

# A bioinformatics workflow for detecting signatures of selection in genomic data

Murray Cadzow<sup>1,2</sup>, James Boocock<sup>1,2</sup>, Hoang Tan Nguyen<sup>1,2</sup>, Phillip Wilcox<sup>2,3</sup>, Tony R Merriman<sup>1</sup>  
and Michael A Black<sup>1</sup>

<sup>1</sup>Department of Biochemistry, University of Otago

<sup>2</sup>Department of Mathematics and Statistics, University of Otago

<sup>3</sup>Scion Research, Rotorua, New Zealand

September 9, 2013

## Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Getting Started</b>	<b>3</b>
2.1	Prerequisites . . . . .	3
2.2	Python dependencies . . . . .	3
2.3	Installation . . . . .	4
2.4	Genetic Maps and Impute Haplotypes . . . . .	4
2.5	Ancestral Fasta Files . . . . .	5
<b>3</b>	<b>Tutorial</b>	<b>5</b>
3.1	Selection Signatures at the Lactase Locus . . . . .	5
3.1.1	Getting the Data . . . . .	5
3.2	Setting up Pipeline Run . . . . .	5
3.3	Population Files . . . . .	5
3.4	Run The Tutorial . . . . .	6
3.5	Data Visualisation . . . . .	6
3.5.1	Fst . . . . .	6
3.5.2	Fay and Wu's H . . . . .	8
3.5.3	iHS . . . . .	10
3.5.4	Tajima's D . . . . .	12
3.5.5	Rsb . . . . .	14
<b>4</b>	<b>Output Files</b>	<b>14</b>
4.1	multi_population . . . . .	14
4.1.1	Fst . . . . .	14
4.2	selection_pipeline . . . . .	14

4.2.1	Fay and Wu's H	14
4.2.2	iHS	14
4.2.3	iHH	14
4.2.4	Tajima's D	15
<b>5</b>	<b>Command line Arguments</b>	<b>15</b>
5.1	Multipopulation	15
5.1.1	Input Files	15
5.1.2	Output Files	15
5.1.3	Other parameters (Compulsory)	15
5.1.4	Other parameters (Optional)	16
5.2	Selection Pipeline	16
5.2.1	Input Files	16
5.2.2	Output Files	16
5.2.3	Other parameters(Compulsory)	16
5.2.4	Other parameters(Optional)	16
5.3	Ancestral Annotation	17
5.3.1	Input Files	17
5.3.2	Output Files	18
5.3.3	Other parameters	18
5.4	Configuration File	18
5.4.1	system	18
5.4.2	environment	18
5.4.3	selection_pipeline	19
5.4.4	vcf_tools	19
5.4.5	shapeit	19
5.4.6	impute2	19
5.4.7	plink	20
5.4.8	Rscript	20
5.4.9	python	20
5.4.10	ancestral_allele	20
5.4.11	qctool	21
5.4.12	multicore_ihh	21
<b>6</b>	<b>Log Files</b>	<b>21</b>
6.1	multi_population	21
6.2	selection_pipeline	21
<b>7</b>	<b>Extra Features</b>	<b>21</b>
7.1	Galaxy Intergration	21

# 1 Introduction

The Selection Pipeline utilizes next-generation sequencing *NGS* data to generate selective signatures. The tools used to detect selection are dependent on the selection signature being investigated (Sabeti *et al.*, 2006). The pipeline generates various output files containing within and between population selection signatures. The starting point for the analysis is a variant call format *VCF* file of the genotype data and populations of interest (Danecek *et al.*, 2011). Both  $F_{st}$  and Tajima's D can be calculated from standard genotype data (Weir and Cockerham, 1984) (Tajima, 1989). To compute iHS, rsb and Fay and Wu's requires haplotypes, and thus the genotype data must be phased prior to calculation of these statistics (Voight *et al.*, 2006) (Gautier and Vitalis, 2012) (?). For phasing shapeit2 is used and for imputation impute2 is used (Howie *et al.*, 2009) (Delaneau *et al.*, 2013), furthermore these statistics also require ancestral allele information (Flicek *et al.*, 2012). The pipeline phases if the VCF files do not contain the phasing information and then performs ancestral allele annotation. Once complete, the rehh package provides a simple interface for implementing EHH-based analyses (Gautier and Vitalis, 2012), we extended rehh to include penalties for gaps that match those used in the original iHS paper (Voight *et al.*, 2006), rehh calculates iHH, iHS, iES and Rsb. To calculate Fay and Wu's H a c program variscan was utilised (Vilella *et al.*, 2005). The pipeline implemented in python, takes a VCF file to a set of output files containing selection signatures.

# 2 Getting Started

## 2.1 Prerequisites

The selection pipeline was developed on a 64-bit ubuntu 13.04 system and has been tested on a 64-bit centos and ubuntu 13.10 installation. The pipeline should work on any 64-bit linux deriviant assuming some basic libraries and tools are installed on your system. 20gb of ram should be sufficient *requiredforimputation*.

- python  $\geq$  2.6
- bourne-again Shell (Bash)
- perl5
- R  $\geq$  3.0.0 (with a personal library)
- build-essential tools

The software is installed to the user than runs the script, if the user is not root a R local library is required. The program also installs the scripts to the users `./local/bin` directory this directory should be added to the system path to give to direct access to the programs from the command-line.

## 2.2 Python dependencies

- python-setuptools

- python-numpy

Most linux distributions provide these packages through the official package management repositories.

## 2.3 Installation

To install the package standalone, requiring manual configuration of the configuration file run the following command.

```
./install.sh --standalone
```

The rest of this section will be dedicated to the automatic installation. To perform an automatic installation of the selection pipeline run the command.

```
./install.sh
```

Installation creates a default configuration file located in the base directory of the pipeline. Installation adds a program called `selection_pipeline` to the system path. To test the program is installed correctly run the following command at a terminal prompt.

```
selection_pipeline -h
```

## 2.4 Genetic Maps and Impute Haplotypes

To use the phasing and imputation features of the pipeline requires both genetic map files and haplotype files. For humans these files that conform to the format required for shapeit and impute2 can be found [here](#). For impute2 one reference is available [here](#), download and extract the archive to `referencefiles/impute_ref` and uncompress the contents. For shapeit2 a genetic map can be found [here](#), download and extract the archive to `referencefiles/shapeit_ref`.

To use other reference files with the selection pipeline requires setting a few options in the config file. The question mark character "?" in the config is substituted by the chromosome number, this is used for reference files that are split on chromosomes.

```
...
genetic_map_prefix=genetic_map_chr?_combined_b37.txt
...
impute_map_prefix=genetic_map_chr?_combined_b37.txt
impute_reference_prefix=ALL_1000G_phase1integrated_v3_chr?_impute
...
```

If you decide to store your reference files in another location, further options will require alterations.

```

...
genetic_map_dir= \${HOME}/MerrimanSelectionPipeline/referencefiles/shapeit_ref
...
impute_map_dir= \${HOME}/MerrimanSelectionPipeline/referencefiles/impute_ref
impute_reference_dir= \${HOME}/MerrimanSelectionPipeline/referencefiles/impute_ref
...

