ADZE:
Allelic Diversity Analyzer
Version 1.0

Zachary A. Szpiech[1]
Bioinformatics Program
University of Michigan

Mattias Jakobsson
Bioinformatics Program
Dept. of Human Genetics
University of Michigan

Noah A. Rosenberg
Bioinformatics Program
Dept. of Human Genetics
University of Michigan

October 6, 2008

The *ADZE* software is available at
`http://rosenberglab.bioinformatics.med.umich.edu/adze.html`

[1]Comments on *ADZE* can be sent to szpiechz@umich.edu

Contents

1 Introduction

The analysis of the distributions of alleles across populations is important for elucidating genetic diversity and population relationships. Two fundamental quantities for a population at a given locus are the number of distinct alleles in the population and the number of alleles private to the population (that is, not found in other populations). These quantities are especially informative when populations are studied for highly variable multiallelic markers, such as microsatellites.

The number of distinct alleles and the number of private alleles depend heavily on sample size, and they can be difficult to interpret when sample sizes differ across populations. The rarefaction approach has been an important strategy for producing estimates that are comparable in different populations (Hurlbert, 1971; Petit *et al.*, 1998; Kalinowski, 2004, 2005). The idea of the rarefaction method is to trim unequal samples to the same standardized sample size, a number less than or equal to the smallest sample size across populations. For a standardized size g, populations are compared by considering the estimates of "allelic richness" and "private allelic richness" that would be obtained when averaging across all subsamples of size g. In the rarefaction framework, the estimated allelic richness of a population is the number of distinct alleles expected in a random subsample of size g drawn from the population (Hurlbert, 1971; Petit *et al.*, 1998). The estimated private allelic richness is the number of private alleles expected in the population when random subsamples of size g are taken from each of J populations under consideration (Kalinowski, 2004). Combinatorial formulas make it possible to compute these statistics relatively quickly.

Often, as was noted by Kalinowski (2004), especially if several populations in a sample are closely related, few alleles are private to individual populations. Instead, alleles may be private to groups of populations—that is, alleles may be found in each of several members of a larger set of populations. We therefore introduce a generalization of the private allelic richness concept of Kalinowski (2004). We compute a generalized private allelic richness statistic that uses a rarefaction approach to measure the number of distinct alleles private to a *group* of populations and found in *all* populations in the group. This statistic makes it possible to evaluate the sample size-corrected number of distinct alleles private to any set of populations, and it reduces to private allelic richness when the group of populations consists of only a single population. We demonstrate the application of the new generalized private allelic richness statistic using microsatellite genotypes from human populations. By considering the sample size-corrected number of distinct alleles private to various combinations of major geographic regions, this analysis produces evidence in support of the hypothesis that an early human migration from Africa to Oceania did not have an appreciable effect on genetic variation in modern populations of Asia. We have developed a computer program, *ADZE*—a tool for "chopping" samples down to standardized sizes for data analysis—that implements

computations of allelic richness, private allelic richness, and our new measure of generalized private allelic richness.

1.1 Allelic richness

Consider a locus with I distinct alleles, and define N_{ij} as the number of copies of allele type i in a sample from population j. $N_j = \sum_{i=1}^{I} N_{ij}$ is the sample size of population j at the locus. The probability of finding no copies of allele type i in a subsample of size g alleles from population j is

$$Q_{ijg} = \frac{\binom{N_j - N_{ij}}{g}}{\binom{N_j}{g}}. \tag{1}$$

Then the probability of finding at least one copy of allele type i in a sample of size g alleles from population j is $P_{ijg} = 1 - Q_{ijg}$, and

$$\hat{\alpha}_g^{(j)} = \sum_{i=1}^{I} P_{ijg} \tag{2}$$

is the estimated allellic richness of a sample of size g from population j (Hurlbert, 1971; Petit *et al.*, 1998; Kalinowski, 2004). Equation 2 estimates the expected number of distinct alleles that will be observed in population j in a sample of size g.

