`Selscan v2.1` User Manual

Zachary A. Szpiech

April 8, 2025

Contents

1 Introduction

Extended Haplotype Homozygosity (EHH) (Sabeti *et al.*, 2002), Integrated Haplotype Score (iHS) (Voight *et al.*, 2006), Cross-population Extended Haplotype Homozygosity (XPEHH) (Sabeti *et al.*, 2007), iHH12 (Garud *et al.*, 2015; Torres *et al.*, 2018), nSL (Ferrer-Admetlla *et al.*, 2014), and XP-nSL (Szpiech *et al.*, 2021) are statistics designed to use phased genotypes to identify putative regions of recent or ongoing positive selection in genomes. (Starting with selscan v2.0.0, the –unphased flag can be used to compute iHS, nSL, XP-EHH, and XP-nSL with unphased data. (Szpiech, 2024)) They are all based on the model of a selective sweep, where a *de novo* adaptive mutation arises on a haplotype that quickly sweeps toward fixation, reducing diversity around the locus (a "hard" sweep). If selection is strong enough, this occurs faster than recombination or mutation can act to break up the haplotype, and thus a signal of high haplotype homozygosity can be observed extending from an adaptive locus. nSL and XP-nSL in retain power to detect soft sweeps as well.

As genetics data sets grow larger both in number of individuals and number of loci, there is a need for a fast and efficient publicly available implementation of these statistics. Below we introduce these statistics and provide concise definitions for their calculations.

When using `selscan` please cite Szpiech and Hernandez (2014) as well as the appropriate paper that introduced the statistic: EHH (Sabeti *et al.*, 2002), iHS (Voight *et al.*, 2006), XP-EHH (Sabeti *et al.*, 2007), nSL (Ferrer-Admetlla *et al.*, 2014), iHH12 (Garud *et al.*, 2015; Torres *et al.*, 2018), XP-nSL (Szpiech *et al.*, 2021), and unphased versions of the statistics (Szpiech, 2024).

This introduction has been adapted from Szpiech and Hernandez (2014).

2 Obtaining `selscan`

2.1 Pre-built Binaries

Pre-built `selscan` binaries are available at `https://github.com/szpiech/selscan`. Binaries are provided for macOS Universal (including both `arm64` and `x86_64`/OSX), Linux, and Windows, and should work on most versions of these operating systems. You can find them in the `bin` directory.

2.2 Compile from source

For some users, compilation from source is preferable. To compile from source, change directories to the `src/` directory and simply type `make`.

If you prefer to use specific Makefiles, run `make -f Makefile_linux` for Linux, `make -f Makefile_win` for Windows, or `make -f Makefile_macos` for macOS. You may need to make minor adjustments to the Makefile, such as commenting or uncommenting certain lines, depending on your target OS and the level of platform-specific optimization desired.

The companion program `norm` depends on GNU GSL (`http://www.gnu.org/software/gsl/gsl.html`), and precompiled static libraries for each OS are included in the source code.

2.2.1 Environment setup For Windows

`selscan` relies on POSIX threads (via the C++17 standard) and the zlib library (`http://www.zlib.net/`). Since the C++17 threading library is not fully supported with GCC on Windows, building requires `clang++` with MinGW-w64 (`http://www.mingw.org/`). Additional requirements include Git, `make`, and zlib (a static version is provided).

The provided Windows binaries were built using `clang++` in a MinGW-w64 environment via MSYS2 (`https://www.msys2.org/`), a lightweight Unix-like shell and package manager for Windows that provides up-to-date development tools and libraries.

Installing MSYS2, Clang++/MinGW Toolchain, and Zlib (optional) and GSL (optional) To compile C++ code with Clang++ and link against libraries like pthread and zlib on Windows, first install the MSYS2 environment (download from `https://www.msys2.org/` and follow the installer instructions). Then, use `pacman` to install the Clang-based MinGW toolchain along with the necessary packages. Static libraries for zlib and GSL are included, but optionally can be installed separately if desired.

First system update is performed:

```
pacman -Syu
# Restart shell after update
```

Then, packages are installed:

```
pacman -S --noconfirm \
  git \
  base-devel \
  mingw-w64-clang-x86_64-toolchain \
  mingw-w64-clang-x86_64-zlib \
  mingw-w64-clang-x86_64-gsl
```

2.2.2 Environment setup For MacOS/Linux

For macOS and Linux, the requirements to compile from source are minimal: just `git`, `make` and a C++ compiler such as `g++` or `clang++`, which are typically included in most modern distributions. Static libraries are already provided.

2.2.3 Commands to Clone and Build Project

```
git clone https://github.com/szpiech/selscan.git
cd selscan && git checkout main
cd src && make
```

You can also use custom makefiles specific to your architecture. For example, instead of `make`, use `make -f Makefile_linux` for linux.

3 Basic Usage

To calculate EHH:

```
selscan --ehh <locusID> --hap <haplotypes> --map <mapfile> --out <outfile>
```

To calculate iHS:

```
selscan --ihs --hap <haplotypes> --map <mapfile> --out <outfile>
```

To calculate XP-EHH:

```
selscan --xpehh --hap <pop1 haplotypes> --ref <pop2 haplotypes> --map <mapfile> --out <outfile>
```

To calculate nSL:

```
selscan --nsl --hap <haplotypes> --map <mapfile> --out <outfile>
```

To calculate iHH12:

```
   selscan --ihh12 --hap <haplotypes> --map <mapfile> --out <outfile>
```

To calculate XP-nSL:

```
   selscan --xpnsl --hap <pop1 haplotypes> --ref <pop2 haplotypes> --out <outfile>
```

4 Statistics implemented

Here we describe the various statistics implemented in selscan. Sections 4.1, 4.2, and 4.3 are reproduced with minor modifications from Szpiech and Hernandez (2014).

