populations

The **populations** program will analyze a population of individual samples computing a number of population genetics statistics as well as exporting a variety of standard output formats. A map specifying which individuals belong to which population is submitted to the program and the program will then calculate population genetics statistics such as expected/observed heterozygosity, π, and F_{IS} at each nucleotide position. The **populations** program will compare all populations pairwise to compute F_{ST}. If a set of data is reference aligned, then a kernel-smoothed F_{ST} will also be calculated. The **populations** program can also compute a number of haplotype-based population genetics statistics including haplotype diversity, Φ_{ST}, and F_{ST}'.

Program Options

```
populations -P dir -b batch_id [-O dir] [-M popmap] (filters) [--fstats] [-k [--window_size=15000
populations -V vcf -O dir [-M popmap] (filters) [--fstats] [-k [--window_size=150000] [--bootstra

    -P,--in_path   — path to the directory containing the Stacks files.

    -b,--batch_id   — Batch ID to examine when exporting from the catalog
         (required by -P).

    -V,--in_vcf   — path to an input VCF file.

    -O,--out_path   — path to a directory where to white the output files.
         (Required by -V; otherwise defaults to value of -P.)

    -M,--popmap   — path to a population map. (Format is 'SAMPLE1POP1\n...'.)

    -t,--threads   — number of threads to run in parallel sections of code.

    -s,--sql_out   — output a file to import results into an SQL database.

  Data Filtering:

    -p [int]   — minimum number of populations a locus must be present in to
         process a locus.

    -r [float]   — minimum percentage of individuals in a population required
         to process a locus for that population.

    --min_maf [float]   — specify a minimum minor allele frequency required to
         process a nucleotide site at a locus (0 < min_maf < 0.5).

    --max_obs_het [float]   — specify a maximum observed heterozygosity
         required to process a nucleotide site at a locus.

    -m [int]   — specify a minimum stack depth required for individuals at a
         locus.

    --lnl_lim [float]   — filter loci with log likelihood values below this
         threshold.

    --write_single_snp   — restrict data analysis to only the first SNP per
         locus.

    --write_random_snp   — restrict data analysis to one random SNP per locus.
```

-B — path to a file containing Blacklisted markers to be excluded from the export.

-W — path to a file containing Whitelisted markers to include in the export.

Merging and Phasing:

-e,--renz — restriction enzyme name.

--merge_sites — merge loci that were produced from the same restriction enzyme cutsite (requires reference-aligned data).

--merge_prune_lim — when merging adjacent loci, if at least X% samples posses both loci prune the remaining samples out of the analysis.

Fstats:

--fstats — enable SNP and haplotype-based F statistics.

--fst_correction — specify a correction to be applied to Fst values: 'p_value', 'bonferroni_win', or 'bonferroni_gen'. Default: off.

--p_value_cutoff [float] — maximum p-value to keep an Fst measurement. Default: 0.05. (Also used as base for Bonferroni correction.)

Kernel-smoothing algorithm:

-k,--kernel_smoothed — enable kernel-smoothed π, F_{IS}, F_{ST}, F_{ST}', and Φ_{ST} calculations.

--sigma [int] — standard deviation of the kernel smoothing weight distribution. Default 150kb.

--bootstrap — turn on boostrap resampling for all smoothed statistics.

-N,--bootstrap_reps [int] — number of bootstrap resamplings to calculate (default 100).

--bootstrap_pifis — turn on boostrap resampling for smoothed SNP-based Pi and Fis calculations.

--bootstrap_fst — turn on boostrap resampling for smoothed Fst calculations based on pairwise population comparison of SNPs.

--bootstrap_div — turn on boostrap resampling for smoothed haplotype diveristy and gene diversity calculations based on haplotypes.

--bootstrap_phist — turn on boostrap resampling for smoothed Phi_st calculations based on haplotypes.

--bootstrap_wl [path] — only bootstrap loci contained in this whitelist.

File output options:

--ordered_export — if data is reference aligned, exports will be ordered; only a single representative of each overlapping site.

--genomic — output each nucleotide position (fixed or polymorphic) in all population members to a file (requires --renz).