1.2 Private allelic richness

Using this notation, the estimated private allelic richness for a sample size g from population j can be written as

$$\hat{\pi}_g^{(j)} = \sum_{i=1}^{I} \left[P_{ijg} \left(\prod_{\substack{j\prime=1 \\ j\prime \neq j}}^{J} Q_{ij\prime g} \right) \right], \tag{3}$$

where J is the total number of populations (Kalinowski, 2004). This formula sums over distinct allele types, i, the probability that a random subsample of size g from population j contains allele type i and that subsamples of size g from the remaining populations do not contain i.

1.3 Private allelic richness for a combination of populations

Generalizing the concept of private allelic richness, we can consider the number of distinct alleles private to some combination of k populations selected from $\{1, 2, \ldots, J\}$. Consider a set of J populations labeled 1 to J, and let $\mathcal{S} = \{1, 2, \ldots, J\}$. Let $\mathcal{C}_k$ be the set of all possible combinations of k elements from $\mathcal{S}$. There are $\binom{J}{k}$ possible combinations in $\mathcal{C}_k$. We label these combinations by $\mathcal{C}_{km}$, where m ranges from 1 to $\binom{J}{k}$. Using the following equation we can

calculate $\hat{\pi}_{gk}^{(m)}$—the estimated number of distinct alleles private to the mth combination of k populations, when samples of size g are drawn from each of the J populations:

$$\hat{\pi}_{gk}^{(m)} = \sum_{i=1}^{I} \left[\left(\prod_{j \in \mathcal{C}_{km}} P_{ijg} \right) \left(\prod_{j\prime \in \mathcal{S} \backslash \mathcal{C}_{km}} Q_{ij\prime g} \right) \right]. \tag{4}$$

$\mathcal{S}\backslash\mathcal{C}_{km}$ denotes the set $\mathcal{S}$ excluding the elements of $\mathcal{C}_{km}$. For $k = 1$, $\hat{\pi}_{gk}^{(m)}$ reduces to private allelic richness as in Equation 3. For $k = J - 1$, Equation 4 can be considered a measure for "missing allelic richness", and it reduces to

$$\hat{\mu}_{g}^{(j)} = \sum_{i=1}^{I} \left[Q_{ijg} \left(\prod_{\substack{j\prime=1 \\ j\prime \neq j}}^{J} P_{ij\prime g} \right) \right]. \tag{5}$$

In this equation, $\hat{\mu}_{g}^{(j)}$ gives a sample size-corrected measure of the number of distinct alleles found in all populations other than population j.

1.4 Summary statistics

ADZE implements these calculations (Equations 2–4) at each of many loci in a data set, and its primary method of reporting results gives the mean, variance, and standard error of the mean across these loci. Section 4.3 has information on how to output locus-specific estimates in addition to the summary statistics. The mean across loci is calculated at each g and for each population (or population grouping) in the following manner:

$$\bar{x} = \frac{1}{L} \sum_{\ell=1}^{L} x_{\ell}, \tag{6}$$

where L is the total number of loci used in the calculations, and x_{ℓ} is one of the statistics described in the above sections calculated at locus ℓ. Variance across loci is calculated at each g and for each grouping as

$$s^2 = \frac{1}{L-1} \sum_{\ell=1}^{L} \left(x_{\ell} - \bar{x} \right)^2, \tag{7}$$

and standard error is calculated at each g and for each grouping as

$$S_E = \frac{s}{\sqrt{L}}. \tag{8}$$

2 Getting started

Executables are available for *ADZE* for Linux and Windows (source code available upon request). The *datafile* is expected to be in the same format as used by *structure* (see section 3.2).

2.1 Availability

Pre-compiled executables for Linux (32-bit and 64-bit) and Windows (32-bit only) are available at:
`http://rosenberglab.bioinformatics.med.umich.edu/adze.html`
Use the following citation for *ADZE*.

