SpEED-Ne User Guide
March, 2017

SpEED-Ne is a Matlab app and function package to estimate genetic effective population size (Ne) from estimates of two locus disequilibrium (r^2) based on a single sample of genotype data. SpEED-Ne also provides functions that can be used to simulate genotype data and make Ne estimates from simulated data to better understand expected values and distributions of Ne estimates.

SpEED-Ne is Copyright (C) 2017 by Matthew B. Hamilton. The program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version. This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details (http://www.gnu.org/licenses/).

Citation
Hamilton MB, and Battocletti A. SpEED-Ne: Open source software to simulate and estimate genetic effective population size (N_e) from genotypic disequilibrium observed in single samples. To be submitted to Methods in Ecology and Evolution.

Installing the App
In Matlab, click on the Apps tab in the main window. There is an Install App icon at the top left. Click this icon to bring up a file finder dialog where you can select the SpEED_Ne.mlappinstall file.

Once the App is installed its icon will appear under the Apps tab in the main Matlab window.

Running the SpEED-Ne App
To run the App version of SpEED-Ne, click on the icon under the Apps tab. This will bring up an interface window where the input file can be specified and run options can be set.

The *Pick input data file* button bring up a file finder dialog to select an input file of genotype data.

The *minimum allele frequency threshold* field sets the minimum allele frequency (as a proportion). Only alleles with frequencies above this value are used in estimates. This avoids the sometimes extreme estimates of r^2 that result from low frequency alleles. This allele frequency threshold approach is complemented by allele frequency weighting of estimates carried out automatically.

The *Permute genotype data* option is required (the box is always checked) since permutation is one method used to estimate background r^2 from finite sampling and other conditions such as excess homozygosity from null alleles. The number of genotype permutations can be set in the *Genotype data permutations:* field. More permutations are better but are also more time consuming to compute. Simulations validating Ne estimates have employed the default value of 2500 permutations.

The width of confidence intervals (1-α) can be set with the *Total width of confidence intervals* field that sets the value of α.

Several options can be used to provide fuller details of intermediate values used to produce the summary estimates given in the output. These options are useful for verifying input data and to obtain intermediate results to better understand multilocus estimates of Ne. The *Output table of allele states and frequencies* check box will provide a summary of the alleles found at each locus.

The *Output graphs* check box will provide histograms r^2 estimate distributions (without the bias correction of S/(S-1)) for all permuted genotype data sets, all sub-estimates from jackknifing over all pairs of alleles, all sub-estimates from jackknifing over all individuals, and all sub-estimates from jackknifing over all loci.

The check box for *Output r-squared estimated for all pairs of allele within each pair of loci* is helpful to understand the results for each pair of alleles. This option will generate a table for each unique pair loci in the data set.

The *Output CSV text files of estimates and values* check box will generate two output files. One contains all Ne estimates for all unique pairs of alleles within all unique pairs of loci, the other file contains all Ne estimates for all unique pairs of loci.

There are two check box options to generate delete-one jackknife distributions and confidence intervals, one by deleting individuals and one by deleting loci. The distributions and resulting confidence intervals have distinct biological causes and therefore distinct interpretations.

Input data format

SpEED-Ne makes very few assumptions about the format of input data. The data format is simple, making input files easy to prepare. Input files are composed of two columns for each locus, with one column containing the numeric score of the first allele and the second column containing the numeric score of the second allele. Each row contains the genotype information of one individual. The allele scores can be any integer that is appropriate to describe the alleles, such as 1 and 2 to code two SNP locus alleles, or the size in base pairs or the number of repeats for microsatellite loci. For microsatellites, different loci can be scored in both repeats and base pair sizes, such as having one locus scored in repeats and another scored in base pairs.

An example of the input data format for three loci:

10	12	123	127	…	20	21
8	11	-9	-9	…	20	20
…	…	…	…	…	…	…
10	10	123	123	…	-9	21

The example shows missing data coded as -9. Missing data can be given any value of zero or less. Single alleles as well as both alleles in a genotype can be scored as missing. The program has a option for whether or not a scored allele paired with a missing allele will be included in estimates or if the entire genotype will be treated as missing if one allele is scored as missing.

