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

Murray Cadzow[1], James Boocock[1], Hoang Tan Nguyen[1,2], Phillip Wilcox[1,3], Tony R Merriman[1] and Michael A Black[1]

[1]Department of Biochemistry, University of Otago
[2]Department of Mathematics and Statistics, University of Otago
[3]Scion Research, Rotorua, New Zealand

December 9, 2013

## Contents

## 1 Introduction

**[*** JAMES - WHATS UP WITH THE UGLY GREEN LINKS FOR CITATIONS? ;) ***]**

This selection analysis workflow utilizes genotype data derived from next-generation sequencing (NGS) or high-denisity microarray (e.g., “SNP chip”) experiments to identify the presence of signatures of selection. The tools used to detect selection are dependent on the selection signature being investigated (Sabeti *et al.*, 2006). The pipeline presented here 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 H requires haplotype information, 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 performs phasing if the VCF files do not contain phase information, and then performs ancestral allele annotation. Once complete, the rehh package for R provides a simple interface for implementing EHH-based analyses (Gautier and Vitalis, 2012). Here we have extended rehh to include penalties for gaps that match those used in the original iHS paper (Voight *et al.*, 2006). rehh is used to calculate iHH, iHS, iES and Rsb. To calculate Fay and Wu’s H, a C program, variscan, was utilised (Vilella *et al.*, 2005). The pipeline is implemented in Python, and takes a VCF file as input. The output is a colection of files relating to selection signatures dected by the various software tools.

## 2 Getting Started

### 2.1 Prerequisites

The selection pipeline was developed on a 64-bit ubuntu 13.04 system and has been tested on 64-bit centos and ubuntu 13.10 installations. The pipeline should work on any 64-bit linux derivative assuming some basic libraries and tools are installed on the system. 8GB of RAM should be sufficient for all computation steps (imputation is the most RAM-intensive component of the pipeline).

- Python $>=$ 2.6
- Bourne-again Shell (Bash)
- Perl5
- R >= 3.0.0 (with a personal library) **[*** JAMES - WHAT IS A “PERSONAL LIBRARY”? ***]**
- Build-essential tools **[*** JAMES - SPECIFICALLY...? ***]**

The software is installed with the same permissions as the user than runs the script: if the user is not root then a local R library is required. The program also installs the scripts to the user’s  /.local/bin directory. This directory should be added to the system PATH to give direct access to the programs from the command-line.

## 2.2 Python dependencies

The following Python packages are required for the pipeline:

- python-setuptools
- python-numpy

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

## 2.3 Installation

**[*** JAMES - NEED TO TELL THEM HOW TO GET THE FILES ***]**

To install a standalone version of the pipeline (which will require manual adjustment of the configuration file), run the following command:

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

**[*** JAMES - NEED TO EXPLAIN THE DIFFERENCE BETWEEN MANUAL AND AUTOMATIC INSTALLATION ***]**

The remainder of this section is dedicated to the automatic installation. To perform an automatic installation of the selection analysis pipeline, run the command.

```
./install.sh
```

The installation process creates a default configuration file located in the base directory of the pipeline. It also adds a program called selection_pipeline to the system path. To test that 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, 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 some 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.

**[*** JAMES - IS THAT ENOUGH DETAIL FOR USING OTHER REFERENCE FILES? ***]**

```
...
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 the reference files in another location, further options require alteration in the config file:

```
...
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. Be 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 alter 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

In humans, lactase is encoded by the LCT gene, which is located on Chromosome 2 at the coordinates: 136,545,410-136,594,750. For this example we will use a 10 megabase region containing the LCT, and genotype data from the CEU and YRI populations from the 1000 Genomes Project. In order to demonstrate the functionality of 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 those of European ancestory. **[*** JAMES - CITATION? ***]**.

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

**[*** JAMES - I RECKON THATS NOT THE REAL LOCATION. :) ***]**.

Extract the example data into a new folder.