4.1 Extended Haplotype Homozygosity (EHH)

Extended Haplotype Homozygosity (EHH) was introduced by Sabeti *et al.* (2002). In a sample of n chromosomes, let $\mathcal{C}$ denote the set of all possible distinct haplotypes at a locus of interest (named x_0), and let $\mathcal{C}(x_i)$ denote the set of all possible distinct haplotypes extending from the locus x_0 to the i^{th} marker either upstream or downstream from x_0. For example, if the locus of interest x_0 is a biallelic SNP where 0 represents the ancestral allele and 1 represents the derived allele, then $\mathcal{C} := \{0, 1\}$. If x_1 is an immediately adjacent marker, then the set of all possible haplotypes is $\mathcal{C}(x_1) := \{11, 10, 00, 01\}$.

EHH of the entire sample, extending from the locus x_0 out to marker x_i, is calculated as

$$EHH(x_i) = \sum_{h \in \mathcal{C}(x_i)} \frac{\binom{n_h}{2}}{\binom{n}{2}}, \tag{1}$$

where n_h is the number of observed haplotypes of type $h \in \mathcal{C}(x_i)$.

In some cases, we may want to calculate the haplotype homozygosity of a sub-sample of chromosomes all carrying a 'core' haplotype at locus x_0. Let $\mathcal{H}_c(x_i)$ be a partition of $\mathcal{C}(x_i)$ containing all distinct haplotypes carrying the core haplotype, $c \in \mathcal{C}$, at x_0 and extending to marker x_i. Note that

$$\mathcal{C}(x_i) = \bigcup_{c \in \mathcal{C}} \mathcal{H}_c(x_i). \tag{2}$$

Following the example above, if the derived allele (1) is chosen as the core haplotype, then $\mathcal{H}_1(x_1) := \{11, 10\}$. Similarly, if the ancestral allele is the core haplotype, then $\mathcal{H}_0(x_1) := \{00, 01\}$

We calculate the EHH of the chromosomes carrying the core haplotype c to marker x_i as

$$EHH_c(x_i) = \sum_{h \in \mathcal{H}_c(x_i)} \frac{\binom{n_h}{2}}{\binom{n_c}{2}}, \tag{3}$$

where n_h is the number of observed haplotypes of type $h \in \mathcal{H}_c(x_i)$ and n_c is the number of observed haplotypes carrying the core haplotype ($c \in \mathcal{C}$).

4.2 Integrated Haplotype Score (iHS)

Integrated Haplotype Score (iHS) was introduced by Voight *et al.* (2006). iHS is calculated by using Equation 3 to track the decay of haplotype homozygosity for both the ancestral and derived haplotypes extending from a query site. To calculate iHS at a site, we first calculate the integrated haplotype homozygosity (iHH) for the ancestral (0) and derived (1) haplotypes ($\mathcal{C} := \{0, 1\}$) via trapezoidal quadrature.

$$\begin{aligned} iHH_c = & \\ & \sum_{i=1}^{|\mathcal{D}|} \frac{1}{2} \left(EHH_c(x_{i-1}) + EHH_c(x_i) \right) g(x_{i-1}, x_i) + \\ & \sum_{i=1}^{|\mathcal{U}|} \frac{1}{2} \left(EHH_c(x_{i-1}) + EHH_c(x_i) \right) g(x_{i-1}, x_i), \end{aligned} \tag{4}$$

where $\mathcal{D}$ is the set of markers downstream from the current locus such that $x_i \in \mathcal{D}$ denotes the i^{th} closest downstream marker from the locus of interest (x_0). $\mathcal{U}$ and $x_i \in \mathcal{U}$ are defined similarly for upstream markers. $g(x_{i-1}, x_i)$ gives the genetic distance between two markers. The (unstandardized) iHS is then calculated as

$$\ln\left(\frac{iHH_1}{iHH_0}\right). \tag{5}$$

Note that this definition differs slightly from that in Voight *et al.* (2006), where unstandardized iHS is defined with iHH_1 and iHH_0 swapped.

Finally, the unstandardized scores are normalized in frequency bins across the entire genome.

$$iHS = \frac{\ln\left(\frac{iHH_1}{iHH_0}\right) - E_p\left[\ln\left(\frac{iHH_1}{iHH_0}\right)\right]}{SD_p\left[\ln\left(\frac{iHH_1}{iHH_0}\right)\right]}, \tag{6}$$

where $E_p\left[\ln\left(\frac{iHH_1}{iHH_0}\right)\right]$ and $SD_p\left[\ln\left(\frac{iHH_1}{iHH_0}\right)\right]$ are the expectation and standard deviation in frequency bin p.

In practice, the summations in Equation 4 are truncated once $EHH_c(x_i) < 0.05$ or the computation extends more than 1Mbp from the core. Additionally with low density SNP data, if the physical distance b (in kbp) between two markers is > 20, then $g(x_{i-1}, x_i)$ is scaled by a factor of $20/b$ in order to reduce possible spurious signals induced by lengthy gaps. During computation if the start/end of a chromosome arm is reached before $EHH_c(x_i) < 0.05$ or if a gap of $b > 200$ is encountered, the iHS calculation is aborted for that locus. iHS is not reported at core sites with minor allele frequency < 0.05. In `selscan`, the EHH truncation value, gap scaling factor, and core site MAF cutoff value are all flexible parameters definable on the command line.

