

# **iHDSel Manual v 0.5.3**

*Antonio Carvajal-Rodríguez*

*Facultad de Biología, Campus Lagoas-Marcosende*

*Departamento de Bioquímica Genética e Inmunología*

*Universidad de Vigo, Vigo 36310, Spain*

*Email: [acraaj@uvigo.es](mailto:acraaj@uvigo.es)*

*Web: <http://acraaj.webs.uvigo.es>*

# Table of Contents

Versions.....	3
Introduction.....	4
The Program.....	5
Window size.....	5
No predefined window size.....	5
1) Automatic blocks.....	5
2) Blocks around the outliers.....	6
3) Blocks around candidate sites.....	6
Predefined window size.....	6
Download.....	7
Ready-to-use executables.....	7
Compiling the program manually (linux/macOS).....	7
Input.....	7
VCF format.....	8
VCF header.....	8
The columns of a VCF.....	8
MS format.....	9
FASTA format.....	11
1) From separate files.....	11
2) From one file with two samples with equal sample size. .	12
3) From one file with two unequal samples.....	12
FastPHASE 1.4 format.....	12
HapMap phased format.....	13
Genepop format.....	13
BayeScan format.....	13
PLINK flat files (MAP/PED) format.....	14
MAP files.....	14
User-defined candidates for selective loci.....	16
Special value: inf (all shared SNPs).....	17
Reference haplotype in spatial and temporal comparisons.....	17
Temporal comparisons.....	19
Run the program.....	19
Command line options.....	19
Default Values.....	22
Minimum allele frequency.....	22
Run in MacOSX or Linux.....	22

Run in Windows.....	22
Usage examples.....	23
Windows.....	23
Linux/MacOS.....	23
Output.....	24
$J_{HAC}$ result.....	24
EOS result.....	25
References.....	25

## Versions

### Version iHDSel 0.5.3 (April 2026):

- Computational efficiency improvement: Identified and resolved a bottleneck in low-frequency SNP filtering across populations, substantially reducing runtime.
- The `-selectives` argument now supports `inf`, enabling computation of the  $J_{Hac}$  statistic across all shared positions.
- Fixed a bug in VCF reading that collapsed diploid genotypes into a single haplotype.
- Relaxed the requirement for VCF files to contain the same SNPs.
- Set `doEOS` to 0 (False) by default.

### Version iHDSel 0.5 (May 2024):

Minor changes to improve speed.

### Version iHDSel 0.4 (April 2024):

Calculation of blocks centered on outliers (`-useblocks 0`) has been slightly fixed to be done the same as `-useblocks 1`.

### Version iHDSel 0.3 (April 2024):

Fixed a bug where haplotype blocks could be unusually long if there were some fixed SNPs in the reference population.

### Version iHDSel 0.2 (April 2024):

Three changes from previous versions:

- The SNPs at the beginning and end of each block are now given in the output file.

- Under the `-useblocks 0` option if mean  $F_{st}$  is  $>0.5$  there is not outlier test (EOS) performed but the 90-percentile are used as centers of haplotype blocks.
- New `doEOS` option (True by default) to allow the user to bypass the EOS test by setting `-doEOS 0` in the command line arguments.

#### **Version iHDSel 0.1 (November 2023):**

This is a completely renewed version. The `nvd` method has been replaced by the  $J_{HAC}$  method which uses Jeffreys divergence (population stability index) to compare the distribution of haplotype classes between two populations. The minimum allele frequency (MAF) is by default 0.01 and is computed for the metapopulation (both populations jointly) for the haplotype-based methods but for each population separately for the non-haplotype-based methods (see the corresponding section below).

#### **Version HacDivSel 1.5 (June 2021):**

- Corrects a minor bug that would provoke the program break if the major allele was a gap ('-') when reading the fastfasta format.
- Adds the possibility for the user of introducing specific positions as selective candidates via the argument `-selectives` (see the section of Candidates for selective loci).
- Minimum allele frequency (**MAF**): MAF changed to be a per population relative value (now by default 0.01) previously was an absolute metapopulation value (by default 4).

## **Introduction**

iHDSel is a program for the detection of selection in pairs of populations that may be separated in space or time. It includes a new statistic, called  $J_{HAC}$ , which is based on the information theory interpretation of the Price equation (Frank, 2012) and computes the Jeffreys divergence (population stability index) between the haplotype allelic class (Labuda *et al.*, 2007; Hussin *et al.*, 2010) distributions of both populations.