```

## 2.5 Ancestral Fasta Files

The generation of results for iHS requires assigning the ancestral allele. The selection pipeline uses the ancestral alleles from the 6-way EPO (Enredo-Pecan-Ortheus) alignment pipeline. The files can be downloaded from [here](#). Make sure to extract contents of the archive after download. The default directory to store the ancestral reference files is referencefiles/ancestral\_ref/.

If you downloaded your reference to a different location you can set the following setting in your config file.

```

...
ancestral_fasta_dir = # directory you downloaded alignment to #
...

```

## 3 Tutorial

### 3.1 Selection Signatures at the Lactase Locus

#### 3.1.1 Getting the Data

The lactase gene is located on Chromosome 2 between 136,545,410-136,594,750 positions. For the example we will use a 10 megabase region containing the Lactase gene and the CEU and YRI populations from the 1000 genomes. In order to demonstrate how to use the pipeline we will use the chromosome 2 region 130,000,000-140,000,000. To download the example dataset enter the command below. The lactase gene is an example of strong selection in the last 5,000-10,000 years in human populations specifically European-derived populations (cite).

```
wget http://tutorial_file_location.com
```

Extract the example data into a new folder.

### 3.2 Setting up Pipeline Run

### 3.3 Population Files

Population files are required for any cross population comparisons. The commands below will initiate the data generation step. Population files are line separated files the first line contains the population name every successive line contains and individual ID from that population.

```
<POPULATION\_IDENTIFIER>
<INDIVIDUAL ID 1>
<INDIVIDUAL ID 2>
.....
<INDIVIDUAL ID N>
```

### 3.4 Run The Tutorial

The default configuration file is located in the base directory of the selection pipeline. To run the pipeline run the command below in the folder you extracted the example data and change the `--config-file` parameter to match the location you have installed the pipeline.

```
multipop\_selection_pipeline -p CEU_ids.txt -p YRI_ids.txt
-i CEU_YRI_lactase.vcf --config-file defaults.cfg
--fst-window-size 1000 --fst-window-step 1000
```

The generated folders and current folder have all the data required to perform further selection analysis. Within each population folder 4 output files are generated these contain Tajima's D, iHH, an updated VCF and Fay and Wu's H statistic these files are located in the results folder inside each population subfolder. Fst is calculated between each population and results are located in the fst folder. Fst results are calculated using the Weir and Cockerham estimator.

### 3.5 Data Visualisation

The purpose of the pipeline is to generate standard signatures of selection from a VCF formatted input file. In order to express the usefulness of the pipeline it is pertinent to illustrate the effectiveness of the pipeline. The next section describes some basic plotting of these data using the R programming language. All following commands are run in a R session with the working directory in the base directory you are running the tutorial in. In each case the blue lines outline the lactase gene.

#### 3.5.1 Fst

```
CEUYRIfst=read.table("fst/2CEUYRI.fst", header=TRUE)
# Plot FST 1 megabase each side of the lactase gene
# Plot the weighted fst value for each region
plot(CEUYRIfst[,5]~CEUYRIfst[,2], pch=16, cex=.4, type="p",xlab='',ylab='FST')
rect(136545410,0,136594750,1,border="Blue")
```

Figure 1 shows clustering of high FST values close to the lactase gene plotting one megabase downstream and upstream either side of the gene.