ADZE: A rarefaction approach for counting alleles private to combinations of populations Zachary A. Szpiech, Mattias Jakobsson, and Noah A. Rosenberg. Bioinformatics 2008; doi: 10.1093/bioinformatics/btn478

2.2 Installing *ADZE*

2.2.1 Linux

Open a terminal and move to the location of the `.tar.gz` file. Extract it by typing: `tar -xzvf adze-1.0-XX.tar.gz`, where XX is the appropriate architecture. This will create a new directory called `ADZE-1.0` containing the executable.

2.2.2 Windows

Extract the file `adze-1.0-win32.zip`. This will create a directory called `ADZE-1.0` containing the executable.

2.3 Running *ADZE*

When executing *ADZE*, the *datafile* specified in the *paramfile* must be in the same directory as the *ADZE* executable, or the whole path must be specified in the *paramfile*.

2.3.1 Linux

Open a terminal and path to the directory containing *ADZE*. Execute the program by typing `./adze-1.0 paramfile` where *paramfile* is the name of your parameter file. You must specify the whole path if your *paramfile* is located in another directory. If no *paramfile* is specified, *ADZE* will search in the current directory for a *paramfile* named "paramfile.txt". If this file

is not found, the program will create an empty template *paramfile* named "paramfile.txt" in the current directory and will then exit with an error message.

2.3.2 Windows

Open a command prompt by going to the START menu, clicking on RUN, and typing `cmd` or `command`. Specify the path to the directory containing the *ADZE* executable, and execute the program by typing `adze-1.0 paramfile`, where *paramfile* is the name of your parameter file. If your *paramfile* is located in another directory you must specify the whole path. If no *paramfile* is specified, *ADZE* will search in the current directory for a *paramfile* named "paramfile.txt". If this file is not found, the program will create an empty template *paramfile* named "paramfile.txt" in the current directory and will then exit with an error message. The program may also be executed by double clicking on the *ADZE* icon; however, there must be a valid *paramfile* named "paramfile.txt" in the same directory or a blank template will be created. No command line arguments can be passed when *ADZE* is run this way.

2.4 *ADZE* running time and memory usage

The running time of *ADZE* is highly dependent on the size and complexity of the *datafile*, the specified parameters, and the computer system on which it is run. *ADZE* will process a data set of ∼500 loci and ∼1000 diploid individuals on a 64-bit 2.4 GHz machine in seconds when skipping the calculation of private alleles to combinations of populations. However when calculating the private alleles for combinations of populations, the calculations for all combinations of size k will be proportional to $\binom{J}{k}$, which has the possibility of quickly growing to an astronomical running time. Additionally, *ADZE* allocates an amount of memory approximately equal to ten times the size of the *datafile*.

3 Input files

ADZE takes two files as input—the *paramfile* and the *datafile*. These are described below.

3.1 *paramfile*

The *paramfile* contains all of the parameters that *ADZE* needs in order to run. The default *paramfile* is "paramfile.txt" and does not need to be entered in at the command line unless other command line flags are used. Any alternative *paramfile*s need to be specified at the command line as outlined in section 4.4. When writing a *paramfile* the '#' character can be used to comment out the remainder of a line. All whitespace is completely ignored (except newlines) when parsing the *paramfile*. For example, the following parameter specifications are

equivalent,

```
MAX_G 70
MAX_G70
MAX_G 7 0
```

as are,

```
R_OUT richness_out
R_OUTrichness_out
R_OUT richness _ out
```

This is not, however, valid for command line flags (see section 4.4). To create an empty template *paramfile*, run *ADZE* when the default *paramfile* does not exist and without any command line arguments. The following parameters need to be explicitly defined by the user for *ADZE* to run: *MAX_G, DATA_LINES, LOCI, NON_DATA_ROWS, NON_DATA_COLS, GROUP_BY_COL, DATA_FILE, R_OUT, P_OUT*. The following parameters do not need to be explicitly defined for *ADZE* to run (although specifying certain parameters may require that others also be specified): *MISSING, TOLERANCE, COMB, K_RANGE, C_OUT, FULL_R, FULL_P, FULL_C, PRINT_PROGRESS*. For more information on these parameters see section 4.