4.3 Cross-population Extended Haplotype Homozygosity (XP-EHH)

Cross-population Extended Haplotype Homozygosity (XP-EHH) was introduced by Sabeti *et al.* (2007). To calculate XPEHH between populations A and B at a marker x_0, we first calculate iHH for each population separately, integrating the EHH of the entire sample in the population (Equation 1).

$$\begin{aligned} iHH = & \\ & \sum_{i=1}^{|\mathcal{D}|} \frac{1}{2} \left(EHH(x_{i-1}) + EHH(x_i) \right) g(x_{i-1}, x_i) + \\ & \sum_{i=1}^{|\mathcal{U}|} \frac{1}{2} \left(EHH(x_{i-1}) + EHH(x_i) \right) g(x_{i-1}, x_i) \end{aligned} \tag{7}$$

If iHH_A and iHH_B are the iHHs for populations A and B, then the (unstandardized) XPEHH is

$$\ln\left(\frac{iHH_A}{iHH_B}\right), \tag{8}$$

and after genome-wide normalization we have

$$XPEHH = \frac{\ln\left(\frac{iHH_A}{iHH_B}\right) - E\Big[\ln\left(\frac{iHH_A}{iHH_B}\right)\Big]}{SD\Big[\ln\left(\frac{iHH_A}{iHH_B}\right)\Big]}. \tag{9}$$

In practice, the sums in each of iHH_A and iHH_B (Equation 7) are truncated at x_i—the marker at which the EHH of the haplotypes *pooled across populations* is $EHH(x_i) < 0.05$ or if the computation extends more than 1Mbp from the core. Scaling of $g(x_{i-1}, x_i)$ and handling of gaps is done as for iHS, and these parameters are definable on the `selscan` command line. For XP-EHH computations, data provided with the flags `--hap`, `--tped`, or `--vcf` correspond to population A, and data provided with the flags `--ref`, `--tped-ref`, or `--vcf-ref` correspond to population B.

4.4 nSL

nSL is a statistic related to iHS and was introduced by Ferrer-Admetlla *et al.* (2014). nSL can be reformulated to conform to the notation given above. nSL is calculated as a log-ratio of the SL statistic calculated for the ancestral and derived haplotype pools. The essential difference from iHS is the removal of genetic map information. Whereas in Equation 4 we integrate with respect to a genetic map, for the calculation of nSL $g(x_i, x_j) = |j - i|$. Thus,

$$\begin{aligned} SL_c = \\ \sum_{i=1}^{|\mathcal{D}|} \frac{1}{2}\left(EHH_c(x_{i-1}) + EHH_c(x_i)\right) g(x_{i-1}, x_i) + \\ \sum_{i=1}^{|\mathcal{U}|} \frac{1}{2}\left(EHH_c(x_{i-1}) + EHH_c(x_i)\right) g(x_{i-1}, x_i), \end{aligned} \tag{10}$$

and

$$nSL = \frac{\ln\left(\frac{SL_1}{SL_0}\right) - E_p\Big[\ln\left(\frac{SL_1}{SL_0}\right)\Big]}{SD_p\Big[\ln\left(\frac{SL_1}{SL_0}\right)\Big]}. \tag{11}$$

Note that, for nSL, $g(x_{i-1}, x_i) = 1$.

For the nSL option, there is no EHH decay cutoff, but the computation stops when more than 200 snps are included in building the haplotypes (can be changed with `--max-extend-nsl`). Scaling of $g(x_{i-1}, x_i)$ and handling of gaps is done as for iHS, and these parameters are definable on the `selscan` command line.

4.5 XP-nSL

XP-nSL is a statistic related to XP-EHH in the same way nSL is related to iHS and was introduced by Szpiech *et al.* (2021). XP-nSL can be reformulated to conform to the notation given above. XP-nSL is calculated as a log-ratio of the SL statistic calculated for each population's haplotype pools. The essential difference from XP-EHH is the removal of genetic map information. Whereas in Equation 4 we integrate with respect to a genetic map, for the calculation of nSL $g(x_i, x_j) = |j - i|$. Thus,

$$SL = \sum_{i=1}^{|\mathcal{D}|} \frac{1}{2}\left(EHH(x_{i-1}) + EHH(x_i)\right) g(x_{i-1}, x_i) + \sum_{i=1}^{|\mathcal{U}|} \frac{1}{2}\left(EHH(x_{i-1}) + EHH(x_i)\right) g(x_{i-1}, x_i), \tag{12}$$

and

$$XP - nSL = \frac{\ln\left(\frac{SL_A}{SL_B}\right) - E\left[\ln\left(\frac{SL_A}{SL_B}\right)\right]}{SD\left[\ln\left(\frac{SL_A}{SL_B}\right)\right]}. \tag{13}$$

Note that, for nSL, $g(x_{i-1}, x_i) = 1$.

For the XP-nSL option, there is no EHH decay cutoff, but the computation stops when more than 200 snps are included in building the haplotypes (can be changed with `--max-extend-nsl`). Scaling of $g(x_{i-1}, x_i)$ and handling of gaps is done as for iHS, and these parameters are definable on the `selscan` command line. For XP-nSL computations, data provided with the flags `--hap`, `--tped`, or `--vcf` correspond to population A, and data provided with the flags `--ref`, `--tped-ref`, or `--vcf-ref` correspond to population B.

4.6 Integrated Haplotype Homozygosity Pooled (iHH12)

iHH12 (Torres *et al.*, 2018) is adapted from the H12 statistic (Garud *et al.*, 2015), with better power than iHS to detect soft sweeps. For this statistic, we calculate the integrated haplotype homozygosity of the entire sample, but we first collapse the top two most frequent haplotypes into a single class.

To calculate iHH12 at a site, we first calculate the integrated haplotype homozygosity (iHH) for the sample using $EHH12$ via trapezoidal quadrature.

$$iHH = \sum_{i=1}^{|\mathcal{D}|} \frac{1}{2}\left(EHH12(x_{i-1}) + EHH12(x_i)\right) g(x_{i-1}, x_i) + \sum_{i=1}^{|\mathcal{U}|} \frac{1}{2}\left(EHH12(x_{i-1}) + EHH12(x_i)\right) g(x_{i-1}, x_i), \tag{14}$$

where $\mathcal{D}$ is the set of markers downstream from the current locus such that $x_i \in \mathcal{D}$ denotes the i^{th} closest downstream marker from the locus of interest (x_0). $\mathcal{U}$ and $x_i \in \mathcal{U}$ are defined similarly for upstream markers. $g(x_{i-1}, x_i)$ gives the genetic distance between two markers. $EHH12$ is then defined as

$$EHH12(x_i) = \frac{\binom{n_{h_1}+n_{h_2}}{2}}{\binom{n}{2}} + \sum_{h \in \mathcal{C}(x_i)\setminus\{h_1,h_2\}} \frac{\binom{n_h}{2}}{\binom{n}{2}}, \tag{15}$$

where h_i is the i^{th} most frequent haplotype in the sample and n_{h_i} is the number of observed h_i haplotypes.