There is no other information in an input file such as lines for text or for locus names.

Input files can be saved as Excel (.xls and .xlsx) files, comma separated value (.csv) files, and as tab or space delimited value (.txt) files. The files are read using the Matlab function *importdata* which can accommodate numerous file formats. See https://www.mathworks.com/help/matlab/import_export/supported-file-formats.html.

The '\data format functions" directory contains functions to read and write genotype data formatted in Genepop and Arlequin (.arp) formats as well as example files. These functions can be employed to convert existing data sets or to export data sets for analysis with other software.

Simulations

SpEED-Ne is a collection of functions that can be used to 1) simulate genotype data, and then 2) estimate Ne using the same methods that are used in the App version. This allows

SpEED-Ne to be used as a tool to better understand bias and variance in estimates of Ne depending on the processing acting to generate the genotypes used to make Ne estimates. This makes SpEED-Ne extensible to study Ne estimates based on r^2 in any new population genetic model or scenario that is coded in Matlab. Using the SpEED-Ne estimation functions should facilitate simulations since only the genotype data generation simulations need to be written – all of the estimation functions can be reused.

Using SpEED-Ne to carry out simulations requires knowledge of programming and of Matlab.

There are several main event loop functions where the functions of SpEED-Ne, along with some additional functions to simulate genotype data, can be used as a simulator. One function is simulate_nulls_r.m which simulates genotype data with null alleles and then estimates Ne for each simulated genotype data set. Another function is simulate_r_fsc.m which iteratively reads in genotype data files generated with fastsimcoal2 and then estimates Ne for each of the data files. Both of these main event loop functions and their required functions can be found in the directory \simulation functions.

The function generate_pop.m simulates genotype data from a population of known Ne.

Each of the functions that are part of the SpEED-Ne App (allele_fre_table.m, alleles.m, comp_hap_table.m, estimate_r_squared.m, jackknife_individuals.m, jackknife_loci.m, make_estimate.m, missing_data_recode_alleles.m, missing_data_recode_genotypes.m, normal_dist_CI.m, percentile_CI.m, permute_data.m, and wd_confidence_intervals_v2.m) has comments describing the input and output data structures. The main event loop for the App is in the function SpEED_Ne.m.

simulate_r.m - Program designed as a simulator that calls functions to estimate r_squared_comp and r_squared_delta using simulated data. This function utilizes the same estimation functions as the GUI version of SpEED-Ne along with generate_pop.m to simulate genetic data with a time-forward algorithm.

simulate_nulls_r.m - Program designed as a simulator that calls functions to estimate r_squared_comp and r_squared_delta using simulated data. This function utilizes the same estimation functions as the GUI version of SpEED-Ne along with generate_pop.m to simulate genetic data with a time-forward algorithm. Null alleles are simulated by calling sim_null_alleles.m after simulated data is generated.

simulate_r_fsc.m - Program designed as a simulator that calls functions to estimate r_squared_comp and r_squared_delta using simulated data. This function utilizes the same estimation functions as the GUI version of SpEED-Ne. This version uses simulated data files generated by fastsimcoal in arlequin format read with function read_arp.m.

generate_pop.m - Function to generate a sample of diploid genotypes from a population with known effective population size. All loci have the same number of alleles, all alleles

have equal initial frequencies, and initial allelic states are adjacent integers. Used by simulate_r.m.

sim_null_alleles.m - Given a data set of diploid genotypes, the function will sample one allele at random with a frequency ≤ max_freq_null at each locus. All heterozygous genotypes containing the null allele are altered to be homozygous for the non-null allele and all genotypes homozygous for the null allele were re-coded as missing data. If a locus has no alleles ≤ pnull the locus is skipped. Used by simulate_nulls_r.m.

read_arp.m - Function to read simulated genetic data in arlequin format and export it as simple two columns per locus format. Used by simulate_r_fsc.m.