3.2 *datafile*

The *datafile* must be in *structure* format (Figure 1). The first line is a listing of locus names optionally followed by a specified number of non-data lines. Then each individual is printed on two consecutive lines (for haploid or polyploid data, the number of lines per individual may differ). Immediately preceeding the data on every line is a specified number of columns that contain classifications of the individual. The file in Figure 1 contains 5 columns before the genotype data. In this particular example, these columns represent (1) individual code numbers, (2) population code numbers, (3) population names, (4) countries of origin, and (5) geographic regions of origin. The default missing data value is *-9* but can be changed by setting *MISSING* in the *paramfile* or *-m* on the command line (section 4.3). The *datafile* is specified in the *paramfile* and must be in the same directory as *ADZE*; otherwise the whole path to the *datafile* must be provided. Allele representation is not limited to numbers: any string void of whitespace can serve to represent an allele.

4 Usage options

The user can specify eighteen parameters in the *paramfile*. Eight of these must be explicitly defined for *ADZE* to function properly: these are described in section 4.1. There are three additional parameters that are mutually dependent, as described in section 4.2. The remaining

```
D12S1638 D14S1007 D9S1779 D9S1825 D7S2477
1037 87 Pima Mexico AMERICA 126 128 124 -9 142
1037 87 Pima Mexico AMERICA 126 128 142 -9 142
1038 87 Pima Mexico AMERICA 120 124 124 129 142
1038 87 Pima Mexico AMERICA 126 128 152 137 142
1039 87 Pima Mexico AMERICA 120 124 124 129 142
1039 87 Pima Mexico AMERICA 126 128 124 133 142
1040 87 Pima Mexico AMERICA 126 124 124 133 142
1040 87 Pima Mexico AMERICA 126 128 142 137 142
1357 24 Basque France EUROPE 126 120 124 -9 154
1357 24 Basque France EUROPE 128 122 126 -9 156
1358 24 Basque France EUROPE 126 128 124 -9 156
1358 24 Basque France EUROPE 126 128 124 129 156
1359 24 Basque France EUROPE 120 124 124 129 156
1359 24 Basque France EUROPE 126 126 134 131 168
747 684 Japanese Japan EAST_ASIA 126 -9 124 129 158
747 684 Japanese Japan EAST_ASIA 128 -9 144 135 158
748 684 Japanese Japan EAST_ASIA 124 124 124 137 142
748 684 Japanese Japan EAST_ASIA 126 128 140 137 156
749 684 Japanese Japan EAST_ASIA 124 124 124 137 142
749 684 Japanese Japan EAST_ASIA 126 136 142 139 160
750 684 Japanese Japan EAST_ASIA 126 116 126 135 156
750 684 Japanese Japan EAST_ASIA 126 124 144 137 164
```

Figure 1: A small *datafile* in *structure* format

parameters are defined in section 4.3. Additionally, *ADZE* takes nineteen optional command line arguments, which are described in section 4.4.

4.1 Main parameters

The following parameters are necessary and need to be explicitly defined.

G (int) The maximum standardized sample size to calculate. *ADZE* will calculate allelic richness for a group j at each g from 2 to $\min\{G, N_j\}$ and private allelic richness at each g from 2 to $\min\{G, N_1, N_2, \ldots, N_J\}$, where $N_1, N_2, \ldots, N_J$ are the sample sizes of the groups defined by *GROUP_BY_COL*.

DATA_LINES (int) The number of rows of data in the *datafile* (not inclusive of the row containing names of loci). For diploid individuals this number is $2N$, where N is the number of individuals in the data set. This value is used to error-check against the *datafile*.

LOCI (int) The number of loci in the data set. This value is used to error-check against the *datafile*.