Finally, the scores are normalized across the entire genome.

$$iHH12 = \frac{iHH12 - E\Big[iHH12\Big]}{SD\Big[iHH12\Big]}, \tag{16}$$

In practice, the summations in Equation 14 are truncated once $EHH12_c(x_i) < 0.05$ or the computation extends more than 1Mbp from the core. Additionally with low density SNP data, if the physical distance b (in kbp) between two markers is > 20, then $g(x_{i-1}, x_i)$ is scaled by a factor of $20/b$ in order to reduce possible spurious signals induced by lengthy gaps. During computation if the start/end of a chromosome arm is reached before $EHH12_c(x_i) < 0.05$ or if a gap of $b > 200$ is encountered, the iHH12 calculation is aborted for that locus. iHH12 is not reported at core sites with minor allele frequency < 0.05. In `selscan`, the EHH truncation value, gap scaling factor, and core site MAF cutoff value are all flexible parameters definable on the command line.

4.7 Mean Pairwise Sequence Difference (π)

The mean pairwise sequence difference among a sample of n haplotypes is

$$\pi = \frac{1}{\binom{n}{2}} \sum_{i=1}^{n-1} i(n-i)\xi_i, \tag{17}$$

where ξ_i is the unfolded site frequency spectrum.

5 Program Options

Using the command line flag `--help`, will print a help dialog with a summary of each command line option.

5.1 Input Files

All genetic data is required to be coded 0/1 and must not contain missing data. Consecutive loci are assumed to be in order with respect to their physical location on the chromosome. `selscan` assumes only one non-homologous chromosome per file, and different non-homologous chromosomes should be computed separately.

These methods (excluding π) assume *phased* haplotypes. If your haplotypes are unphased you will have substantially reduced power to identify regions undergoing positive selection. Two popular programs that perform haplotype phasing are SHAPEIT2 (Delaneau *et al.*, 2013) and Beagle v4.0 (Browning and Browning, 2007).

Selscan treats the allele coded as '1' in the input as the derived allele and the allele coded as '0' as the ancestral allele. If you have ancestral *polarization* information, you will need to process your input and flip allele codes accordingly. It is possible to run these analyses without polarizing alleles, at a small power cost.

Any file format that `selscan` supports may be directly read as a gzipped (`http://www.gzip.org/`) version without first decompressing.

5.1.1 `--hap` and `--ref` (msprime-style haplotype matrix)

Use `--hap` to specify a `.hap` file containing phased, binary haplotype data. For XP-EHH, specify a reference population using `--ref`. Both files must contain the same set of loci.

This format, typically produced by simulators like `msprime` (i.e. via `tskit.genotype_matrix()`), represents *haplotypes as rows* and *variant sites as columns.* Each entry indicates the presence (1) or absence (0) of the derived allele:

```
% 4 haplotypes, 5 sites
0 1 0 0 1
1 1 0 0 0
0 1 1 0 1
1 0 1 1 0
```

A corresponding `.map` file is required to provide genetic and physical coordinate information.

5.1.2 `--thap` and `--thap-ref` (IMPUTE-style haplotype matrix)

In `selscan v2.1`, we now support another binary haplotype matrix format as an efficient alternative input, that is transpose of msprime-style format. Use `--thap` for the IMPUTE-style matrix, where rows are variant sites and columns are haplotypes, the transpose of the msprime-style matrix. For XP-EHH, specify the reference population using `--thap-ref`.

This format mirrors the layout of genotype matrix used in tools like `IMPUTE2`, `bcftools` and `vcftools`, and must be binary, and whitespace-delimited:

```
% 5 sites, 4 haplotypes
0 1 0 1
1 1 1 0
0 0 1 1
0 0 0 1
1 0 1 0
```

As with previous format, both files must contain the same set of loci, and a matching `.map` file is required.

This format is supported in `selscan v2.1` for efficiency and compatibility with modern pipelines. You can extract this format using `vcftools` directly from a phased VCF file:

```
vcftools --vcf input.vcf --IMPUTE --out output_prefix
```

This will produce `output_prefix.impute.hap` — haplotype matrix (IMPUTE-style). We can also generate impute style hap using `bcftools` via following command.

```
bcftools convert --hapsample example example.vcf.gz
```

5.1.3 `--vcf` and `--vcf-ref`

Use `--vcf` to specify a .vcf file (see `https://github.com/samtools/hts-specs` for exact specifications) containing genetic variant information. If computing XP-EHH, use `--vcf-ref` to specify a .vcf formatted sample from the desired reference population. Reference population files are expected to contain the exact same loci as the non-reference file.

A .vcf file organizes variant data with rows representing consecutive loci and columns delimited by whitespace representing a diploid sample. For a given diploid sample at a particular locus a genotype is represented by two alleles separated by either / or |. The first nine columns contain information about the locus and the file is organized in the following way:

```
<chr#> <physical position> <id> <reference allele> <alternate allele> <
   quality> <filter> <info> <format> <individual 1 genotype> ... <
   individual N genotype>
```

All rows preceeded by a # symbol will be ignored. While a VCF file can encode quite a lot of information, `selscan` assumes biallelic phased genetic data and will not perform any checks on those assumptions nor perform any filtering (except filtering low frequency variants if `--skip-low-freq` is set). In fact, a .vcf file passed to `selscan` need not strictly conform to the general specifications. For example,

```
1 100 rs1 0 1 . . . . 0|1 0|0
1 200 rs2 0 1 . . . . 1|1 1|0
1 300 rs3 0 1 . . . . 0|0 1|0
1 400 rs4 0 1 . . . . 0|0 1|0
1 500 rs5 0 1 . . . . 1|0 0|1
```

is sufficient to represent two dipoid samples with variant information at five loci.