## 3.2 Setting up the Pipeline Run

## 3.3 Population Files

Population files are required for any cross population comparisions. The commands below will initiate the data generation step. Population files are line seperated files, where the first line contains the population name, and every successive line contains an 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, execute the command below in the folder in which the exaple data were extracted, and change the –config-file parameter to match the location where the pipeline is installed.

```
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 four output files are generated. These contain Tajima's D, iHH, an updated VCF, and Fay and Wu's H statistic. The files are located in the results folder inside each population subfolder. $F_{ST}$ is calculated between each population and results are located in the fst folder. $F_{ST}$ results are calculated using the Weir and Cockerham estimator.

## 3.5 Data Visualisation

The purpose of the analysis pipeline is to generate standard signatures of selection from a VCF formatted input file. In order to assist with exploring/interpretting the results, visualization of the pipeline output can be extrenely useful. The next section describes some basic approaches to plotting these data using the R programming language. All of the following commands are run in a R session with the working directory set as the base directory from which the tutorial is being run. In each of the plots that follow, the vertical blue lines indicate the position of the lactase gene (LCT).

### 3.5.1 Visualizing $F_{ST}$ values

Read in the $F_{ST}$ data:

```
CEUYRIfst=read.table("fst/2CEUYRI.fst", header=TRUE)
```

Plot the $F_{ST}$ results across a region 1 megabase on each side of the lactase gene. Weighted $F_{ST}$ is being plotted in each case.

```
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 $F_{ST}$ values close to the lactase gene plotting one megabase downstream and upstream either side of the gene.

**[*** JAMES - HOW INTERESTING IS THIS? THERE ARE A LOT OF APPARENT SPIKES. :) Also, the axis labels for the graphs could be improved by using the "expression" function in R. Within the plot command, "ylab=expression(F[ST])", will produce a subscript for "ST". Also, a label is needed for the x axis (I'd suggest "Chromosome 2, position (bp)"). ***]**

### 3.5.2 Fay and Wu's H

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 shows the Fay and Wu's H statistic one megabase downstream and upstream of the lactase gene. Plot the Fay and Wu's H values for the YRI population:

```
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 shows Fay and Wu's H statistic one megabase downstream and upstream of the lactase gene.

### 3.5.3 iHS

Plot the iHS values around the lactase gene for the CEU population:

```
CEUihs = read.table('CEU/results/CEUchr2.ihs')
#plot IHS pvalues
plot(CEUihs[,3] ~ 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. Plot the iHS values around the lactase gene for the YRI population:

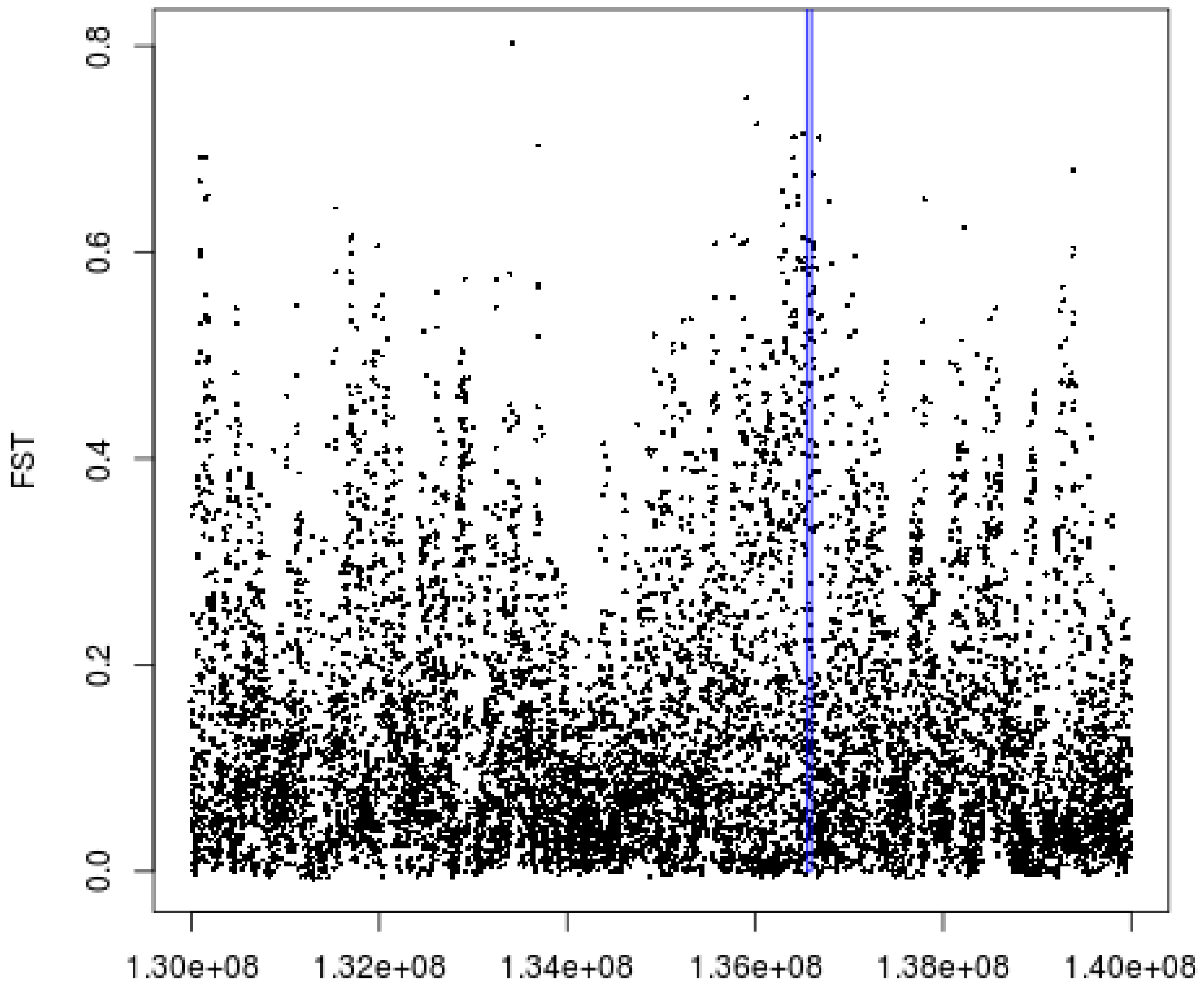

Figure 1: Values of $F_{ST}$ generated for comparing the YRI and CEU populations across a ten megabase region around the LCT gene.