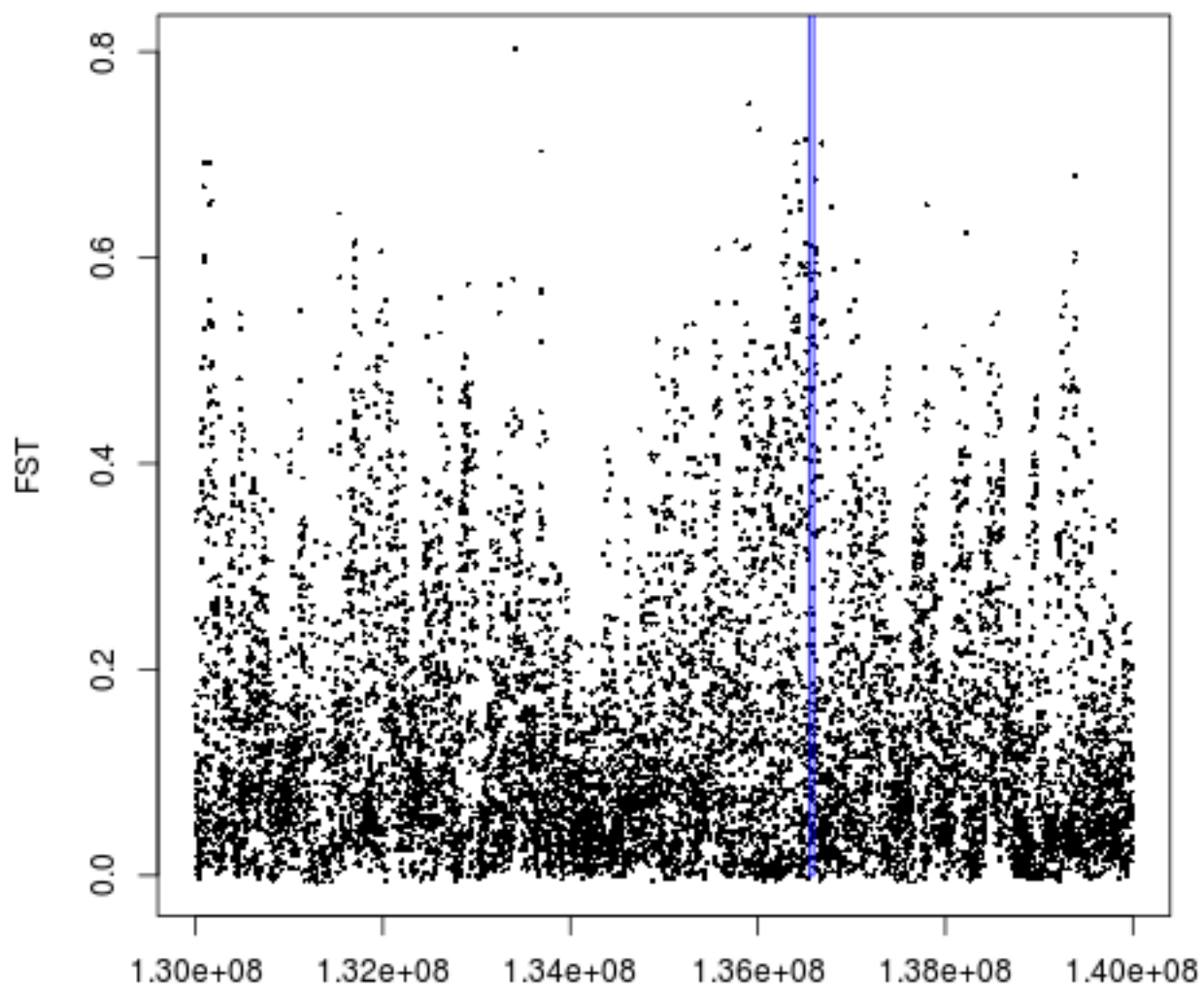


Figure 1: CEU YRI Fst Values

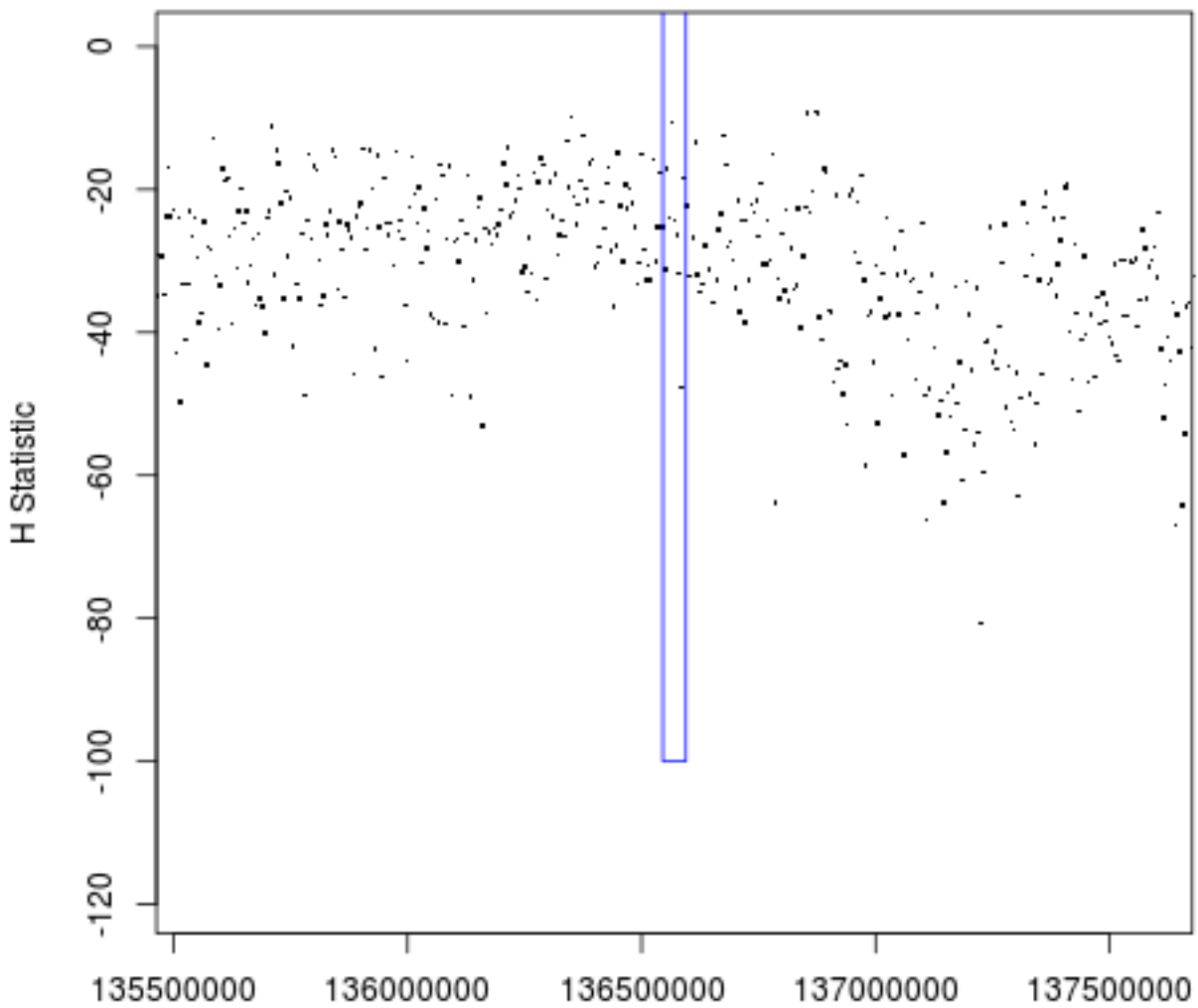


Figure 2: CEU Fay and Wu's H

### 3.5.2 Fay and Wu's H

To plot the Fay and Wu's H values for the CEU population.

```
CEUFay=read.table('CEU/results/CEU2.faw',comment.char="#")
#Plot Fay and Wu's H
plot(CEUFay[,15] ~ CEUFay[,1],xlim=c(136545410-1e6,136594750+1e6),pch='.',cex=2,
xlab='',ylab="H Statistic")
rect(136545410,-1000,136594750,100,border="Blue")
```

Figure 2 show the Fey and Wu's H statistic one megabase downstream and upstream of the lactase gene

To plot the Fay and Wu's H values for the YRI population.