Note that even though physical position information is given on column two, `selscan` expects a .map file to provide genetic and physical map information.

5.1.4 `--tped` and `--tped-ref`

Use `--tped` to specify a .tped (transposed PLINK; Purcell *et al.* (2007)) file (see `http://pngu.mgh.harvard.edu/` for exact specifications) containing genetic variant information. If computing XP-EHH, use `--tped-ref` to specify a .tped formatted sample from the desired reference population. Reference population files are expected to contain the exact same loci as the non-reference file. Note that `selscan` expects .tped formatted data to be coded 0/1 only.

A .tped file organizes variant data with rows representing consecutive loci and columns delimited by whitespace representing haploid samples. The first four columns contain information about the locus and the file is organized in the following way:

```
<chr#> <id> <genetic position> <physical position> <haploid copy 1> ... <
   haploid copy N>
```

For example,

```
1 rs1 0.01 100 0 0 1 0 0
1 rs2 0.02 200 0 1 1 1 0
1 rs3 0.03 300 0 0 0 1 0
1 rs4 0.04 400 0 0 0 1 0
1 rs5 0.05 500 0 1 0 0 1
```

is sufficient to represent two dipoid samples with variant information at five loci.

Note that for .tped files `selscan` does not expect a .map file to provide genetic and physical map information, as this information is contained in the first four columns of the file. When calculating XP-EHH, map information is taken only from the file specifed with `--tped`.

5.1.5 `--map`

For all file formats except .tped, `selscan` requires a PLINK (Purcell *et al.*, 2007) formatted map file (unless `--pmap` is used with vcf.)

The columns are delimited by whitespace and contain information about each locus. The file is organized in the following way:

```
<chr#> <id> <genetic position> <physical position>
```

For example,

```
1 rs1 0.01 100
1 rs2 0.02 200
1 rs3 0.03 300
1 rs4 0.04 400
1 rs5 0.05 500
```

5.2 Output Files

`selscan` produces two files as output. Results are output to a .out file and a log is output to a .log file. The .log file will record the runtime parameters as well as any information regarding the exclusion of particular loci. Results (.out) files are formatted in the following way.

For EHH the file will be named `<outfile>.ehh.<locusID>[.alt].out` and formatted as

```
<physicalPos> <geneticPos> <`1' EHH> <`0' EHH>
```

For iHS the file will be named `<outfile>.ihs[.alt].out` and formatted as

```
<locusID> <physicalPos> <'1' freq> <ihh1> <ihh0> <unstandardized iHS>
```

or if `--ihs-detail` is set as

```
<locusID> <physicalPos> <'1' freq> <ihh1> <ihh0> <unstandardized iHS> <
   derived_ihh_left> <derived_ihh_right> <ancestral_ihh_left> <
   ancestral_ihh_right>
```

For XP-EHH the file will be named `<outfile>.xpehh[.alt].out` and formatted as

```
<locusID> <physicalPos> <geneticPos> <popA '1' freq> <ihhA> <popB '1'
   freq> <ihhB> <unstandardized XPEHH>
```

For nSL the file will be named `<outfile>.nsl[.alt].out` and formatted as

```
<locusID> <physicalPos> <'1' freq> <sl1> <sl0> <unstandardized nSL>
```

For XP-nSL the file will be named `<outfile>.xpnsl[.alt].out` and formatted as

```
<locusID> <physicalPos> <geneticPos> <popA '1' freq> <slA> <popB '1' freq
   > <slB> <unstandardized XPnSL>
```

For iHH12 the file will be named `<outfile>.ihh12[.alt].out` and formatted as

```
<locusID> <physicalPos> <'1' freq> <unstandardized iHH12>
```

For π the file will be named `<outfile>.pi.<winsize>bp.out` and formatted as

```
<window start> <window end> <pi>
```

Handling Undefined Statistics For certain loci, some statistics may be undefined or ambiguous in interpretation. In such cases, the locus is skipped during processing, and the reason is documented in the log file. In the case of EHH, if either `EHH1` or `EHH0` is undefined, the defined value is reported normally, while the undefined one is represented as `-1`.

5.2.1 `--out`

Use `--out` to provide a base name for an output file. This will be used in place of `<outfile>` above. Default value is `outfile`.

5.3 Choice of Statistic

5.3.1 `--ehh <string>`

Use `--ehh <Locus ID>` to specify a core locus and calculate EHH curves for the ancestral and derived haploytpes extending out to a fixed distance. Can use locus ID or physical position. Associated flags: `--ehh-win`.

5.3.2 `--ihs`

Set `--ihs` to calculate iHS (see Section 4.2). Associated flags: `--pmap`, `--cutoff`, `--maf`, `--gap-scale`, `--max-gap`, `--max-extend`, `--trunc-ok`, `--alt`, `--skip-low-freq`.

5.3.3 `--xpehh`

Set `--xpehh` to calculate XP-EHH (see Section 4.3). Associated flags: `--pmap`, `--cutoff`, `--gap-scale`, `--max-gap`, `--max-extend`, `--trunc-ok`, `--alt`, `--wagh`.

5.3.4 `--nsl`

Set `--nsl` to calculate nSL (see Section 4.4). Associated flags: `--maf`, `--gap-scale`, `--max-gap`, `--max-extend-ns`
`--trunc-ok`, `--alt`.

5.3.5 `--xpnsl`

Set `--xpnsl` to calculate XP-nSL (see Section 4.5). Associated flags: `--maf`, `--gap-scale`, `--max-gap`, `--max-extend-nsl`, `--trunc-ok`, `--alt`.