The  $J_{HAC}$  statistic have three main advantages for detecting divergent selection:

- 1.- Has a known distribution under the null hypothesis of no selection.
- 2.- It is fast because having a known distribution avoids the need for simulations or resampling to assign statistical significance.
- 3.- The statistic directly performs a comparison between distributions, so no correction for multiple testing is required.

As with previous versions, the program also incorporates the extreme-outlier set (EOS) test (Carvajal-Rodríguez, 2017).

## The Program

### Window size

#### No predefined window size

By default there is no predefined window size (`-window 0`) and the program has three different alternative options to inspect candidates for selection and calculate the statistic over an haplotype window centered on each candidate.

#### 1) Automatic blocks

The default option is for the program to find blocks of haplotypes and set the candidate in the middle of each block (`-useblocks 1`). The minimum size of a block is, by default, 11 SNPs (can be changed using the `-minwin` command line argument with range `[7,2147483646]`). The program compute haplotype blocks and set the candidate  $c$  in the middle of each block. An haplotype block is computed as a sequence of SNPs with lenght  $w$  that satisfies  $r^2_{(c-w/2, c-w/2+1)}, \dots, r^2_{(c-1, c)}, r^2_{(c, c+1)}, r^2_{(c+1, c+2)}, \dots, r^2_{(c+w/2-1, c+w/2)}, \dots$  where  $r$  is the correlation coefficient calculated from the sample of size  $n$  so that  $Pr(nr^2) \leq \alpha$  and  $nr^2$  has a Chi square distribution, in addition, a given SNP  $c+1$  is included in the block only if  $D'_{(c, c+1)} \geq 0.4$  where  $D'$  is the

normalized linkage disequilibrium (Lewontin, 1964). The block is extended until any of both tests are rejected i.e.  $Pr(nr^2_{(c+W-1, c+W)} > \alpha)$  or  $D'_{(c+W-1, c+W)} < 0.4$ . If the program does not find any block, it concludes that there is no selection.

## **2) Blocks around the outliers**

The user can decide not to use the default option simply by calling the program with the tag `-useblocks 0`. Two scenarios can happen here. First, if there are outliers the program will use them (extreme or not) and will construct the haplotype blocks around them as indicated above. Second, if there are no outliers, if the mean  $F_{st}$  is greater than 0.5, it will use the 90th percentile values as candidates and again construct the haplotype blocks around that sites. If there are no outliers and the mean  $F_{st}$  is not greater than 0.5, then the program will conclude that there are no candidates to build blocks and terminate.

It could happen that the haplotype blocks built around the outliers were too short or there were no blocks at all. For blocks around outliers, the criteria is not as strict as for automatic blocks. The blocks are discarded only if they do not meet the minimum condition of having an average  $D' > 0$ .

## **3) Blocks around candidate sites**

The user can propose candidate sites using the `-selectives` tag and the corresponding arguments in the command line to introduce the selective positions (see the section [User-defined candidates for Selective Loci](#)). The program uses the user-specified positions as candidate sites to construct haplotype blocks, as described above. When the value is `inf`, all positions shared between the two populations are considered as candidates.

## **Predefined window size**

The window size can be defined by the user by the command line tag `-window`. The size of the window must be at least 11 (the `minwin` value). If the window size is defined the `-useblocks` option is automatically set to 0. There are two options now.

- 1) There are not selective sites defined by the user. Then, the program applies the `-usebloks 0` option as was explained in the previous section (option 2 in no predefined window size).
- 2) There are user-defined selective sites. The program centers each candidate site in a window with the predefined size.

## **Download**

The program can be downloaded from

<https://acraaj.webs.uvigo.es/iHDSel.html>

## **Ready-to-use executables**

Windows 64 bits: iHDSel0.5.2.exe

Ubuntu: iHDSel0.5.2\_U

macOS: iHDSel0.5.2\_M3Pro

## **Compiling the program manually (linux/macOS)**

In a terminal window (console) go to the source folder that should include a file called Makefile, and just type make.

## **Input**

The acceptable formats for the input file are VCF (see the VCF subsection below), MS, Fasta, HapMap phased, FastPhase, Genepop, BayeScan (codominant markers) and PLINK. The three last formats allow only for the execution of the outlier-based method (EOS).

The VCF format must begin with `##fileformat` tag. For the remaining formats iHDSel accepts as commentaries those lines in the very initial part of the file beginning with `#`.