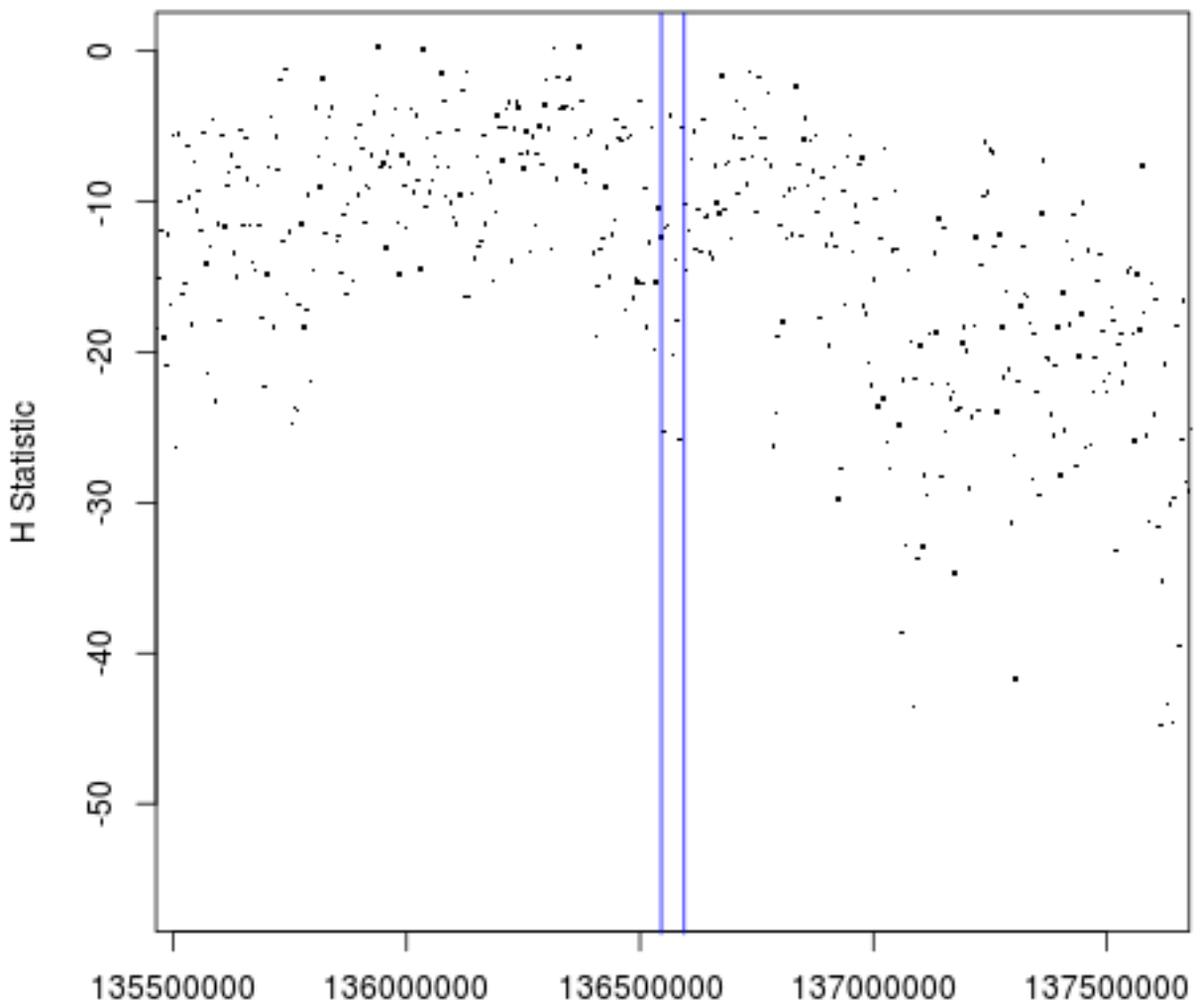


Figure 3: YRI Fay and Wu's H

```
YRIFay=read.table('YRI/results/YRI2.faw',comment.char="#")
#Plot Fay and Wu's H
plot(YRIFay[,15] ~ YRIFay[,1],xlim=c(136545410-1e6,136594750+1e6),pch='.',cex=2,
xlab='',ylab="H Statistic")
rect(136545410,-1900,136594750,100,border="Blue")
```

Figure 3 show the Fey and Wu's H statistic one megabase downstream and upstream of the lactase gene

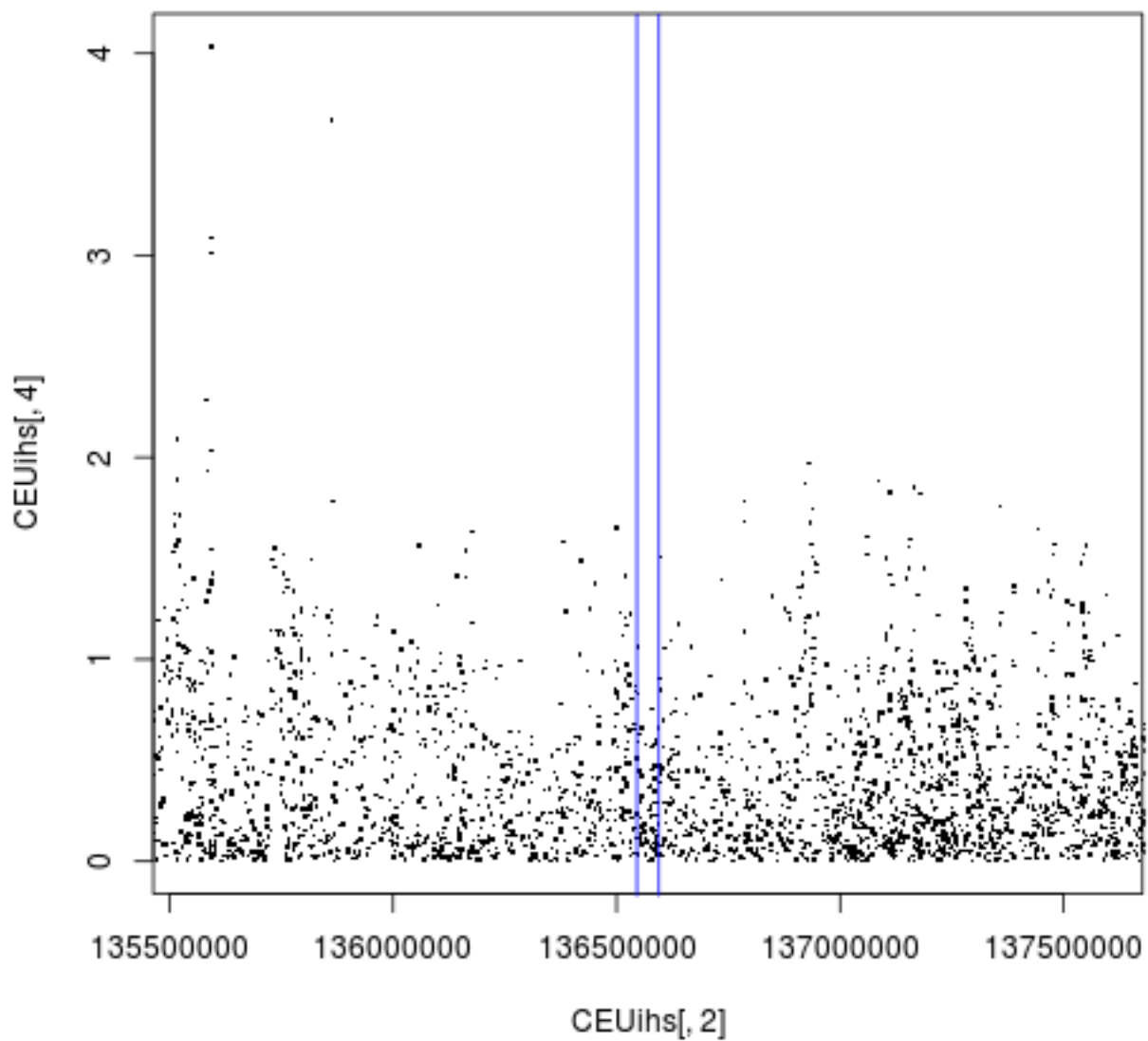


Figure 4: CEU IHS

### 3.5.3 iHS

To plot the iHS values around the lactase gene for the CEU population.

```
CEUihs = read.table('CEU/results/CEUchr2.ihs')
#plot IHS pvalues
plot(CEUihs[,4] ~ CEUihs[,2],xlim=c(136545410-1e6,136594750+1e6),pch='.',cex=2)
rect(136545410,-10,136594750,10,border="Blue")
```

Figure 4 shows iHS pvalues around the lactase gene one megabase upstream and downstream.

To plot the iHS values around the lactase gene for the YRI population.