5.3.6 `--ihh12`

Set `--ihh12` to calculate iHH12 (see Section 4.6). Associated flags: `--maf`, `--gap-scale`, `--max-gap`, `--max-extend`, `--trunc-ok`, `--alt`.

5.3.7 `--pi`

Set `--pi` to calculate the mean pairwise sequence difference within a sliding window along the genome (see Section 4.7). Associated flag: –pi-win.

5.4 Controling How a Statistic is Computed

5.4.1 `--unphased`

Set `--unphased` to use unphased implementations of iHS, nSL, XP-EHH, and XP-nSL computations.

5.4.2 `--pmap`

Set `--pmap` to use physical distances instead of genetic distances for iHS and XP-EHH computations.

5.4.3 `--cutoff <double>`

Use `--cutoff` to set the EHH decay stopping condition. When computing iHS or XP-EHH, the EHH decay curve is truncated and integrated once the EHH decay cutoff is reached. Default is 0.05.

5.4.4 `--maf <double>`

Use `--maf <double>` to set a minor allele frequency (MAF) threshold. Any site below this will not be used as a core site for iHS and nSL scans. Default is 0.05.

5.4.5 `--gap-scale <int>`

Use `--gap-scale <int>` to set the gap scale parameter (Voight *et al.*, 2006). When computing iHS, XP-EHH, and nSL, if a gap of B bp is encountered and is greater than GAP_SCALE, then the distance function $g(x_i, x_j)$ is weighted by GAP_SCALE/B. Default is $20,000$ bp.

5.4.6 `--max-gap <int>`

Use `--max-gap <int>` to set the maximum allowed gap between loci when assembling haplotypes for iHS, XP-EHH, and nSL computations. If a gap greater than this is encountered before a stop condition is reached, the computation at the current core locus is aborted. Default is $200,000$ bp.

5.4.7 `--max-extend <int>`

Use `--max-extend <int>` to set an additional stopping condition for iHS and XP-EHH computations. If the EHH decay curve has extended MAX_EXTEND bp away from the core without reaching the ehh decay cutoff, truncate the curve here and integrate. Default is $1,000,000$ bp; set ≤ 0 for no restriction.

5.4.8 `--max-extend-nsl <int>`

Use `--max-extend-nsl <int>` to set a stopping condition for nSL and XP-nSL computations. If the EHH decay curve has extended MAX_EXTEND loci away from the core, truncate the curve here and integrate. Default is 100 loci; set ≤ 0 for no restriction.

5.4.9 `--trunc-ok`

Core loci near the boundaries of the data set are unlikely to reach a stopping condition before running out of haplotype information. Typically in this case, EHH curves are truncated and thrown out. Set `--trunc-ok` to integrate these anyway.

5.4.10 `--alt`

Set `--alt` to compute EHH using sample haplotype frequencies:

$$EHH(x_i) = \sum_{h \in \mathcal{C}(x_i)} \left(\frac{n_h}{n}\right)^2 \tag{18}$$

and

$$EHH_c(x_i) = \sum_{h \in \mathcal{H}_c(x_i)} \left(\frac{n_h}{n_c}\right)^2. \tag{19}$$

Contrast with Equations 1 and 3.

5.4.11 `--skip-low-freq`

This flag no longer functions. It is effectively on by default. If you wish to include low frequency sites use `--keep-low-freq`. Set `--skip-low-freq` during an iHS scan to pre-filter all sites with a MAF less than that specified with `--maf`. If sites are not pre-filtered, `selscan` will use these sites to construct haplotypes, but will not use them as core sites (and thus will not report a score).

5.4.12 `--keep-low-freq`

Set `--keep-low-freq` during an iHS scan to use low MAF sites to construct haplotypes, but `selscan` will not use them as core sites (and thus will not report a score).

5.4.13 `--ehh-win <int>`

Set `--ehh-win <int>` to define for a single EHH computation the maximum extension in base pairs from the query locus. Default is $100,000$ bp.

5.4.14 `--pi-win <int>`

Set `--pi-win <int>` to define the size of the non-overlapping windows in base pairs for calculating π. Default is 100 bp.

5.5 Other Options

5.5.1 `--ihs-detail`

Set `--ihs-detail` to write out left and right iHH scores for ancestral (0) and derived (1) in addition to iHS.

5.5.2 `--threads <int>`

`selscan` uses shared memory parallelism to speed up computations on multi-core workstations. Use `--threads <int>` to set the number of concurrent threads that `selscan` will use. `selscan v2.1` uses thread pool to achieve multi-threading across loci. Default is 1.

5.5.3 `--multi-param <string>`

Use `--multi-param <string>` to provide a config file in json format containing parameter sets.

Example:

The command line looks like this:

`selscan --vcf chr1.vcf --multi-param config.json`

where a sample `config.json` looks like this

```
{
  "params": [
    {
      "maf": 0.01,
      "max-extend": 1000000,
      "cutoff": 0.05
    },
    {
      "maf": 0.05,
      "max-extend": 1000000,
      "cutoff": 0.05
    }
   ]
}
```

Here two output files are generated.

If multiple statistics are provided in command line, i.e.

```
selscan \
--vcf chr1.popA.vcf \
--vcf-ref chr1.popB.vcf \
--xpehh --nsl --ihh12 \
--multi-param config.json
```

Then, 2 × 4 = 8 output files are generated.

6 Change Log

```
15APR2025 - selscan v2.1.0 - Introducing fast and memory-efficient
   versions for all statistics. See Rahman, et al. (2025) Biorxiv for
   details.

There is a new batch option for efficient processing of multiple
   statistics or parameters at once. See the manual for full details.
       --multi-param <filename>: Specify a JSON file with multiple
          parameter sets. Each set should match the structure of command-
          line arguments. The program will run the analysis for each set,
          generating separate outputs. Useful for batch processing and
          exploring different configurations."

Support for impute-style hap format is added (see --thap and --thap-ref).
```

17NOV2023 - selscan v2.0.1 - Bug fixes for --ehh flag. Total EHH at the core snp will now be reported correctly (i.e. homozygosity of the site and not as 0). Also implemented --unphased for --ehh, and EHH output files now have a header line.