--fasta — output full sequence for each unique haplotype, from each sample locus in FASTA format, regardless of plausibility.

--fasta_strict — output full sequence for each haplotype, from each sample locus in FASTA format, only for biologically plausible loci.

--vcf — output SNPs in Variant Call Format (VCF).

--vcf_haplotypes — output haplotypes in Variant Call Format (VCF).

--genepop — output results in GenePop format.

--structure — output results in Structure format.

--phase — output genotypes in PHASE format.

--fastphase — output genotypes in fastPHASE format.

--beagle — output genotypes in Beagle format.

--beagle_phased — output haplotypes in Beagle format.

--plink — output genotypes in PLINK format.

--hzar — output genotypes in Hybrid Zone Analysis using R (HZAR) format.

--phylip — output nucleotides that are fixed-within, and variant among populations in Phylip format for phylogenetic tree construction.

--phylip_var — include variable sites in the phylip output encoded using IUPAC notation.

--phylip_var_all — include all sequence as well as variable sites in the phylip output encoded using IUPAC notation.

--treemix — output SNPs in a format useable for the TreeMix program (Pickrell and Pritchard).

Additional options:

-h,--help — display this help messsage.

-v,--version — print program version.

--verbose — turn on additional logging.

--log_fst_comp — log components of F_{ST}/Φ_{ST} calculations to a file.

Population Map

Individuals run through the Stacks pipeline can be split into different population groups for the purposes of computing summary statistics such as π, F_{IS} and F_{ST}. By feeding the populations program different population maps for the same individuals you can split the data in different dimensions. For example, if I collect 25 individuals from one environment, I can run all 25 as a single population, or I could split the population according to phenotypic groups present in that environment by changing the population map. I can also exclude individuals from an analysis, or process subsets of the full dataset by excluding samples from the population map.

Statistics such as F_{IS} and F_{ST} will be computed relative to the population groups. You can have as many groups as you like, summary statistics will be computed for each population, then F_{ST} will be computed across every pair of populations specified. Here are a few examples of a population map for six individuals, in the first case treated as a single population, in the second case treated as two populations. These are simple, tab-separated text files:

A single population in the population map

```
indv_01<tab>1
indv_02     1
indv_03     1
indv_04     1
indv_05     1
indv_06     1
```

Two populations in the population map

```
indv_01     1
indv_02     1
indv_03     1
indv_04     2
indv_05     2
indv_06     2
```

We can also use short strings to represent the populations

```
samp01     red
samp02     red
```

```
samp03      red
samp04      green
samp05      green
samp06      green
```

See the manual for additional information on using population maps.

Whitelists and Blacklists

The **populations** program allows the user to specify a list of catalog locus IDs (also referred to as *markers*) to the program. In the case of a whitelist, the program will only process the loci provided in the list, ignoring all other loci. In the case of a blacklist, the listed loci will be excluded, while all other loci will be processed. These lists apply to the entire locus, including all SNPs within a locus if they exist.

A whitelist or blacklist are simple files containing one catalog locus per line, like this:

```
% more whitelist
3
7
521
11
46
103
972
2653
22
```

SNP-specific Whitelists

In the **populations** program it is possible to specify a whitelist that contains catalog loci *and specific SNPs* within those loci. To create a SNP-specific whitelist, simply add a second column (separated by tabs) to the standard whitelist where the second column represents the *column* within the locus where the SNP can be found. Here is an example:

```
% more whitelist
1916<tab>12
517      14
517      76
1318
1921     13
195      28
260       5
28       44
28       90
5933
19369    18
```

See the manual for additional information on using whitelists.

Bootstrap resampling

The bootstrap resampling procedures are designed to determine the statistical significance of a particular sliding window value relative to the generated empirical distribution. Bootstrap resampling will generate a p-value describing the statistical significance of a particular sliding window and therefore requires a reference genome.

The bootstrap resampling process will center a window on each variable nucleotide position in the population and resample it X times (with replacement), and then calculate a p-value. Bootstrap resampling can be applied to all smoothed values, including the population summary statistics F_{IS}, Π, and haplotype diversity, as well as the calculation of F_{ST} and Φ_{ST} between pairs of populations. If you have tens of thousands of variable sites (not unusual) and lots of populations, this calculation has to be repeated for every variable site in each population to bootstrap the summary statistics and for all variable sites

between each pair of populations for F_{ST} and Φ_{ST}. So, bootstrap resampling can take a while.