Additionally, under the formats Genepop, HapMap and PLINK, any line in any part of the file can be a commentary if it begins with `#`. Therefore, for these files we can comment some lines in order to eliminate specific individuals (Genepop and PLINK-ped) or SNPs (HapMap) from the sample.

## VCF format

The new version iHDSel accepts a restricted subset of the VCF format. To use this format:

```
-format vcf
```

An example of a program calling

```
./iHDSel -input1 snps_rads.chr5.phased_A.vcf -input2  
snps_rads.chr5.phased_B.vcf -format vcf -output iHDSv0_Chr5 &
```

or the equivalent alternative with explicit default values

```
./iHDSel -path ./ -input1 snps_rads.chr5.phased_A.vcf -input2  
snps_rads.chr5.phased_B.vcf -format vcf -output iHDSv0_Chr5 -verbose 0 -minwin  
11 -maf 0.01 -useblocks 1 &
```

## VCF header

The header must begin with the `##fileformat` tag and the last header line must include the `FORMAT` tag before the sample names, for example if there are two samples the last header line could be:

```
#CHROM POS ID REF ALT QUAL FILTER INFO FORMAT sample1 sample2
```

## The columns of a VCF

The chromosome must be the same within and between the files of the two populations. Likewise, the positions must coincide in both populations to be analyzed.

The format type must be GT (genotypes) and phased e.g. 0|0 or 0|1 or 1|1.

For example a valid file for the first population with 4 SNPs and 3 samples could be:

```
##fileformat=VCFv4.2  
  
##FILTER=<ID=PASS,Description="All filters passed">  
  
##source=shapeit4.1.3  
  
##contig=<ID=5>  
  
##INFO=<ID=AF,Number=A,Type=Float,Description="Allele Frequency">  
  
##INFO=<ID=AC,Number=1,Type=Integer,Description="Allele count">
```



```
##FORMAT=<ID=GT,Number=1,Type=String,Description="Phased genotypes">
```

#CHROM	POS	ID	REF	ALT	QUAL	FILTER	INFO	FORMAT	S1	S2	S3
5	243588	.	C	A	.	.	AC=3;AF=0.0194805;CM=0				GT
		0 0	0 0	0 0							
5	254234	.	T	G	.	.	AC=1;AF=0.06935;CM=0.06				GT
		0 0	0 0	0 0							
5	261675	.	T	G	.	.	AC=3;AF=0.01805;CM=0.01				GT
		0 0	0 0	0 0							
5	420619	.	G	T	.	.	AC=13;AF=0.0846;CM=0.177				GT
		0 0	0 0	0 0							

and for the second populations with the same 4 SNPs and 2 samples:

```
##fileformat=VCFv4.3
```

#CHROM	POS	ID	REF	ALT	QUAL	FILTER	INFO	FORMAT	S7	S15
5	243588	.	C	A	.	.	AC=3;AF=0.004805;CM=0			GT
		0 1	0 0							
5	254234	.	T	G	.	.	AC=1;AF=0.04351;CM=0.010646			
		GT	0 0	1 0						
5	261675	.	T	G	.	.	AC=3;AF=0.00805;CM=0.018087			
		GT	0 0	0 0						
5	420619	.	G	T	.	.	AC=13;AF=0.0156;CM=0.177031			
		GT	1 0	0 0						

## MS format

The MS format is the standard Hudson's ms output format. For example:

```
ms 2 2 -t 60
```

```
104642148
```

```
//
```

```
segsites: 3
```

```
positions: 0.002771      0.004018      0.014038
```

```
000
```

111

//

segsites: 3

positions: 0.1      0.4      0.8

010

101

To use this format (is the default so no argument is needed at all):

`-format ms`

We can read two runs from the same file as if they were different populations (the example above) as produced in the output of the old GenomePop software for samples of two subpopulations. In this case the sample size is assumed to be the same for both populations. For example, if we have generated the file called mu\_r\_60Nm\_10.txt that has 50 haplotypes sampled from each population; to analyze it under iHDSel we type:

```
./iHDSel -path ./ -input mu_r_60Nm_10.txt -output test.out -sample 50 -format  
ms -maf 0.01
```

Alternatively, we could have just one run in the file that is a mixture of two populations as produced by using the corresponding arguments when executing the MS program

```
./ms 100 1 -t 60 -r 60 1000000 -I 2 55 45 40 >> mu_r_60Nm_10.txt
```

that produces one run with a sample of 100 sequences: 55 coming from one population and 45 from another. Then we get, for example:

```
ms 100 1 -t 60 -r 60 1000000 -I 2 55 45 40
```

104642148

//

segsites: 300

```

positions: 0.002771      0.004018      0.014038 ...

000 ...

111 ...

110 ...

010 ...

.

.

000 ...