NON_DATA_ROWS (int) The number of non-data rows that precede the data and should be ignored. This parameter is always at least 1, indicating a row for loci names. Loci names are always assumed to be on the first row. This value is used to error-check against the *datafile*.

NON_DATA_COLS (int) The number of non-data columns that preceed the data. For example, in Figure 1, `NON_DATA_COLS 5` is the correct definition. This value is used to error-check against the *datafile*.

GROUP_BY_COL (int) The non-data column by which to classify the individuals. This parameter must be $\leq NON\_DATA\_COLS$ and ≥ 1. In Figure 1, to classify by population name, set `GROUP_BY_COL 3` in the *paramfile*.

DATA_FILE (string) The name of the *datafile* (possibly including the path to *datafile*, if the file is not in same directory as *ADZE*).

R_OUT (string) The name of the file to which the allelic richness summary statistics—mean, variance, and standard error of the mean across loci—will be output. See section 5.1 for more details.

P_OUT (string) The name of the file to which the private allelic richness summary statistics—mean, variance, and standard error across loci—will be output. See section 5.1 for more details.

4.2 Combination parameters

The following parameters are necessary for calculating private allelic richness for a combination of populations (see section 1.3).

COMB (boolean) Set to 1 to calculate the private richness of the combinations defined at *K_RANGE*. If set to 1, *K_RANGE* and *C_OUT* must be defined. The default setting is 0.

K_RANGE (string) A string specifying the sizes of combinations to calculate. All values must be ≥ 1 and less than or equal to the number of divisions of the data into groups (e.g. the number of populations). Note that calculations for combinations of size 1 are identical to calculations of private allelic richness. Integer ranges may be specified with the '-' character.

Integers and integer ranges must be delimited by commas. Duplicate numbers will be removed. Examples of valid specifications (without curly braces) include: {*2*},{*2,5,7*},{*2,4-6*}. These examples would ask for *ADZE* to calculate private richness for all combinations of 2 groups; all combinations of 2 groups, 5 groups and 7 groups; and all combinations of 2 groups, 4 groups, 5 groups, and 6 groups.

C_OUT (string) The name of the file to which the private allelic richness summary statistics for the specified combinations—mean, variance, and standard error across loci—will be output. The string '_#' (where # is the combination size k being calculated) will be concatenated onto the end of the file name. For multiple-combination calculations, multiple files will be created. See section 5.2 for more details.

4.3 Advanced parameters

These parameters allow for addtional output, filtering of loci with large amounts of missing data, or definition of an alternate representation of missing data.

MISSING (string) A string defining how missing data is represented in the *datafile*. The default setting is *-9*.

TOLERANCE (double) A value ≥ 0 and ≤ 1 that specifies the largest allowed fraction of missing data at any given division for a locus. Any locus with at least one division that has more than this fraction of missing data will be left out of the calculations. The names of discarded loci will be written to the *_deletedloci* file. The default setting is 1. See section 5.4.

FULL_R (boolean) A value of 1 will create an additional file named with the string specified at *R_OUT* with "_fulldata" concatenated at the end (and before the extention if specified). This file will contain the data for each locus as well as the summary statistics. Default setting is 0. See section 5.3.

FULL_P (boolean) A value of 1 will create an additional file named with the string specified at *P_OUT* with "_fulldata" concatenated at the end. This file will contain the data for each locus as well as the summary statistics. Default setting is 0. See section 5.3.

FULL_C (boolean) A value of 1 will create an additional file named with the string specified at *C_OUT* with "_fulldata" concatenated at the end. This file will contain the data for each locus as well as the summary statistics. Default setting is 0. See section 5.3.

PRINT_PROGRESS (boolean) A value of 1 will calculate and display a progress report during potentially long calculations. A value of 0 will suppress these calculations and output. Default setting is 1.