- selscan v2.0.2 - Small change that should result in faster runtime when --pmap set.

22OCT2021 - selscan v2.0.0 - Introducing unphased versions of iHS, nSL, XP-EHH, and XP-nSL. Use with --unphased flag. See ZA Szpiech (2024) Bioinformatics for details. Normalize as you would with the phased statistics.

20MAY2020 - selscan v1.3.0 - Log ratios are now output as log10 not natural logs (beware comparisons with raw selscan computations from versions prior to v1.3.0). New statistics implemented.

```
        --pmap <bool>: Set this flag to use physical distance instead of
            genetic map
```

Introduction of XP-nSL, this statistic is a cross population statistic for identifying hard/soft sweeps. Does not require a genetic map. XP-nSL:nSL::XP-EHH:iHS

```
        --xpnsl <bool>: Set this flag to calculate XP-nSL.
        Default: false
```

Normalize XP-nSL with --xpnsl flag in norm.

lasugden adds the option to calculate XP-EHH with either definition of EHH. By default, uses original denominator (N choose 2). To use denominator defined in Wagh et al. for better performance on incomplete sweeps, use flag --wagh

```
        --wagh <bool>: Set this flag to calculate EHH with Wagh
            denominator. For xpehh only. DO NOT use with --alt
        Default: false
```

Normalize these computations with --xpehh flag in norm.

norm v1.3.0 - Now supports --xpnsl flag, which is identical to using --xpehh.
--qbins now has a default value of 10 instead of 20.
--bp-win analyses have been changed when analyzing XP-EHH and XP-nSL scores. Since positive scores suggest adaptation in the first (non-ref) population and negative scores suggest adaptation in the second (ref

) population, we split windows into those enriched for extreme positive scores and those enriched for extreme negative scores.

min and max scores are given for each window for XP statistics, and the max |score| is reported for iHS and nSL stats.

*.windows output files therefore have additional columns:

For XP stats:

<win start> <win end> <# scores in win> <frac scores gt threshold> <frac scores lt threshold> <approx percentile for gt threshold wins> <approx percentile for lt threshold wins> <max score> <min score>

For iHS and nSL:

<win start> <win end> <# scores in win> <frac scores gt threshold> <frac scores lt threshold> <approx percentile for gt threshold wins> <approx percentile for lt threshold wins> <max score> <min score>

18SEPT2017 - norm v1.2.1a, selcan v1.2.0a iHH12 output files have a header line, XPEHH header line has an extra column name, fixed norm bugs relating to normalization of ihh12 files.

25AUG2017 - norm v1.2.1 released to fix a crash when --nsl flag is used.

18JUL2017 - Support for iHH12 calculations. norm has --ihh12 and --nsl flags.

09JAN2017 - Fixed buggy --crit-percent flag in norm binary.

05SEPT2016 - Fixed misleading error messages when --trunc-ok used.

12FEB2016 - v 1.1.0b - The flag --skip-low-freq is now on by default and no longer has any function. selscan now filters low frequency variants by default. A new flag --keep-low-freq is available if you would like to include low frequency variants when building haplotypes (low frequency variants will still be skipped over as core loci), using this option may reduce the power of iHS scans.

28OCT2015 - Updates to norm so that it can handle output from selscan when --ihs-detail is used.

18JUNE2015 - v1.1.0a - When calculating nSL, a mapfile is no longer required for VCF. Physical distances will be read directly from VCF. A mapfile specifying physical distances is stille required for .hap files when calculating nSL. selscan now appropriately reports an error if this is not provided.

15JUNE2015 - Release of 1.1.0. tomkinsc adds the --ihs-detail parameter which, when provided as an adjunct to --ihs, will cause selscan to write out four additional columns to the output file of iHS calculations (in order): derived_ihh_left, derived_ihh_right, ancestral_ihh_left, and ancestral_ihh_right.

An example file row follows, with header added for clarity.

```
locus           phys-pos        1_freq          ihh_1           ihh_0
            ihs             derived_ihh_left     derived_ihh_right
   ancestral_ihh_left      ancestral_ihh_right

16133705        16133705        0.873626        0.0961264       0.105545
          -0.0934761      0.0505176              0.0456087
   0.0539295               0.0516158
```

From these values we can calculate iHS, but it is preserved in the output for convenience. Having left and right integral information may assist certain machine learning models that gain information from iHH asymmetry.

selscan can now calculate the nSL statistic described in A Ferrer-Admetlla, et al. (2014) MBE, 31: 1275-1291. Also introduced a check on map distance ordering. Three new command line options.

--nsl <bool>: Set this flag to calculate nSL.
        Default: false

--max-extend-nsl <int>: The maximum distance an nSL haplotype is allowed to extend from the core.
        Set <= 0 for no restriction.
        Default: 100

--ihs-detail <bool> : Set this flag to write out left and right iHH scores for '1' and '0' in addition to iHS.

06MAY2015 - Release of 1.0.5. Added basic VCF support. selscan can now read .vcf and .vcf.gz files but without tabix support. A mapfile is required when using VCF. Two new command line options.

13MAY2015 - norm v1.0.5 is released. norm will now normalize ihs or xpehh scores. Two new command line options.

--ihs <bool>: Do iHS normalization.

--xpehh <bool>: Do XP-EHH normalization.