```

In this case if we want that iHDSel reads the file we should indicate both sample sizes

```

./iHDSel -path ./ -input mu_r_60Nm_10.txt -output test2.out -sample 55 -
sample2 45 -format ms -maf 0.01

```

## FASTA format

The program reads the FASTA sequential format. It allows symbols A, C, G and T. Entries with any other symbol represent unresolved nucleotides and are treated as null alleles. When preparing the data for the computation, the following parsing is performed. In the reference population (the first, by default under equal sample sizes or the one with the largest sample size), the most frequent allele at each position will be noted as '0' and any alternate as '1'. If the most frequent symbol is N that site is ignored. Thus, if the most frequent allele in the reference population is, say, T then this base will be noted as '0' and any other as '1' for this site in the whole data set (all populations).

To use this format:

```

-format fasta

```

The two populations can be read from the same file or from separate files.

### 1) From separate files

```

./iHDSel -path ./TESTS/ -input1 filt_EN3.fas -input2 filt_SA.fas -format fasta
-output filt_EN3_SA -maf 0.01 &

```

## 2) From one file with two samples with equal sample size

```
./iHDSel -path ./TESTS/ -input EN3_SA.fasta -format fasta -output EN3_SA_mix -maf 0.01 &
```

In this case the program assumes the file EN3\_SA.fasta have half of the sequences from and sample and half from the other. If that is not the case the following option should be used.

## 3) From one file with two unequal samples

```
./iHDSel -path ./TESTS/ -input EN3_SA.fasta -format fasta -output EN3_SA -maf 0.01 -tag ENGLAND &
```

Note the use of `-tag ENGLAND` to separate sequences that include that word in the name from other sequences that do not. The sample carrying the tag in the sequences name will be considered the first sample.

There is also a tag `-save` to store the obtained samples in the corresponding files.

```
./iHDSel -path ./TESTS/ -input1 filt_EN3.fasta -input2 filt_SA.fasta -format fasta -output filt_EN3_SA -maf 0.01 -save 1 &
```

## FastPHASE 1.4 format

To use this format:

```
-format fp
```

The program expect two files, one for each population (can have different sample sizes). The SNPs in both files must be the same in the same order. That is, if the *i*-th SNP in population-1 is rs11901199 then rs11901199 is also expected in the *i*-th position in the population-2.

To analyze the files F1\_hapguess\_switch and F2\_hapguess\_switch that are in a folder called FastPhase, we use the following arguments:

```
./iHDSel -path ./FastPhase/ -input F1_hapguess_switch -input2 F2_hapguess_switch -output hapguess_ -format fp
```

## HapMap phased format

Files are organized in a SNPs x haplotypes format, so each row represents a SNP and each column a haplotype. The first row (header row) contains the identifiers of the individuals that have been phased. The first column (header column) contains the rsID of the SNPs. The second column contains the physical position of these SNPs in the particular chromosome. Entries with anything other than A, C, G, or T represent unresolved nucleotides and are treated as null alleles. To use this format:

```
-format hm
```

The program expects two files, one for each population (can have different sample sizes). To make the program read one population from each file we need to add the argument `-singlepopfile 1`. For example, to analyze the files `hm3_chr2_ceu.txt` and `hm3_chr2_yri.txt` that are in a folder called `HapMap`, we use the following arguments:

```
./iHDSel -path ./HapMap/ -input hm3_chr2_ceu.txt -input2 hm3_chr2_yri.txt -  
output hmCEUvsYRI -format hm -singlepopfile 1
```

## Genepop format

This format has no haplotype information thus, only the EOS test is performed using the GST estimator (Nei, 1973). It allows variable sample size as inferred from the data.

To use this format:

```
-format gp
```

```
./iHDSel -path ./ -input mu_r_60Nm_10.genepop -output test.out -format gp -  
maf 0.01
```

## BayeScan format

This format has no haplotype information thus, only the EOS test is performed using the GST estimator. The version of the data format is for codominant markers. It assumes equal sample size in both populations. For example if the number of copies is  $2N = 100$  we set `-sample 50`. To use this format:

```
-format bs

./iHDSel -path ./ -input mu_r_60Nm_10.bs -output test.out -format bs -sample
50 -maf 0.01
```

## **PLINK flat files (MAP/PED) format**

This format has no haplotype information thus, only the EOS test is performed using the GST estimator. It allows variable sample size which is inferred from the data.

