped SS-H123 (alternatively grouped SS-G123 for MLGs), computed only if `first_group` and `second_group` are not `none`, and following the grouping specified in `first_group` and `second_group`

- window_center, the physical position of the window center on the input chromosome
- global_SNPs, the number of SNPs in the entire sample of all populations
- sample_SNPs, the number of SNPs in the sample of populations under analysis (smaller than global_SNPs unless your sample consists of all populations in the input file)

Note that if analysis is performed on only two populations, then max(SS-H12) and min(SS-H12) will be identical, as will max(SS-H123) and min(SS-H123). Thus, the same values will be repeated. Furthermore, grouping is meaningless for samples consisting of individuals from only two populations.

5 Examples

In the `examples` directory within `SS-X12_program`, we provide compressed example data in both `SS-X12` format and as VCF, as well as example output files. Data come from Phase 3 of the 1000 Genomes Project [Auton et al., 2015]. The included data consist of the `population header` file for the 2,504 sequenced individuals within the 1000 Genomes Project dataset, the first 1,000 SNPs of chromosome 12 from the 1000 Genomes Project (as .vcf.gz and .txt.gz), and the `SS-X12` output files for the scan of `chr12phase3_first1000_hap.txt.gz`, for analysis of `CEU,GBR`, `CEU,JPT`, `CEU,GIH`, and `CEU,YRI` two-population analyses, as well as `CEU/GBR,YRI` three-population analysis with CEU and GBR consisting of one group, and YRI consisting of the other.

The commands required to generate the output files are:

- To convert `chr12phase3_first1000_hap.vcf.gz` to `chr12phase3_first1000_hap.txt.gz`, enter
  `python vcf2ssx.py examples/chr12phase3_first1000_hap.vcf.gz yes no` (the VCF file is not required to have `_hap` or `_mlg` in the filename at this stage)
- To scan `chr12phase3_first1000_hap.txt.gz` for the `CEU,YRI` population pair and output results to the current directory, enter
  `python SS-X12.py examples/chr12_first1000_hap.txt.gz CEU,YRI examples/1000genomes_phase3_head.txt examples/chr12_first1000_CEUYRI 20000 2000 none none`
- To scan `chr12phase3_first1000_hap.txt.gz` for the `CEU/GBR,YRI` with grouping of CEU and GBR, enter
  `python SS-X12.py examples/chr12_first1000_hap.txt.gz CEU,GBR,YRI examples/1000genomes_phase3_head.txt examples/chr12_first1000_CEUGBR-YRI 20000 2000 1,2 3`

References

A Auton, G R Abecasis, and The 1000 Genomes Project Consortium. A global reference for human genetic variation. *Nature*, 526:68–74, 2015.

A M Harris and M DeGiorgio. Identifying and classifying shared selective sweeps from multilocus data. *bioRxiv*, 2018.