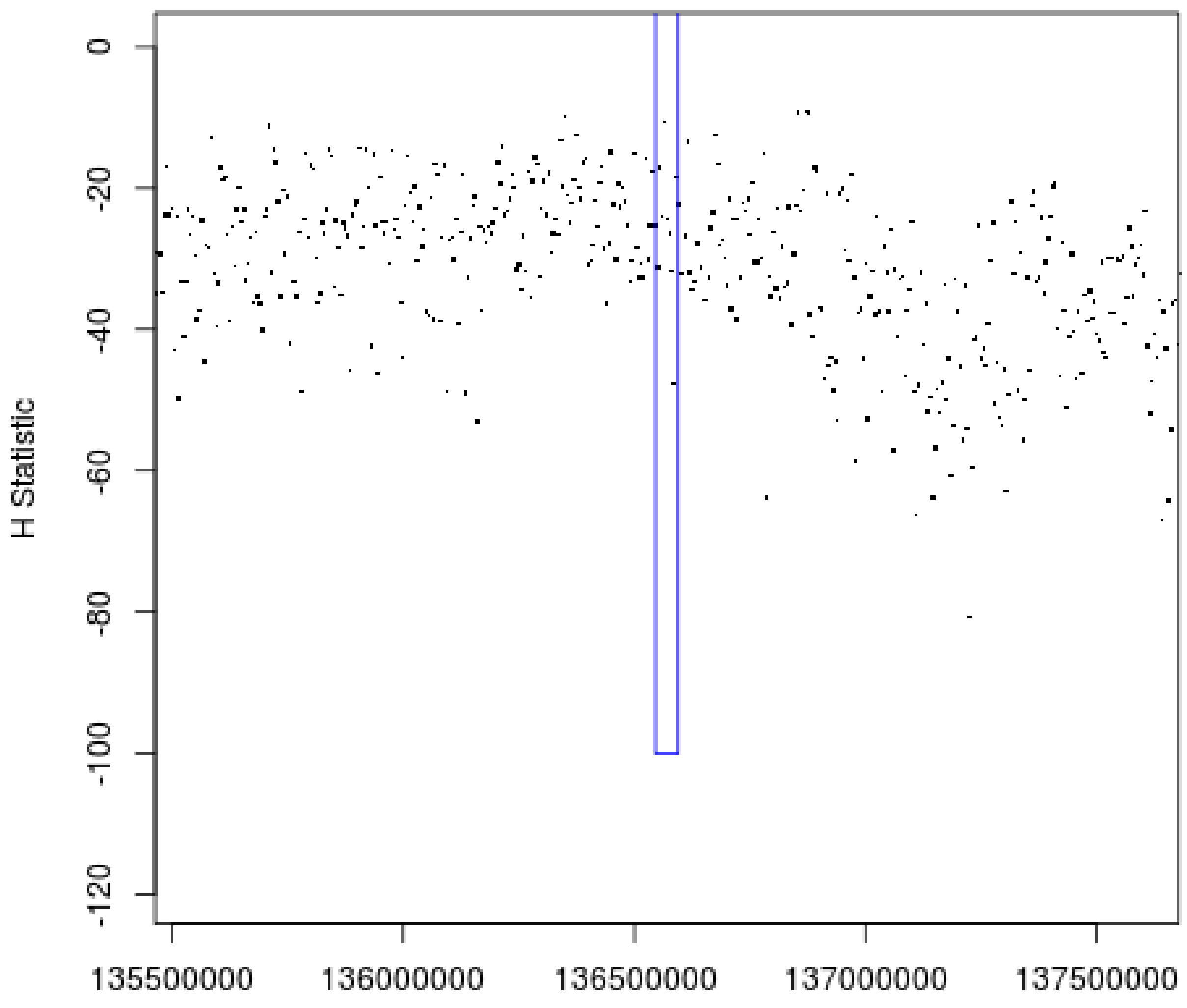

Figure 2: Fay and Wu's H statistic in the CEU population, across a two megabase region around the LCT gene. **[*** NEED X-AXIS LABEL. Also, code for plot has blue rectangle extending down to -1000, but figure only has it to -100: need to update figure. ***]**

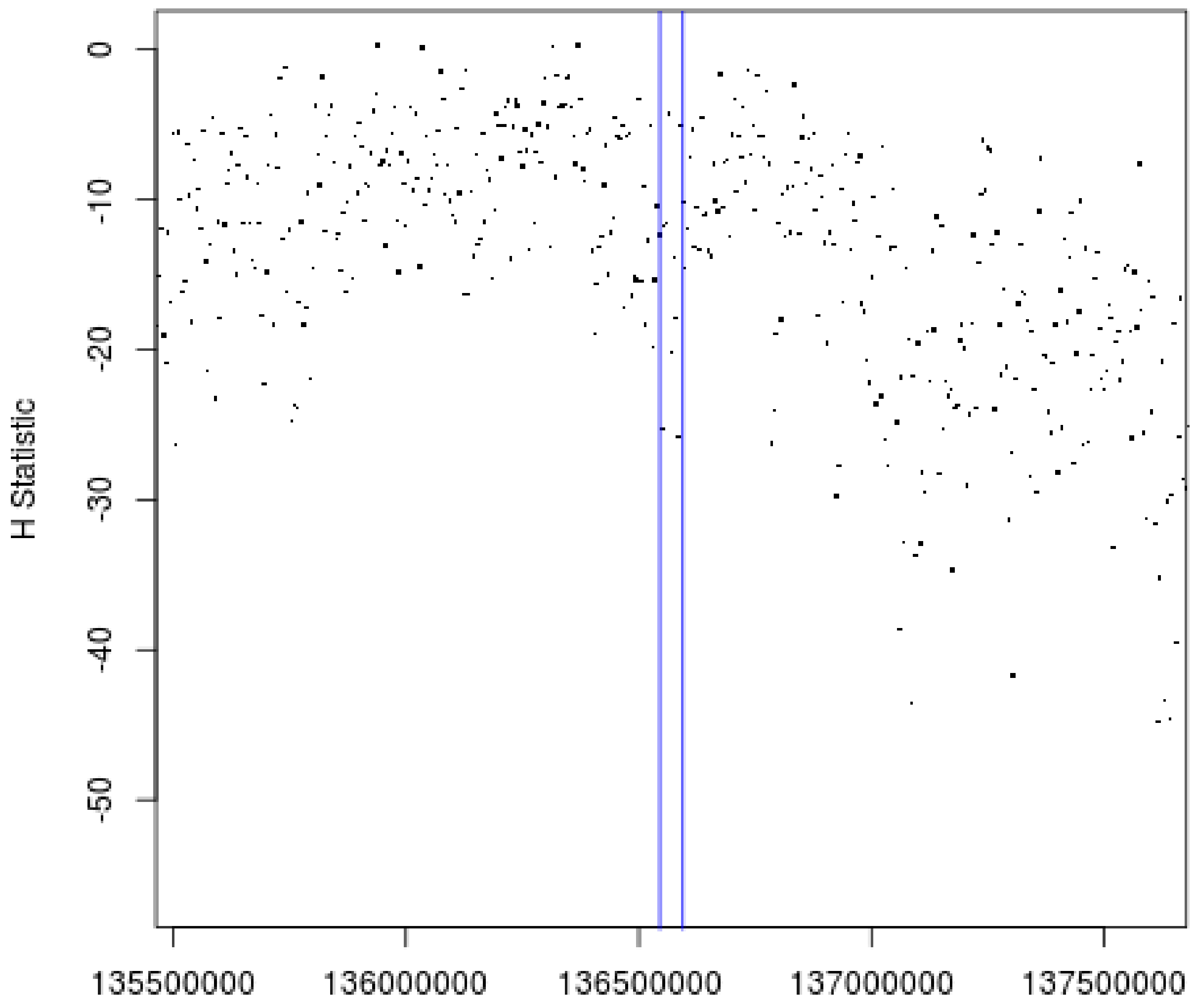

Figure 3: Fay and Wu's H statistic in the YRI population, across a two megabase region around the LCT gene. **[*** NEED X-AXIS LABEL. ***]**