### **MAP files**

The fields (separated by space or tab) in a MAP file are:

- Chromosome
- Marker ID
- Genetic distance
- Physical position

For example:

```
2    rs11901199  0    8856
2    rs7594567   0    11833
```

The column with the genetic distances is optional. The following example is also valid:

```
2    rs11901199  8856
2    rs7594567   11833
```

Thus, the number of columns must be 4 or 3, a mixture of both is not valid.

### **PED files**

The fields (separated by space or tab) in a PED file are

- Population ID
- Individual ID

- Paternal ID
- Maternal ID
- Sex (1=male; 2=female; other=unknown)
- Phenotype (0=unknown; 1=unaffected; 2=affected)
- Genotypes (space or tab separated, 2 for each marker)

For example, corresponding to the map file above, the first line could be:

```
Pop1  NA19176      0 0 1 1      G G C C
```

...

Note that the length of the genotype must be 2x the number of data rows in the MAP file.

The genotypes can be nucleotides or numbers. In the case of nucleotides there are only 4 alleles (A, C, G or T) and any other symbol will be considered as missing information (N).

To use this format with nucleotides:

```
-format pl
```

In the case that genotypes are numbers, the maximum number of alleles allowed in the current version is 255 and 0 is the symbol for a missing allele.

To use this format with numbers:

```
-format pl2
```

In any case, if the name of map and ped files is the same e.g. mu\_r\_60Nm\_10.map and mu\_r\_60Nm\_10.ped then the input file for the program must be

```
-input mu_r_60Nm_10
```

without the extension.

For example, for nucleotides:

```
./iHDSel -path ./ -input mu_r_60Nm_10 -output test.out -format pl
```

Or for numbers:

```
./iHDSel -path ./ -input mu_r_60Nm_10 -output test.out -format pl2 -maf 0.01
```

That read both the map and ped files. However, if the names of map and ped files are different e.g. mu\_r\_60Nm\_10.map and mu\_r\_60Nm\_10\_nuc.ped then the input for the program must be

```
-input mu_r_60Nm_10 -input2 mu_r_60Nm_10_nuc -singlepopfile 1
```

again without the extension and with the flag singlepopfile to indicate that two different file names are being considered. Note that in this latter case the -input2 name must be always the ped file name. For example, with nucleotides:

```
./iHDSel -path ./ -input mu_r_60Nm_10 -input2 mu_r_60Nm_10_nuc -singlepopfile  
1 -format pl
```

## User-defined candidates for selective loci

For any haplotype-based input format, one or more SNPs can be specified as candidate sites for selection. Since not all formats include SNP identifiers, the program requires specifying candidates by their position within the haplotype.

For FASTA and fastPHASE formats, SNP positions are 0-indexed (i.e. the first position is 0, not 1). Thus, in a haplotype containing 1,000 SNPs, positions range from 0 to 999. For example, a SNP located at position 700 in 1-based indexing corresponds to position 699 in 0-based indexing, and should be specified as:

```
-selectives 699
```

Multiple candidate SNPs must be provided in increasing order, for example:

```
-selectives 1 699 700
```

As an illustration, consider haplotypes from different populations of Atlantic salmon obtained using fastPHASE. Suppose we want to test a SNP located in the NADP-dependent malic enzyme-2 locus (mMEP-2\*), for which independent evidence suggests local adaptation. This SNP (ctg7180001794010\_7928\_SCT) is located at 39,897,822 bp on chromosome Ssa25 and is flanked by two SNPs at -11 Kb and +15.6 Kb.



After filtering the microarray data, the SNP of interest is located at position 3579 (0-based indexing), with the flanking SNPs at positions 3578 and 3580. Additionally, another SNP at 19,827,046 bp (position 2118) is included as a control, as previous analyses indicate a  $J_{HAC}$  value close to 0.

The corresponding command would be:

```
-path /home/workdir/ -input1 atl_25_hapguess_switch.out -input2  
can_25_hapguess_switch.out -format fp -output  
nvdfst_Atl_Can_25_MEPS2_2118 -selectives 2118 3578 3579 3580
```

For the MS format, SNP positions are relative and range from 0 to 1. For VCF and HapMap formats, SNP positions correspond to the physical coordinates given in the second column of the input files.