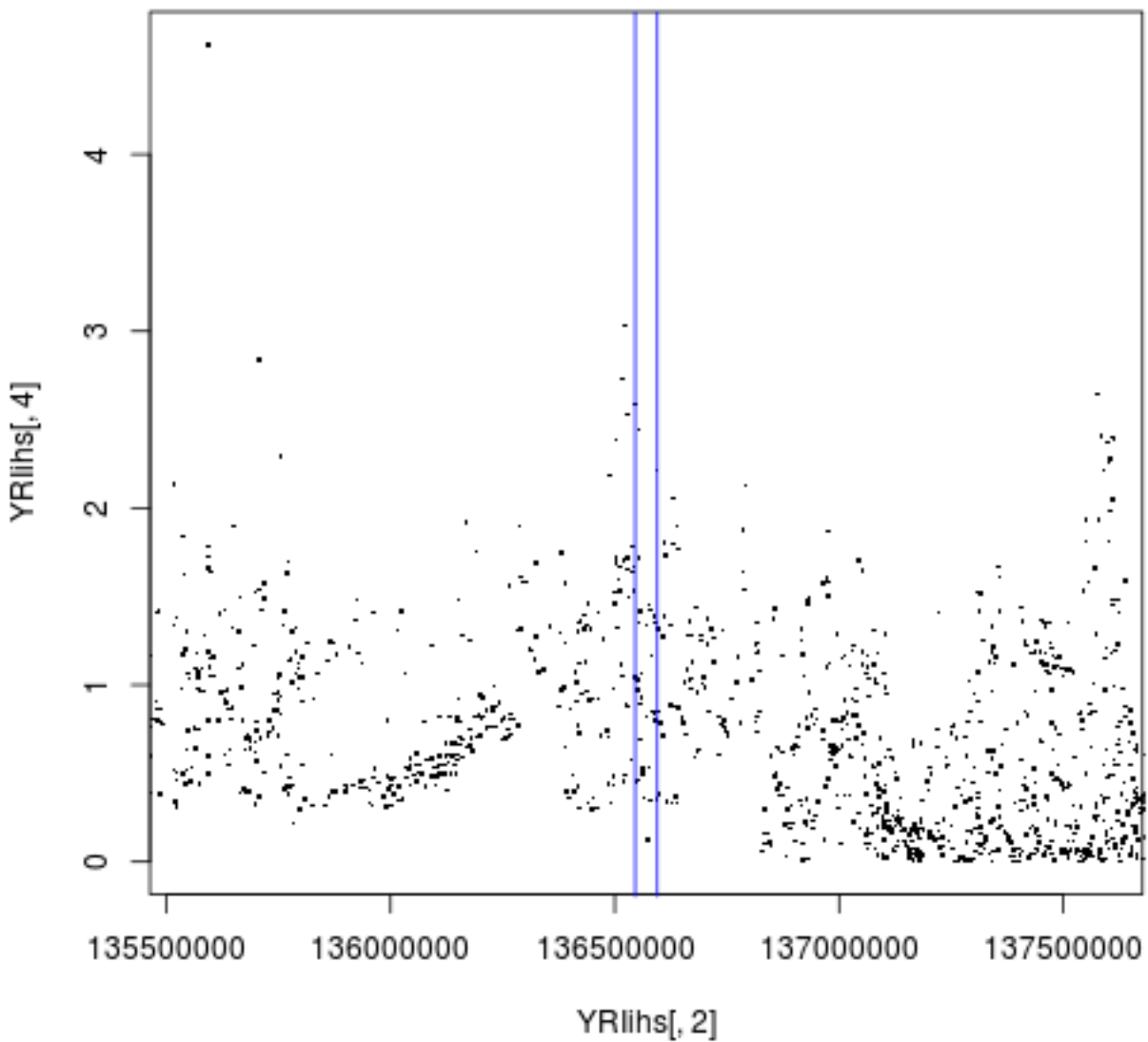


Figure 5: YRI iHS

```
YRIihs = read.table('YRI/results/YRichr2.ihs')
#plot iHS pvalues
plot(YRIihs[,4] ~ YRIihs[,2],xlim=c(136545410-1e6,136594750+1e6),pch='.',cex=2)
rect(136545410,-10,136594750,10,border="Blue")
```

Figure 5 shows iHS pvalues around the lactase gene one megabase upstream and downstream.

To visualize individual snps, a haplotype bifurcation diagram can be used (Gautier and Vitalis, 2012). The snp displayed a strong signal of selection using iHS. We construct a bifurcation diagram for both alleles at this snp.

```
# Plot bifurcation diagram.
```