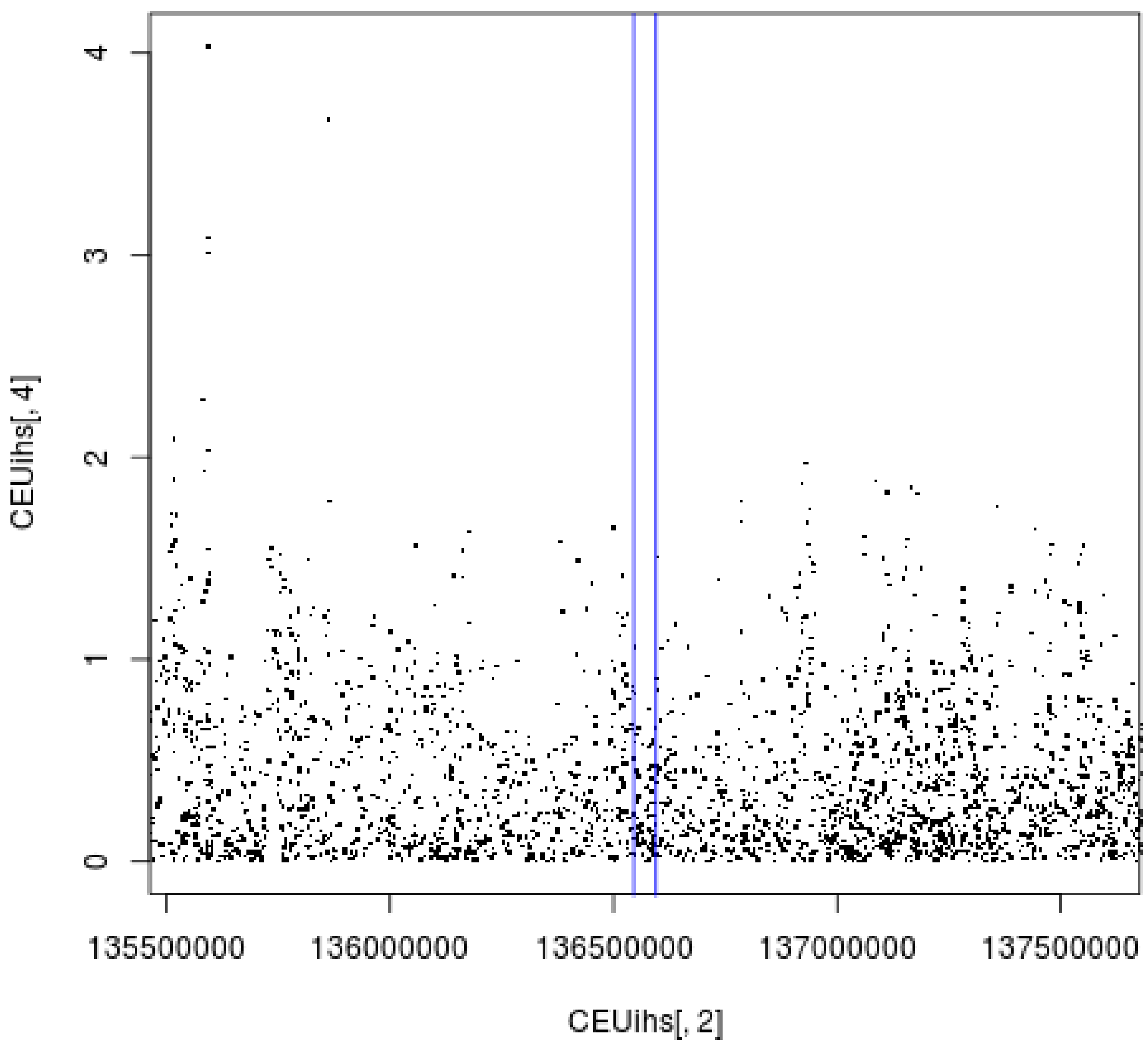

Figure 4: iHS statistic in the CEU population, across a two megabase region around the LCT gene. **[*** NEED BETTER AXIS LABELS ***]**

```
YRIihs = read.table('YRI/results/YRIchr2.ihs')
#plot IHS pvalues
plot(YRIihs[,3] ~ 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 bifurication 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.

**[*** JAMES - STILL NEED CODE FOR THIS (AND A PLOT). :) ***]**