Exactly one of these must be specified when running norm (e.g. ./norm --ihs --files *.ihs.out or ./norm --xpehh --files *.xpehh.out).

```
--vcf <string>: A VCF file containing haplotype data.
        A map file must be specified with --map.

--vcf-ref <string>: A VCF file containing haplotype and map data.
        Variants should be coded 0/1. This is the 'reference'
        population for XP-EHH calculations and should contain the same
           number
        of loci as the query population. Ignored otherwise.

07JAN2015 - norm bug fix and --skip-low-freq works for single EHH queries
   .

12NOV2014 - The program norm has been updated to allow for user defined
   critical values.  Two new command line options.

--crit-percent <double>: Set the critical value such that a SNP with iHS
   in the most extreme CRIT_PERCENT tails (two-tailed) is marked as an
   extreme SNP.
        Not used by default.

--crit-val <double>: Set the critical value such that a SNP with |iHS| >
   CRIT_VAL is marked as an extreme SNP.  Default as in Voight et al.
        Default: 2.00

17OCT2014 - Release of 1.0.4. A pairwise sequence difference module has
   been introduced.  This module isn't multithreaded at the moment, but
   still runs quite fast.  Calculating pi in 100bp windows with 198
   haplotypes with 707,980 variants on human chr22 finishes in 77s on the
    test machine.  Using 100kb windows, it finishes in 34s.  Two new
   command line options.

--pi <bool>: Set this flag to calculate mean pairwise sequence difference
    in a sliding window.
        Default: false

--pi-win <int>: Sliding window size in bp for calculating pi.
        Default: 100

15SEP2014 - Release of 1.0.3.  **A critical bug in the XP-EHH module was
   introduced in version 1.0.2 and had been fixed in 1.0.3.  Do not use
   1.0.2 for calculating XP-EHH scores.**  Thanks to David McWilliams for
    finding this error.  1.0.3 also introduces support for gzipped input
   files.  You may pass hap.gz, map.gz. and tped.gz files interchangably
   with unzipped files using the same command line arguments.  A new
   command line option is available.

--trunc-ok <bool>: If an EHH decay reaches the end of a sequence before
```

```
   reaching the cutoff,
        integrate the curve anyway (iHS and XPEHH only).
        Normal function is to disregard the score for that core.
        Default: false

17JUN2014 - Release of 1.0.2.  General speed improvements have been made,
    especially with threading.  New support for TPED formatted data and
   new command line options are available.

--skip-low-freq <bool>: Do not include low frequency variants in the
   construction of haplotypes (iHS only).
        Default: false

--max-extend: The maximum distance an EHH decay curve is allowed to
   extend from the core.
        Set <= 0 for no restriction.
        Default: 1000000

--tped <string>: A TPED file containing haplotype and map data.
        Variants should be coded 0/1
        Default: __hapfile1

--tped-ref <string>: A TPED file containing haplotype and map data.
        Variants should be coded 0/1. This is the 'reference'
        population for XP-EHH calculations and should contain the same
            number
        of loci as the query population. Ignored otherwise.
        Default: __hapfile2

10APR2014 - Release of 1.0.1.  Minor bug fixes. XP-EHH output header is
   now separated by tabs instead of spaces.  Removed references to
   missing data (which is not accepted), and introduced error checking in
    the event of non-0/1 data being provided.

26MAR2014 - Initial release of selscan 1.0.0.
```

References

Browning, S. R. and Browning, B. L. 2007. Rapid and accurate haplotype phasing and missing data inference for whole genome association studies by use of localized haplotype clustering. *American Journal of Human Genetics*, 81: 1084–1097.

Delaneau, O., Zagury, J. F., and Marchini, J. 2013. Improved whole chromosome phasing for disease and population genetic studies. *Nature Methods*, 10: 5–6.

Ferrer-Admetlla, A., Liang, M., Korneliussen, T., and Nielsen, R. 2014. On detecting incomplete soft or hard selective sweeps using haplotype structure. *Molecular Biology and Evolution*, 31: 1275–1291.

Garud, N. R., Messer, P. W., Buzbas, E. O., and Petrov, D. A. 2015. Recent selective sweeps in north american drosophila melanogaster show signatures of soft sweeps. *PLOS Genetics*, 11(2): 1–32.

Purcell, S., Neale, B., K., T.-B., Thomas, L., Ferreira, M. A. R., Bender, D., Maller, J., Sklar, P., de Bakker, P. I. W., Daly, M. J., and Sham, P. C. 2007. PLINK: a toolset for whole-genome association and population-based linkage analysis. *American Journal of Human Genetics*, 81: 559–575.

Sabeti, P. C., Reich, D. E., Higgins, J. M., Levine, H. Z. P., Richter, D. J., Schaffner, S. F., Gabriel, S. B., Platko, J. V., Patterson, N. J., McDonald, G. J., Ackerman, H. C., Campbell, S. J., Altshuler, D., Cooper, R., Kwiatkowski, D., Ward, R., and Lander, E. S. 2002. Detecting recent positive selection in the human genome from haplotype structure. *Nature*, 419: 832–837.

Sabeti, P. C., Varilly, P., Fry, B., Lohmueller, J., Hostetter, E., Cotsapas, C., Xie, X., Byrne, E. H., McCarroll, S. A., Gaudet, R., Schaffner, S. F., and Lander, E. S. 2007. Genome-wide detection and characterization of positive selection in human populations. *Nature*, 449(7164): 913–918.

Szpiech, Z. A. 2024. selscan 2.0: scanning for sweeps in unphased data. *Bioinformatics*, 40(1): btae006.

Szpiech, Z. A. and Hernandez, R. D. 2014. selscan: an efficient multithreaded program to perform EHH-based scans for positive selection. *Molecular Biology and Evolution*, 31: 2824–2827.

Szpiech, Z. A., Novak, T. E., Bailey, N. P., and Stevison, L. S. 2021. Application of a novel haplotype-based scan for local adaptation to study high-altitude adaptation in rhesus macaques. *Evolution Letters*, 5: 408–421.

Torres, R., Szpiech, Z. A., and Hernandez, R. D. 2018. Human demographic history has amplified the effects of background selection across the genome. *PLoS Genetics*, 14: e1007387.

Voight, B. F., Kudaravalli, S., Wen, X., and Pritchard, J. K. 2006. A map of recent positive selection in the human genome. *PLoS Biology*, 4: e72.