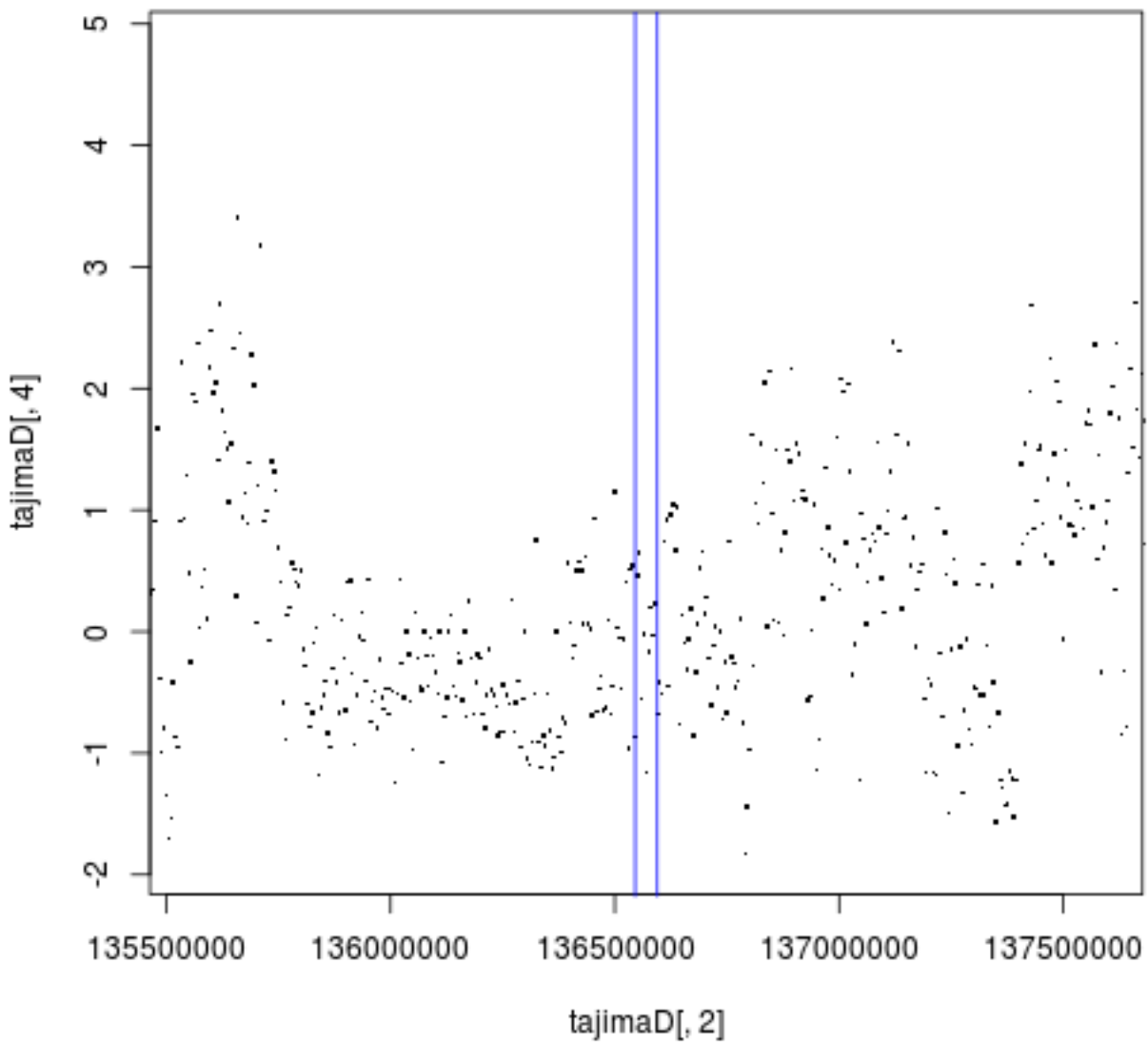


Figure 6: CEU Tajima's D

### 3.5.4 Tajima's D

```
tajimaD=read.table(file="CEU/results/CEU2.taj_d", header=TRUE)
plot(tajimaD[,4] ~ tajimaD[,2],xlim=c(136545410-1e6,136594750+1e6),pch='.',cex=2)
rect(136545410,-10,136594750,10,border="Blue")
```

Figure 6 show the Tajima's D statistic one megabase downstream and upstream of the lactase gene in the CEU population

```
tajimaD=read.table(file="YRI/results/YRI2.taj_d", header=TRUE)
plot(tajimaD[,4] ~ tajimaD[,2],xlim=c(136545410-1e6,136594750+1e6),pch='.',cex=2)
rect(136545410,-10,136594750,10,border="Blue")
```

Figure 7 show the Tajima's D statistic one megabase downstream and upstream of the lactase gene in the YRI population

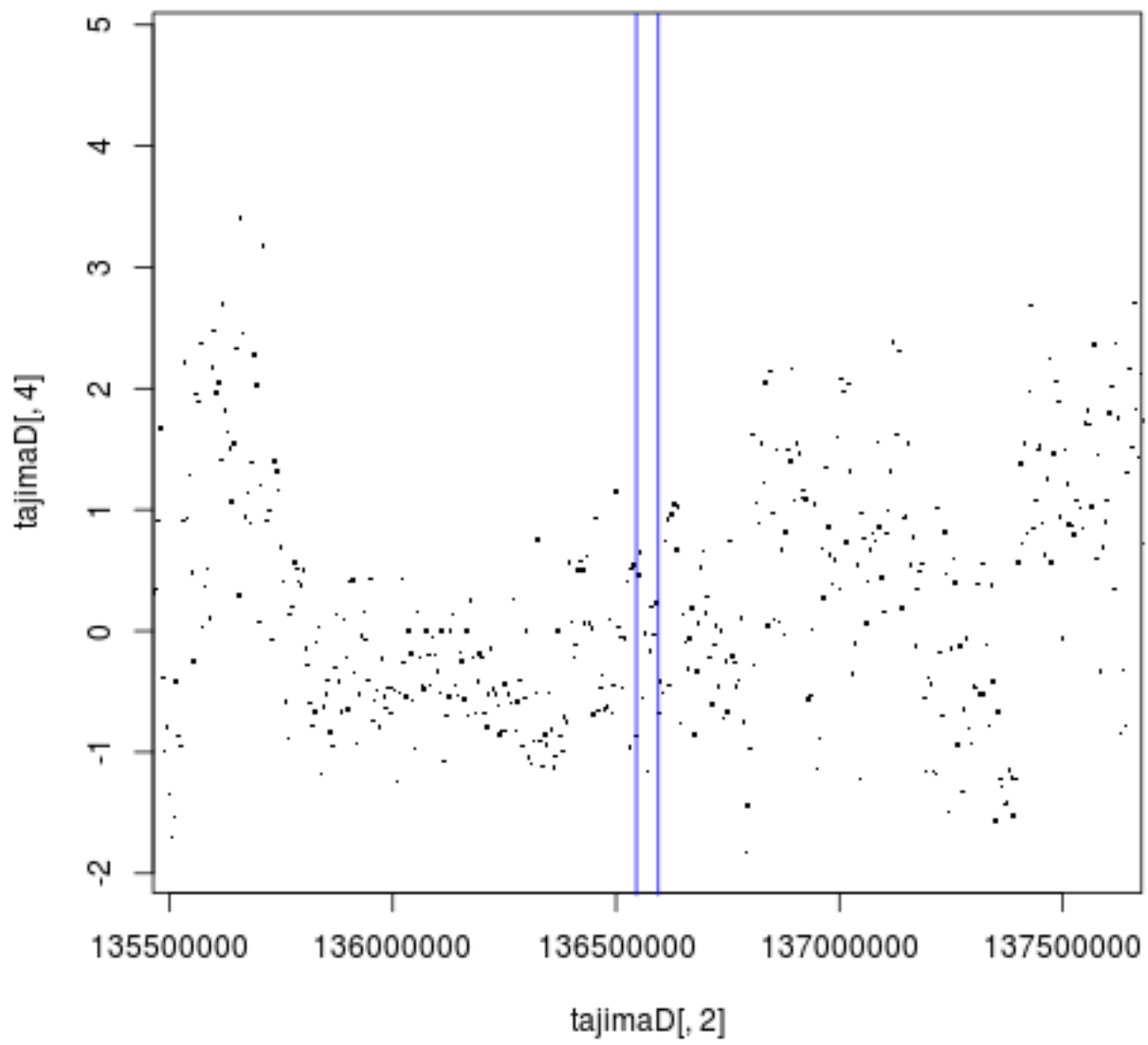


Figure 7: YRI Tajima's D

### 3.5.5 Rsb



## 4 Output Files

The output files are preserved in the same state as the original output from the program used to generate the data.

### 4.1 multi\_population

#### 4.1.1 Fst

Located in the fst folder. Tab-delimited data file containing 1 header line followed data on each subsequent line

CHROM	BIN_START	BIN_END	N_VARIANTS	WEIGHTED_FST	MEAN_FST
2	130000001	130010000	94	0.133102	0.0680276

### 4.2 selection\_pipeline

All the outputs for each population are contained in the results folder. If you ran the tool using *multi\_population* the outputs are located in <pop name>/results.

#### 4.2.1 Fay and Wu's H

Space-delimited data file containing header line that start with a hash character (#). Contains lots of columns if you are only interested in Fay and Wu's H, column 1 provides the position and column 15 provides the H statistic.

#	RefStart	Refend	...	FayWu\_H
130000040	130005039	...	-22.2438460	

#### 4.2.2 iHS

Space-delimited data file containing one header line followed by data on each subsequent line.

"CHR"	"POSITION"	"iHS"	"Pvalue"
"rs4662641"	2	130000272	0.0644902912148128 0.0229261107107533

#### 4.2.3 iHH

Space-delimited data file containing one header line followed by data on each subsequent line.

"CHR"	"POSITION"	"FREQ_a"	"IHHa"	"IHHd"	"IES"
"rs1251176"	2	130000040	0.9823	11558.89	83915.49 11571.13

#### 4.2.4 Tajima's D

Space-delimited data file containing one header line followed by data on each subsequent line.