```
# Plot  bifurcation diagram.
```

### 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

### 3.5.5 rSB

**[*** JAMES - WHAT IS HAPPENING HERE? ***]**

**[*** THIS IS AS FAR AS I'VE EDITED (MIK - 1PM, 10 DEC 2013). ***]**

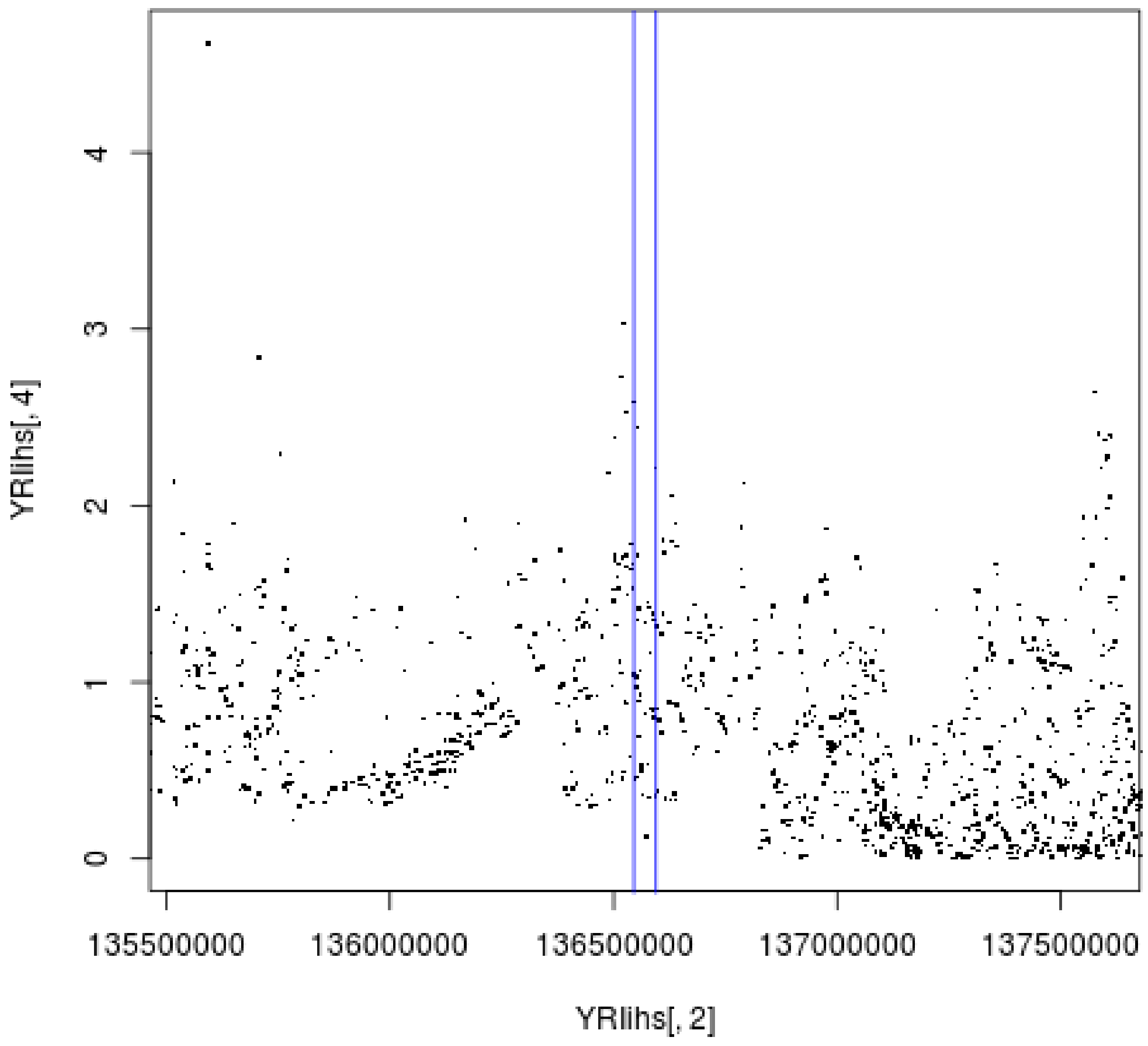

Figure 5: iHS statistic in the YRI population, across a two megabase region around the LCT gene. **[*** NEED BETTER AXIS LABELS ***]**

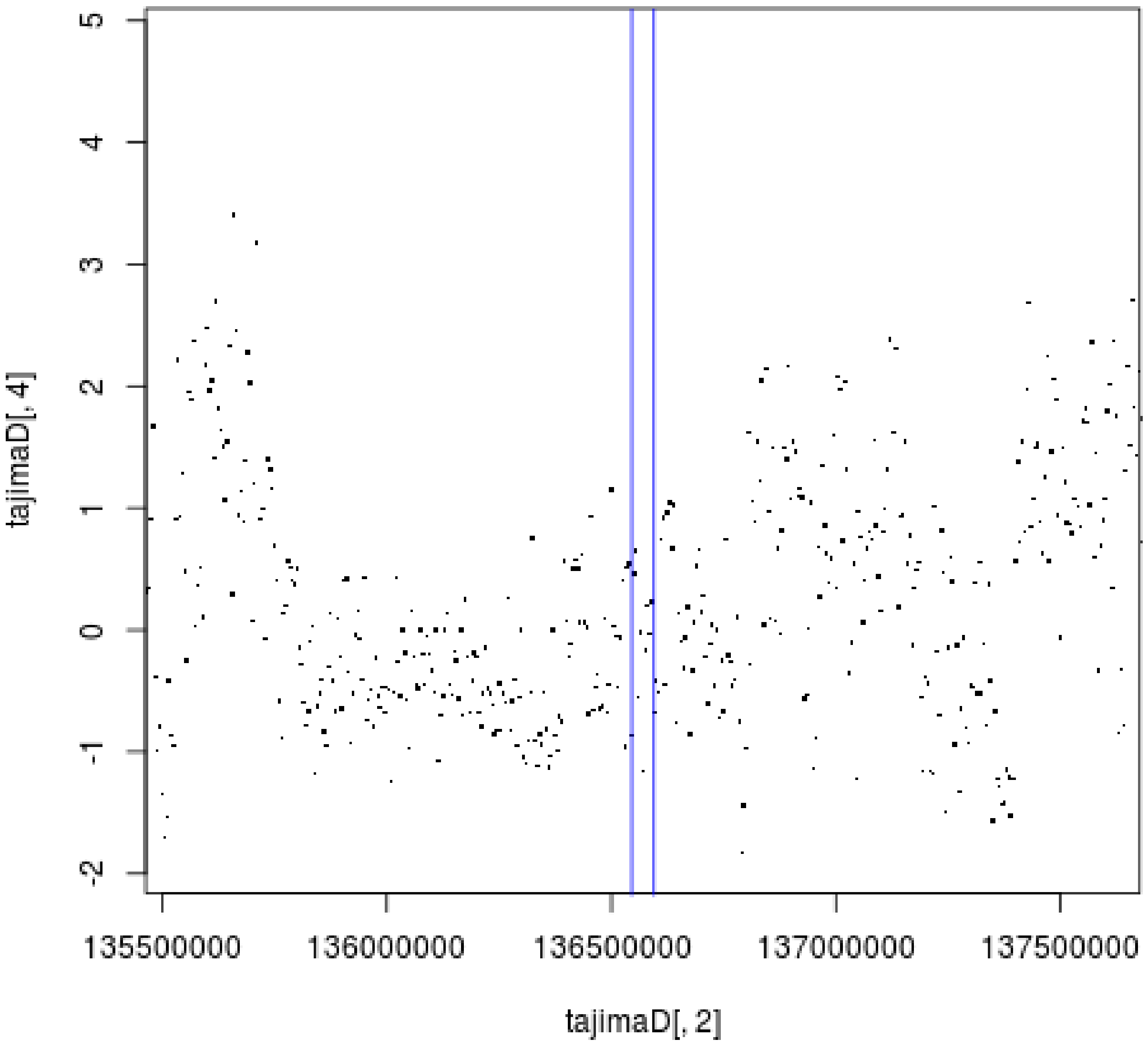

Figure 6: Tajima's D statistic in the CEU population, across a two megabase region around the LCT gene. **[*** NEED BETTER AXIS LABELS ***]**

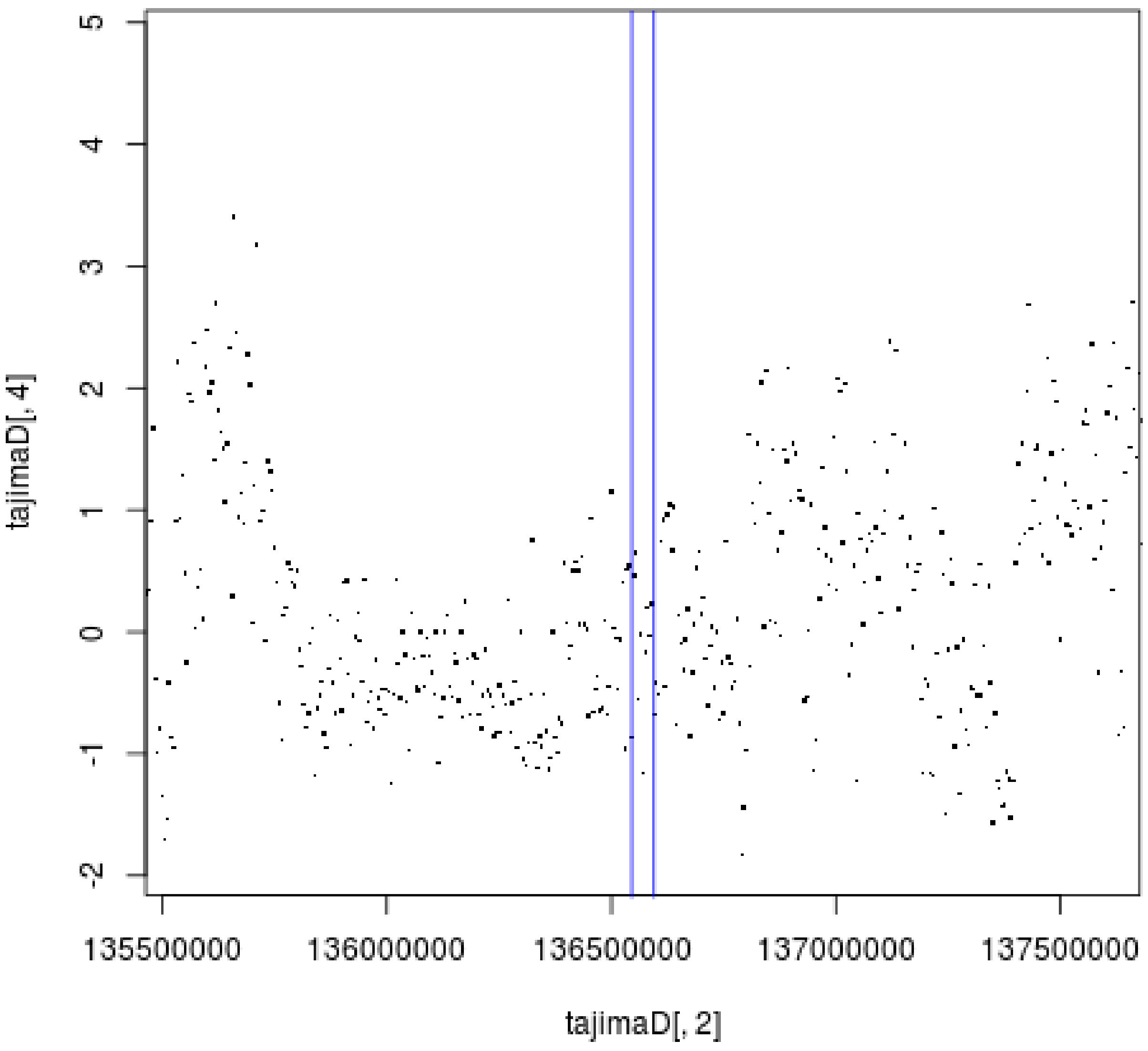

Figure 7: Tajima's D statistic in the YRI population, across a two megabase region around the LCT gene. **[*** NEED BETTER AXISLABELS ***]**

# 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: $s$election_pipeline,$a$a_annotate, and$m$ultipopulation. 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 feauture 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.

### 5.1.4 Other parameters (Optional)

- –no-clean-up

  Do not clean up intermediate data files.

- –fst-window-size <FST window size>

  Argument is passed directly to the VCF tools command line. Default = 1000

- –fst-window-step <FST window step>

  Argument is passed directly to the VCF tools command line. Default = 1000

- –cores

  Number of cores avaliable 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. (Default = 5000)

- –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 avaliable for the pipeline overrides setting in the config file.

## 5.3 Ancestral Annotation

The progam *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 GRCh37$\dot{6}$4 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.

- -s or –sample-file <Sample file output>

  Sample file output name ( currently only works with phased vcf option)

### 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 ommitted 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 $>$ big_gap_threshold , then no value is returded for the core snp. The small_gap_threshold specifies the gap distance to reduce the distance spanned by the gap by a multiplicative factorpecified by small_gap_multiplier. The formula for the penalty is $\frac{small\_gap\_multiplier}{gap\_size}$. 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 avaliable 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 seperate 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.*