4.4 Command line arguments

The command line flags give the user the option to enter information from the command line. All command line arguments will overwrite values specified in the *paramfile*. If any command line arguments are specified, the *paramfile* must also be given. All command line arguments are given after the name of the *paramfile* and then may appear in any order. The command line flag is followed by a space and then the parameter value. For example on a linux platform, `./adze-1.0 paramfile.txt -r richness_out` will change the name of the allelic richness output file to "richness_out" and will use the other parameters defined in "paramfile.txt".

-g (MAX_G) Change the maximum standardized sample size.

-d (DATA_LINES) Change the number of data lines expected (useful when using *-f* for a new *datafile*).

-l (LOCI) Change the number of loci expected (useful when using *-f* for a new *datafile*).

-nr (NON_DATA_ROWS) Change the number of non-data rows expected (useful when using *-f* for a new *datafile*).

-nc (NON_DATA_COLS) Change the number of non-data columns expected (useful when using *-f* for a new *datafile*).

-s (GROUP_BY_COL) Change the non-data column by which to group the data. Must be less than or equal to NON_DATA_COLS and greater than or equal to 1.

-f (DATA_FILE) Change the source *datafile*.

-r (R_OUT) Change the name of the file to which allelic richness output will be written.

-p (P_OUT) Change the name of the file to which private allelic richness output will be written.

-c (COMB) Toggle combination calculations on/off.

-k (K_RANGE) Change the sizes of combinations to calculate.

-o (C_OUT) Change the name of the file to which private allelic richness of combinations output will be written.

-m (MISSING) Change the representation of missing data in the *datafile*.

-t (TOLERANCE) Change the tolerance.

-tnocalc Takes a 1 or 0. When switched to 1, *ADZE* prints an onscreen report of how many loci will be filtered by specified *TOLERANCE*. Suppresses all calculations and file creation except *_deletedloci* and *_summary* files. Default value is 0.

-fr (FULL_R) Takes a 1 or 0. When switched to 1, prints full allelic richness data.

-fp (FULL_P) Takes a 1 or 0. When switched to 1, prints full private allelic richness data.

-fc (FULL_C) Takes a 1 or 0. When switched to 1, prints full private allelic richness data for all combinations being calculated.

-pp (PRINT_PROGRESS) Takes a 1 or 0. When switched to 0, suppresses display and calculation of progress during all potentially long calculations. The default value is 1.

5 Output files

ADZE can create many output files. These are described below.

5.1 *R_OUT* and *P_OUT*

This file contains a listing of summary statistics for either the allelic richness (*R_OUT*) or private allelic richness (*P_OUT*) of each grouping. The first column contains the grouping name (determined from the *datafile* and the parameter *GROUP_BY_COL*). The next column gives the g value at which the results were calculated, and the last three columns give the mean, variance, and standard error of the mean across all loci not filtered due to missing data. Groupings are separated by a newline. This file is in the following format:

```
name1    2    #loci  mean  var  stderr
name1    3    #loci  mean  var  stderr
  ⋮      ⋮      ⋮     ⋮     ⋮     ⋮
name1  max_g  #loci  mean  var  stderr

name2    2    #loci  mean  var  stderr
name2    3    #loci  mean  var  stderr
  ⋮      ⋮      ⋮     ⋮     ⋮     ⋮
name2  max_g  #loci  mean  var  stderr

  ⋮      ⋮      ⋮     ⋮     ⋮     ⋮

nameJ    2    #loci  mean  var  stderr
nameJ    3    #loci  mean  var  stderr
  ⋮      ⋮      ⋮     ⋮     ⋮     ⋮
nameJ  max_g  #loci  mean  var  stderr
```