### **Special value: inf (all shared SNPs)**

When the value inf is provided to `-selectives`, all SNP positions shared between the two populations are automatically considered as candidate sites. This allows computing the  $J_{HAC}$  statistic genome-wide over the full set of shared positions, without manually specifying individual SNPs.

## **Reference haplotype in spatial and temporal comparisons**

The  $J_{HAC}$  method is based on calculating the distribution of HAC values in each sample. The HAC value is calculated as the Hamming distance of each haplotype with respect to a reference configuration (Carvajal-Rodríguez 2017, 2024). This reference configuration is calculated for one of the samples which we refer to as the reference population.

When we do not define the reference population, it remains undefined and the program in principle calculates it for the largest sample or, if both samples have the same size, for the first sample. However, with the reference undefined, if for the assayed reference population, the program do not find blocks it can use the other population as reference.

Alternatively, we can indicate the reference population using the -reference tag with a number 1 or 2 to indicate the first or second sample.

If the two samples to be compared correspond to contemporary populations separated in space, the reference population, where the selective sweep can be estimated may be one or both. However, if the separation between the samples is temporary, the reference population must necessarily be the second one.

Therefore, for the samples T1 and T2, with T2 having the tag 2021-03 the following line

```
-path /home/workdir/ -input1 T1.fas -input2 T2.fas -format fasta -output  
T1vsT2 -useblocks 0
```

will use

- i) The first passed sample (T1) as reference if both sample sizes are equal, or
- ii) The largest sample and
- iii) For both i) and ii), it will also check the block of a given site in the non-reference sample (which for that block becomes the reference) if that site does not have any blocks around it in the other sample.

That is, it will check blocks in both samples, initially in the first reference (the largest sample) and in the other sample for the cases that a block is not found in the largest sample.

In the output file name, this will be indicated by the text "AutoRefPopx" where x is the reference as defined in i) or ii).

On the contrary, the line

```
-path /home/workdir/ -input1 T1.fas -input2 T2.fas -format fasta -output  
T1vsT2ref1 -useblocks 0 -reference 1
```

will only use T1.fas as a reference to calculate the blocks. In the output file name, this will be indicated by the text "RefPop1".

Similarly,

```
-path /home/workdir/ -input1 T1.fas -input2 T2.fas -format fasta -output  
T1vsT2ref2 -useblocks 0 -reference 2
```

will only use only T2.fas as a reference to calculate the blocks. In the output file name, this will be indicated by the text "RefPop2".

In summary, for the same set of candidate sites (as with user-provided candidates or with the outliers as candidates, i.e., the -useblocks 0 option), the first line without the specified reference will produce the same result as the sum of the results of the other two lines.

Under the automatic blocks option (-useblocks 1) the program uses the specified reference or, if not specified, uses the largest sample and if that doesn't work (no block detected) switches to the other sample to calculate the blocks.

### Temporal comparisons

In the previous case, sample T2 is more recent than sample T1, so it is obvious that the possible selective sweep caused by the selection will be present in sample T2 and therefore, when constructing the blocks, this will be the sample we will use, whether with or without automatic blocks

```
-path /home/workdir/ -input1 T1.fas -input2 T2.fas -format fasta -output  
T1vsT2ref2 -useblocks 1 -reference 2
```

## Run the program

### Command line options

The following arguments can be passed to the program:

**-cutoff<DOUBLE>** Define the value numdesvs in the computation of the cutoff for the EOS test ( $\text{cutoff} = \text{fsttot} + \text{numdesvs} * \text{maxdev}$ ). Default is 1.25.

**-doEOS <INTEGER>** From version 0.5.3, this option defaults to 0. When set to 0, the EOS test is not performed. If the input format does not involve haplotypes and/or the useblocks option (see below) is set to 0, doEOS is automatically set to 1.

**-help|-h** Displays this help message.

**-input <STRING>** Specifies the name of the input file.

**-input2** <STRING> Specifies the name of the second input file. It is mandatory only when the format is vcf or the argument -singlepopfile is set to 1.

**-format** <STRING> Specifies the sequence data format (BS, Fasta, VCF, GP, HM, PL or, by default, MS).

**-maf** <DOUBLE> Defines the minimum allele frequency in each population (default is 0.01).

**-maxwin** <INTEGER> Defines the maximum window size (default is 1000) under the variable window size setting. If set to 0 then it defines all available SNPs as the maximum window size.