Since bootstrapping is so computationally intensive, there are several command line options to the **populations** program to allow one to turn bootstrapping on for only a subset of the statistics. In addition, a bootstrap "whitelist" is available so you can choose to only bootstrap certain loci (say the loci on a single chromosome). This allows one to take the following strategy for bootstrapping to appropriate levels:

1. Bootstrap all loci (for example) to 1,000 repetitions.
2. Identify those loci that are below some p-value threshold (say 0.05).
3. Add these loci to the bootstrapping whitelist.
4. Bootstrap again to 10,000 repetitions (now only those loci in the whitelist will be bootstrapped).
5. Identify those loci that are below some p-value threshold (say 0.005)
6. Add these loci to the bootstrapping whitelist.
7. Bootstrap again to 100,000 repetitions (now only those loci in the whitelist will be bootstrapped).
8. And so on to the desired level of significance...

If instead you are interested in the statistical significance of a particular point estimate of an F_{ST} measure, you will want to use the p-value from Fisher's Exact Test, which is calculated for each variable position between pairs of populations and is provided in the F_{ST} output files.

Example Usage

1. Calculate population statistics in a single population and output a Variant Call Format (VCF) SNP file. Run **populations** on 36 processors:

   ```
   ~/% populations -P ./stacks/ -b 1 --vcf -t 36
   ```

2. Include multiple populations using a population map, and turn on kernel smoothing for π, F_{IS}, and F_{ST}:

   ```
   ~/% populations -P ./stacks/ -M ./samples/popmap -b 1 -k -t 36
   ```

3. Filter input so that to process a locus it must be present in 10 of the populations and in 75% of individuals in each population:

   ```
   ~/% populations -P ./stacks/ -M ./samples/popmap -b 1 -k -p 10 -r 0.75 -t 36
   ```

4. Include a p-value correction to F_{ST} scores, if an F_{ST} score isn't significantly different from 0 (according to Fisher's Exact Test), set the value to 0:

   ```
   ~/% populations -P ./stacks/ -M ./samples/popmap -b 1 -k -p 10 -r 0.75 -f p_value -t 36
   ```

5. Output data for **STRUCTURE** and in the **GenePop** format. Only write the first SNP from any RAD locus, to prevent linked data from being processed by **STRUCTURE**:

   ```
   ~/% populations -P ./stacks/ -M ./samples/popmap -b 1 -k -p 10 -r 0.75 -f p_value -t 36 --st
   ```

6. Include a whitelist of 1,000 random loci so that we output a computationally manageable amount of data to **STRUCTURE**:

   ```
   ~/% populations -P ./stacks/ -M ./samples/popmap -b 1 -k -p 10 -r 0.75 -f p_value -t 36 --st
   ```

   Here is one method to generate a list of random loci from a populations summary statistics file (this command goes all on one line):

```
~/% grep -v "^#" batch_1.sumstats.tsv |
cut -f 2 |
sort |
uniq |
shuf |
head -n 1000 |
sort -n > whitelist.tsv
```

This command does the following at each step:

1. Grep pulls out all the lines in the sumstats file, minus the commented header lines. The sumstats file contains all the polymorphic loci in the analysis.
2. cut out the second column, which contains locus IDs
3. sort those IDs
4. reduce them to a unique list of IDs (remove duplicate entries)
5. randomly shuffle those lines
6. take the first 1000 of the randomly shuffled lines
7. sort them again and capture them into a file.

Other Pipeline Programs

Raw Reads

- process_radtags
- process_shortreads
- clone_filter
- kmer_filter

Core

- ustacks
- pstacks
- cstacks
- sstacks
- genotypes
- populations
- rxstacks

Execution control

- denovo_map.pl
- ref_map.pl
- load_radtags.pl

Utilities

- index_radtags.pl
- export_sql.pl
- sort_read_pairs.pl
- exec_velvet.pl