5.2 *C_OUT*

This file contains a listing of summary statistics for the private allelic richness of each combination of k groupings. The first k columns contain the grouping names (determined from the *datafile* and the parameter *GROUP_BY_COL*). The next column gives the g value at which the results were calculated, and the last three columns give the mean, variance, and standard error of the mean across all loci not filtered due to missing data. Each k-grouping is separated by a newline. This file is in the following format:

name11	*name12*	$\cdots$	*name1k*	*2*	*#loci*	*mean*	*var*	*stderr*
name11	*name12*	$\cdots$	*name1k*	*3*	*#loci*	*mean*	*var*	*stderr*
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
name11	*name12*	$\cdots$	*name1k*	*max_g*	*#loci*	*mean*	*var*	*stderr*
name21	*name22*	$\cdots$	*name2k*	*2*	*#loci*	*mean*	*var*	*stderr*
name21	*name22*	$\cdots$	*name2k*	*3*	*#loci*	*mean*	*var*	*stderr*
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
name21	*name22*	$\cdots$	*name2k*	*max_g*	*#loci*	*mean*	*var*	*stderr*
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$name\binom{J}{k}1$	$name\binom{J}{k}2$	$\cdots$	$name\binom{J}{k}k$	*2*	*#loci*	*mean*	*var*	*stderr*
$name\binom{J}{k}1$	$name\binom{J}{k}2$	$\cdots$	$name\binom{J}{k}k$	*3*	*#loci*	*mean*	*var*	*stderr*
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$name\binom{J}{k}1$	$name\binom{J}{k}2$	$\cdots$	$name\binom{J}{k}k$	*max_g*	*#loci*	*mean*	*var*	*stderr*

5.3 *FULL_**

This file contains a listing of summary statistics plus data at all non-filtered loci for either the allelic richness (*FULL_R*), the private allelic richness (*FULL_P*), or the private allelic richness of each combination of k-groupings (*FULL_C*). The first line contains the name of every locus used in the calculations. The first one or more columns contain the grouping name (for *FULL_R* and *FULL_P*) or names (for *FULL_C*). The next column gives the g value at which the results were calculated. The next columns give the calculated richness results at all non-filtered loci, and the last three columns give the mean, variance, and standard error of the mean across all those loci. Each grouping or k-grouping is separated by a newline. This file is in the following format:

POP_GROUPING(s)	*G*	*NUM_LOCI*	*locus1*	*locus2*	⋯	*locusL*	*MEAN*	*VAR*	*STD_ERR*
name(s)1	*2*	*#loci*	*data1*	*data2*	⋯	*dataL*	*mean*	*var*	*stderr*
name(s)1	*3*	*#loci*	*data1*	*data2*	⋯	*dataL*	*mean*	*var*	*stderr*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
name(s)1	*max_g*	*#loci*	*data1*	*data2*	⋯	*dataL*	*mean*	*var*	*stderr*
name(s)2	*2*	*#loci*	*data1*	*data2*	⋯	*dataL*	*mean*	*var*	*stderr*
name(s)2	*3*	*#loci*	*data1*	*data2*	⋯	*dataL*	*mean*	*var*	*stderr*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
name(s)2	*max_g*	*#loci*	*data1*	*data2*	⋯	*dataL*	*mean*	*var*	*stderr*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
name(s)N	*2*	*#loci*	*data1*	*data2*	⋯	*dataL*	*mean*	*var*	*stderr*
name(s)N	*3*	*#loci*	*data1*	*data2*	⋯	*dataL*	*mean*	*var*	*stderr*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
name(s)N	*max_g*	*#loci*	*data1*	*data2*	⋯	*dataL*	*mean*	*var*	*stderr*

5.4 *_deletedloci*

This file is created as a consequence of using the *TOLERANCE* parameter to filter loci with missing data. This file has the same name as the *P_OUT* file, with "_deletedloci" concatenated on the end. It is a report of the number of loci deleted as a result of the specified *TOLERANCE*, followed by a list of deleted locus names.

5.5 *_summary*

This file is created whenever *ADZE* is run. The file has the same name as the *R_OUT* file, with "_summary" concatenated on the end. This is a report of the parameters used and a list of times (from beginning of execution) at which each major calculation finished.

6 Examples

Here we present two examples for using *ADZE* on a Linux platform.