**-minwin** <INTEGER> Defines the minimum block size (default is 11 SNPs).

**-nogaps** <INTEGER> By default is 0. Only available under the Fasta format. When set to 1 it skips columns carrying gaps.

**-onlyEOS** <INTEGER> Defines whether only the EOS test is performed. Default is 0 ( $J_{HAC}$  and EOS performed). It is automatically set to 1 when the format is BS, GP or PL.

**-onlyrsidInfo** <INTEGER> By default is 0. When set to 1 the program generates an output file with the names, positions and frequencies of the set of SNPs.

**-output** <STRING> Specifies the header for the name of the output file.

**-path** <STRING> Specifies the path to the input file.

**-reference** <INTEGER> The population used as reference for computing the haplotype allelic classes. By default is undefined so that the sample with the greatest size is used as reference. When defined, it can have values 1 or 2.

**-sample** <INTEGER> Defines the number of alignments in each population sample (50 by default). It is required under the MS and BayeScan formats. It is not required Under the HM, VCF, Fasta and FastPhase formats because the two files can have different sample sizes that will be recognized automatically. If under the MS format we want to read two populations with different sample sizes we need to add a second argument -sample2 (see below). Under the GenePop format the sample size is allowed to vary between populations. Under the BayeScan format the size is implicitly allowed to vary (total number of genes / ploidy) but the value of sample will be used to compute the minimum allele frequency.

**-sample2** <INTEGER> Defines the number of alignments in a second population sample under the MS format. By default has value 0, in this case the program reads two populations (marked with //) from one MS file and expects that each population has the same sample size. If sample2 has a positive value then the program expects a second input file name (-input2 ...) and would read each population in MS format from a different file: -path ./msfiles/ -input msfile1 -input2 msfile2 -output testmsf -sample 40 -sample2 55 -format ms.

**-save** <INTEGER> By default is 0. When the format is Fasta and the two samples are in the same file and can be separated by a given tag, to set -save 1 permits saving the samples in separate files.

**-selectives** <DOUBLE> User defined candidate sites to look for selection (if there are more than one value they must be separated by spaces). When the value is inf, all positions shared between the two populations are considered as candidates.

**-singlepopfile** <INTEGER> This is only necessary under the HM or PLINK formats. By default is 0 so it is assumed that there are two populations (with equal sample size) at each HM input file. When set to 1 this means that two different filenames (-input and -input2, one for each population) should be provided for the program to work. In the case of PLINK this is used for reading map and ped files with different names (see PLINK format section). In the case of the VCF or fastphase formats it is not necessary since the format already require two files (one per population).

**-SL** <DOUBLE> Defines the significance level ( $\alpha=0.05$  by default).

**-tag** <STRING> When the format is Fasta and we read one file with different sample sizes, it permits to look for the given string in the sequence names to separate the data in two samples.

**-useblocks** <INTEGER> By default is 1. It computes the haplotype blocks with the selective candidates in the middle as indicated in the corresponding section. When set to 0, the program uses the SNPs as the center of the blocks that are computed in a similar way.

**-verbose**<INTEGER> Provides additional information during the computation and a log file at the end.

**-window** <INTEGER> Defines the fixed window size. If 0 the window size will be variable (this is the option by default).

## Default Values

Calling the program without arguments, i.e.

```
./iHDSel
```

It is equivalent to calling

```
./iHDSel -path ./ -format ms -input iHdivselF -input2 iHdivselF -window 0 -  
sample 50 -sample2 0 -maf 0.01 -cutoff 1.25 -output iHDS_out_MAF_0.01.xls -  
SL 0.05 -minwin 11 -maxwin 1000 -window 0 -onlyEOS 0 -doEOS 0 -singlepopfile  
0 - onlyrsidInfo 0 -useblocks 1 -save 0 -nogaps 0 -verbose 0
```

## Minimum allele frequency

The  $J_{HAC}$  method only considers SNPs that have at least a frequency of MAF (by default 0.01) when considering jointly both populations.

However, for the non haplotype formats (BS, GP or PL) it is required that the frequency of the allele to be at least MAF at each population separately.


## Run in MacOSX or Linux

If the downloaded executable has name iHDSel (can vary for different versions), from the console or terminal just type  
./iHDSel jointly with the desired arguments.