CHROM	BIN\_START	N\_SNPS	TajimaD
2	130000000	22	0.775224

## 5 Command line Arguments

The selection pipeline contains three programs: *selection\_pipeline*, *aa\_annotate*, and *multipopulation*. The selection pipeline does all the intra-population statistics calculations. The multipopulation program calculates all the inter-population statistics and calls the selection pipeline. The *aa\_annotate* program annotates a haplotype file or a phased vcf file with the ancestral allele from the 6-way EPO alignment, for other species or alternative ancestral annotation the feature will be added promptly.

### 5.1 Multipopulation

#### 5.1.1 Input Files

- `-i <vcf input file>` VCF file containing all the populations you want to analyse from one chromosome or a part of a chromosome only.

#### 5.1.2 Output Files

- FST

Fst results are stored in the `fst` folder with the chromosome number followed by the two populations. e.g `2CEUYRI.fst`

- Selection Pipeline Results

All single population pipeline results are stored in the subdirectory of the population in a folder named `results`. These contain the `iHH`, Tajima's D and a population VCF file.

#### 5.1.3 Other parameters (Compulsory)

- `-l <log_file>`

Name for the log file. Moved into the `logs` folder at the end of program run.

- `-c <Chromosome>`

Integer for the chromosome being used.

- `-a <Arguments to the selection pipeline>`

Quoted string containing any extra arguments to the `selection_pipeline` program. e.g `"-imputation"`

- `-config-file <path to config file>`

Path to the selection pipeline config file an example config file is located in the base directory of the extracted package.

- `-no-clean-up`

Do not clean up intermediate data files.

#### 5.1.4 Other parameters (Optional)

- `-fst-window-size <FST window size>`

Argument is passed directly to the VCF tools command line.

- `-fst-window-step <FST window step>`

Argument is passed directly to the VCF tools command line.

- `-cores`

Number of cores available for the pipeline overrides setting in the config file.

## 5.2 Selection Pipeline

### 5.2.1 Input Files

- `-i <VCF input file>`

Single population single chromosome VCF input file. VCF should be bgzipped and tabix indexed.

### 5.2.2 Output Files

The Results directory contains all the output files.

- `.ihh` file

The outputted iHH data for each SNP

- `.taj_d` file

Tajima's D output

- `.vcf` file

Single population VCF updated by the pipeline, can contain.

### 5.2.3 Other parameters(Compulsory)

- `-config-file <Config File path>`

Path to the selection pipeline config file an example config file is located in the base directory of the extracted package.

### 5.2.4 Other parameters(Optional)

- `-l <log_file>`

Name for the log file. Moved into the logs folder at the end of program run.

- `-maf <minimum MAF>`

Minor allele frequency filter threshold any SNPs below this threshold will be discarded from the analysis.



- `-hwe` <hardy-weinberg minimum p-value>

A hardy weinberg test is performed on every snp any snps failing the test will be discarded.

- `-remove-missing` <Inclusion threshold for missing genotypes>

Inclusion criteria for SNPs with missing data. SNPs with less than this value will be removed from analysis.

- `-TajimaD` <tajimas D bin size>

Tajima's D statistic bin size.

- `-no-clean-up`

Do not clean up intermediate data files

- `-ehh-window-size` Window size for multicore rehh calculations.

- `-ehh-overlap`

Window overlap for multicore rehh calculations.

- `-daf` <Minimum derived allele frequency>

Derived allele frequencies below this minimum will be discarded.

- `-big-gap`

Gap size in kb for not calculating iHH if the gap is too large.

- `-small-gap`

Gap size in kb for applying a penalty to the area calculated by iHH

- `-small-gap-penalty`

Penalty multiplier for intergration step in iHH calculation.  $multiplier/gap\_size * area$  is the formula we use. Setting the multiplier to the same value as the small gap threshold is recommended.

- `-cores`

Number of cores available for the pipeline overrides setting in the config file.

## 5.3 Ancestral Annotation

The program *ancestral\_annotation* is installed on the program path. The program annotates haps and vcfs files with ancestral allele annotation from the 6-way IPO alignment or the human reference genome.

### 5.3.1 Input Files

- `-i` or `-haps` <HAPS File>

Haplotype File (.haps)

- `-v` <Phased VCF file>

Phased VCF file (.vcf), phased VCF genotypes denoted by a bar ( | ) for each sample.

- -a or -aa <Ancestral allele fasta>

Ancestral allele annotation file. Currently only works on a the full 1000 genomes GRCh3764 reference file or the single chromosome fasta files from the 6-way EPO alignment.

### 5.3.2 Output Files

- -o or -output <Output file name>

Output file name optional argument by default output is sent to the stdout stream.

### 5.3.3 Other parameters

- -c <chromosome number>

The number of the chromosome being used.

- -ref-fasta

Denoting that you are using the human reference allele as the ancestral allele.

- -f or -format <format>

The 6-way EPO alignment denotes ancestral alleles with both high and low confidence. To use only ancestral alleles with high confidence use -format high. To use both high and low confident alleles use -format low. By default the program will use only highly confident alleles.

## 5.4 Configuration File

The selection pipeline requires a configuration file, by default the program looks in the current working directory for a file named defaults.cfg but you can point the program to any file using command line argument -config-file <config\_file\_location>. There are two main programs in the selection pipeline namely *selection\_pipeline* and *multi\_population*. These programs share a config file but certain configuration parameters can be omitted when using the *selection\_pipeline* program exclusively. A clean install of the program generates an example configuration file containing default arguments for all the compulsory parameters. The default config file contains an example of the format.

### 5.4.1 system

- cores\_avaliable

Certain programs in the pipeline can take advantage of multicore computers. This option instructs the pipeline about the maximum number of concurrent processes it is allowed to use.

### 5.4.2 environment

- LD\_LIBRARY\_PATH

Set the library path when running the pipeline, this enables the pipeline to use the shared libraries that are used for some programs in the pipeline. (alter this option with caution!)

- PERL5LIB

Sets the PERL5LIB environment variable, this enables the pipeline to use the perl libraries required by VCFTOOLS.  
(alter this option with caution!)

### 5.4.3 selection\_pipeline

- selection\_pipeline\_executable

Points to the location of the selection\_pipeline\_executable.

### 5.4.4 vcf\_tools

- vcf\_tools\_executable

Points to the vcftools executable, by default it points to the vcftools executable installed with the pipeline.

- vcf\_subset\_executable

Points to the vcf-subset executable, by default pointing to the vcf-subset installed with the pipeline.

- vcf\_merge\_executable

Points to the vcf-merge executable, by default pointing to the vcf-subset installed with the pipeline.

- extra\_args

A quoted string containing extra arguments to send to the vcf\_tools executable.

### 5.4.5 shapeit

- shapeit\_executable

Location of the shapeit executable.

- genetic\_map\_dir

Directory containing the genetic map for shapeit.

- genetic\_map\_prefix

The full file for the genetic map files with a "?" character representing the changing chromosome number.