## Run in Windows

Double click just for running the program under default options. Alternatively, you can go to the command prompt (cmd.exe) and type

```
iHDSel
```

You can also access the Run command by pressing the Windows logo key +r

then drag and drop the .exe file from your folder and add the desired arguments, e.g. if the iHDSel.exe is in the folder work then after drag and drop you will have

```
C:\work\iHDSel.exe
```

now add the desired arguments and then hit the Intro key. For example:

```
C:\work\iHDSel.exe -input neutral.txt -output neutralres
```

## Usage examples

The following examples assume that the terminal/command prompt is located in the directory containing the iHDS0.5.3 executable, and that the input data are stored in the data folder specified in the -path argument.

To analyze files in VCF format using blocks and considering all sites as candidates (i.e. -selectives inf), without including the EOS test, the command line could be:

## Windows

```
iHDSv0.5.3.exe -path C:\Users\PC\data\ ^  
-input1 data_pop1_chr2.vcf ^  
-input2 data_pop2_chr2.vcf ^  
-output pop1vspop2_chr2 ^  
-format vcf -verbose 0 -minwin 11 -maf 0.05 ^  
-useblocks 1 -selectives inf &
```

## Linux/MacOS

```
./iHDS0.5.3 -path /home/data/ \  
-input1 data_pop1_chr2.vcf \  
-input2 data_pop2_chr2.vcf \  
-output pop1vspop2_chr2 \  
-format vcf -verbose 0 -minwin 11 -maf 0.05 \  
-useblocks 1 -selectives inf &
```

## Output

The program generates three files. For example, if the output tag in the argument list was followed by X (`-output X`) then the program generates one file called `X_MAF_0.01_Blocks_FSTs.txt` with the result of the EOS analysis and other two called `X_MAF_0.01_Blocks.tsv` and `X_MAF_0.01_Blocks_SIG.tsv` with the result of the  $J_{HAC}$  analysis.

### $J_{HAC}$ result

The .tsv files would contain the following columns:

**File:** The name(s) of the analyzed file(s).

**#SNPs:** The number of SNPs analyzed.

**Candidate:** The candidate number if more than one were considered.

**SNP:** Position (0-indexed when not VCF or hapmap) of the candidate SNP.

**Wide:** The window size in number of SNPs.

**Ini:** The initial SNP of the haplotype block.

**End:** The terminal SNP of the haplotype block.

**NCLASSES:** The number of classes in which the data were binned to calculate the  $J_{HAC}$  statistic.

**MeanFST:** The average value of FST in the analyzed data.

**FST:** The FST value at the candidate site.

**FSTIndex:** For a given candidate i the  $FST_i$  minus the average FST value.

**fstPval:** The p-value after performing the Lewontin and Krakauer (LK) test.

**n1:** Sample size of the first population.

**n2:** Sample size of the second population.

**JHac:** Value of the  $J_{HAC}$  statistic.

**Pval:** The p-value for the  $J_{HAC}$  statistic value.



**Sig:** Indicates the significance. ns: non-significant. \*: significant (Pval<SL).

## **EOS result**

The .txt file would contain the list of  $F_{ST}$  outliers. First indicates the non-EOS outliers and after them the EOS outliers.

For each type of outliers the first line indicates the file names, the number of SNPs and the number of the given outliers. After that, a table follows with the following columns:

**ID:** The name of the SNP or locus or the position if a name is not available.

**Pos:** The position (0-indexed) of the outlier.

**FST:** The  $F_{ST}$  or  $G_{ST}$  value.

**pval:** The p-value after the LK test.

## **References**

- Carvajal-Rodríguez, A. (2017) HacDivSel: Two new methods (haplotype-based and outlier-based) for the detection of divergent selection in pairs of populations. *PLoS One*, **12**, e0175944.
- Frank, S.A. (2012) Natural selection. V. How to read the fundamental equations of evolutionary change in terms of information theory. *J Evol Biol*, **25**, 2377-96.
- Hussin, J. et al. (2010) Haplotype allelic classes for detecting ongoing positive selection. *BMC Bioinformatics*, **11**, 65.
- Labuda, D. et al. (2007) Patterns of variation in DNA segments upstream of transcription start sites. *Hum Mutat*, **28**, 441-450.
- Lewontin, R.C. (1964) The Interaction of Selection and Linkage. I. General Considerations; Heterotic Models. *Genetics*, **49**, 49-67.
- Nei, M. (1973) Analysis of gene diversity in subdivided populations. *Proceedings of the National Academy of Sciences*, **70**, 3321-3323.