- extra\_args extra arguments to send to shapeit. (Warning: Certain options could potentially break to pipeline use with caution)

### 5.4.6 impute2

- impute\_executable

Location of the impute2 executable

- impute\_map\_dir

Directory containing the genetic map for impute2

- `impute_reference_dir`

Directory containing the reference panel ( `.legend` and `.hap`) files for `impute2`.

- `chromosome_split_size`

Window size for imputation calculation.

- `impute_map_prefix`

The full file name for the genetic map files with a "?" character representing the changing chromosome number

- `impute_reference_prefix`

The full file name for the reference panels minus the extension with a "?" character representing the changing chromosome number.

- `extra_args`

extra arguments to send to `impute2`. (Warning: Certain options could potentially break to pipeline use with caution)

#### 5.4.7 plink

- `plink_executable`

Location of the plink executable

#### 5.4.8 Rscript

- `rscript_executable`

Location of the `rscript` executable. (Program usually on path so just `Rscript` is the default)

- `indel_filter`

Location of the `rscript` `indel_filter` (`hap_indel_and_maf_filter.R`)i

#### 5.4.9 python

- `python_executable`

location of the python executable (2 or 3)

#### 5.4.10 ancestral\_allele

- `ancestral_allele_script`

Location of the `ancestral_annotation` script (`aa_annotate.py`)

- `ancestral_fasta_dir`

Directory containing the ancestral reference files

- `ancestral_prefix`

Full file name for ancestral fasta files containing a "?" character

### 5.4.11 qctool

- qctool\_executable

Location of the qctool executable.

### 5.4.12 multicore\_ihh

- multicore\_ihh

Location of the multicore\_iHH.R script

The rehh package source included with the pipeline has been altered to match the output filters used in Voight's paper. If the  $EHH > 0.05$  reaches the end of a chromosome or the start of a gap  $> \text{big\_gap\_threshold}$ , then no value is returned for the core snp. The `small_gap_threshold` specifies the gap distance to reduce the distance spanned by the gap by a multiplicative factor specified by `small_gap_multiplier`. The formula for the penalty is  $\frac{\text{small\_gap\_multiplier}}{\text{gap\_size}}$ . To match the parameters used by Voight, 200000 should be used for `big_gap_threshold`, 20000 for `small_gap_threshold` and 20000 for `small_gap_multiplier` (Voight *et al.*, 2006).

## 6 Log Files

### 6.1 multi\_population

The location of the log file for *multi\_population* defaults is located in the log directory. Contains all the logging information for the between population selection signature calculations.

### 6.2 selection\_pipeline

The location of the log file for *selection\_pipeline* is located in the log directory. The log file contains all the logging information for the within population selection signature calculations.

## 7 Extra Features

### 7.1 Galaxy Intergration

The galaxy folder contains the scripts required to add the selection pipeline to your local galaxy installation. The pipeline is also available on the galaxy toolshed at `galaxy_url`. To do intergrate the pipeline into galaxy.

## 8 F.A.Q

1. How do I run *multi\_population* with a phased VCF?

In the `-a` argument for *multi\_population* merely add `-phased-vcf` between the quotes this will ensure phasing and imputation will be skipped when *selection\_pipeline* is called.

2. My populations are in separate VCF-Files how do I run *multi\_population*?

To run the pipeline you will need to merge the VCF-files into one large multipopulation VCF file and generate the appropriate population files.

To merge your vcfs you can use the vcf-merge program for this to work correctly outside the selection pipeline you will need to add the following to your .bashrc file.

```
export PERL5LIB=\${PERL5LIB}:<path to selection pipeline>/lib/perl5
```

The command to run vcf-merge is as follows.

```
vcf-merge <vcf1.vcf> <vcf2.vcf> ..... > big\_vcf.vcf
```

### 3. My VCF file is not split by chromosome how do I get my VCF into a single chromosome?

The vcftools program can be used to extract each chromosome from your full vcf file. If you do not have the vcftools program installed the bin/ directory contains exactly what you need. For example for human 1000 genomes data to extract chromosome 2 from your VCF file use the following command.

```
vcf-tools --vcf big\_vcf.vcf --chr 2 --out chr2 --recode
```

The command will generate a vcf file name chr2.recode.vcf containing only data from chromosome 2.

## References

- Danecek P, Auton A, Abecasis G, Albers CA, Banks E, DePristo MA, Handsaker RE, Lunter G, Marth GT, Sherry ST, McVean G, Durbin R, 1000 Genomes Project Analysis Group (2011). “The variant call format and VCFtools.” *Bioinformatics (Oxford, England)*, **27**(15), 2156–2158.
- Delaneau O, Zagury JF, Marchini J (2013). “Improved whole-chromosome phasing for disease and population genetic studies.” *Nature Methods*, **10**(1), 5–6.
- Flicek P, Amode MR, Barrell D, Beal K, Brent S, Carvalho-Silva D, Clapham P, Coates G, Fairley S, Fitzgerald S, Gil L, Gordon L, Hendrix M, Hourlier T, Johnson N, Kähäri AK, Keefe D, Keenan S, Kinsella R, Komorowska M, Koscielny G, Kulesha E, Larsson P, Longden I, McLaren W, Muffato M, Overduin B, Pignatelli M, Pritchard B, Riat HS, Ritchie GRS, Ruffier M, Schuster M, Sobral D, Tang YA, Taylor K, Trevanion S, Vandrovcova J, White S, Wilson M, Wilder SP, Aken BL, Birney E, Cunningham F, Dunham I, Durbin R, Fernández-Suárez XM, Harrow J, Herrero J, Hubbard TJP, Parker A, Proctor G, Spudich G, Vogel J, Yates A, Zadissa A, Searle SMJ (2012). “Ensembl 2012.” *Nucleic acids* . . . .
- Gautier M, Vitalis R (2012). “rehh: an R package to detect footprints of selection in genome-wide SNP data from haplotype structure.” *Bioinformatics (Oxford, England)*, **28**(8), 1176–1177.
- Howie BN, Donnelly P, Marchini J (2009). “A flexible and accurate genotype imputation method for the next generation of genome-wide association studies.” *PLoS Genet.*, **5**(6), e1000529.
- Sabeti PC, Schaffner SF, Fry B, Lohmueller J, Varilly P, Shamovsky O, Palma A, Mikkelsen TS, Altshuler D, Lander ES (2006). “Positive natural selection in the human lineage.” *Science (New York, NY)*, **312**(5780), 1614–1620.
- Tajima F (1989). “Statistical method for testing the neutral mutation hypothesis by DNA polymorphism.” *Genetics*, **123**(3), 585–595.
- Vilella AJ, Blanco-Garcia A, Hutter S, Rozas J (2005). “VariScan: Analysis of evolutionary patterns from large-scale DNA sequence polymorphism data.” *Bioinformatics*, **21**(11), 2791–2793.
- Voight BF, Kudaravalli S, Wen X, Pritchard JK (2006). “A map of recent positive selection in the human genome.” *PLoS biology*, **4**(3), e72.
- Weir BS, Cockerham CC (1984). “Estimating F-statistics for the analysis of population structure.” *evolution*.