6.1 A small data set example

In this example we use a small contrived data set (Fig. 1) to work through the basic operation of *ADZE*. We name this data set *small_data.stru* and place it in the same directory as *ADZE*. We then create a *paramfile* as in Figure 2 and name it *small_paramfile.txt*. To run *ADZE* with this *paramfile*, type `./adze-1.0 small_paramfile.txt` at the command prompt in the directory containing *ADZE* and the two other files. By examining the private richness and combination output files, we can see that even though we set *MAX_G* to 8, *ADZE* was only able to achieve a *MAX_G* of 3. By looking at the richness output file, we can see that the *EUROPE* grouping goes up to the lowest g, and therefore this grouping must have either a small sample size or missing data for at least one locus. Re-examining the *datafile* in Figure 1, we find that the *EUROPE* grouping has a small sample size relative to the others, so the highest *MAX_G* we could hope to achieve for private allele calculations is 6. Additionally, we see that on data lines $9-11$ *EUROPE* is missing data at the fourth locus. A quick survey of the rest of the data reveals that the fourth locus is missing 50% data in the *EUROPE* grouping and 25% in the *AMERICA* grouping. Additionally, the second locus is missing 25% data in the *EAST_ASIA* grouping. We can re-run *ADZE* with a tolerance set, so that it will filter out these loci and allow calculations up to a larger g. Type `./adze-1.0 small_paramfile.txt -t .24` at the command prompt to filter out loci with strictly more than 24% missing data in any given grouping. This will exclude the second and fourth loci from all calculations. Note that we did not change the output file names, so they will be overwritten. On examining the new output files, we see that by filtering loci with missing data, we have achieved the highest possible *MAX_G*, limited only by sample size.

6.2 A larger data set example

The next example uses a larger data set with 1056 individuals from 52 populations across 377 loci. This data file is from the Rosenberg *et al.* (2002) study and can be downloaded at `http://rosenberglab.bioinformatics.med.umich.edu/diversity.html`

Because calculating private alleles for all combinations at the population level would be incredibly time consuming, we will, for the purposes of this example, run *ADZE* at the continent level. Similarly, we will not output the full results at every locus since these result files can grow very large. We use the *paramfile* in Figure 3 named *big_paramfile.txt*. To run *ADZE* with this data set and *paramfile* type `./adze-1.0 big_paramfile.txt`, and we can visualize the results by graphing mean richness vs. g for each grouping. Figure 4 gives the mean allelic richness vs. g, Figure 5 the mean private allelic richness vs. g, and Figure 6 the mean private allelic richness for combinations of six groupings (or "missing" alleles).

```
# This is a parameter file for ADZE.
# The entire line after a '#' will be ignored.

###----Main Parameters----###
MAX_G 8
DATA_LINES 22
LOCI 5
NON_DATA_ROWS 1
NON_DATA_COLS 5
GROUP_BY_COL 5
DATA_FILE small_data.stru
R_OUT small_r
P_OUT small_p

###----Combination Parameters----###
COMB 1
K_RANGE 2
C_OUT small_c

###-----------Advanced Options-----------###
MISSING -9
TOLERANCE 1
FULL_R 1
FULL_P 1
FULL_C 1
###----End of parameter file----###
```

Figure 2: A *paramfile* for our small data set

```
# This is a parameter file for ADZE.
# The entire line after a '#' will be ignored.

###----Main Parameters----###
MAX_G 70
DATA_LINES 2112
LOCI 377
NON_DATA_ROWS 1
NON_DATA_COLS 5
GROUP_BY_COL 5
DATA_FILE diversitydata.stru
R_OUT big_r
P_OUT big_p

###----Combination Parameters----###
COMB 1
K_RANGE 2-6
C_OUT big_c

###-----------Advanced Options-----------###
MISSING -9
TOLERANCE 1
FULL_R 0
FULL_P 0
FULL_C 0
###----End of parameter file----###
```

Figure 3: A *paramfile* for our larger data set

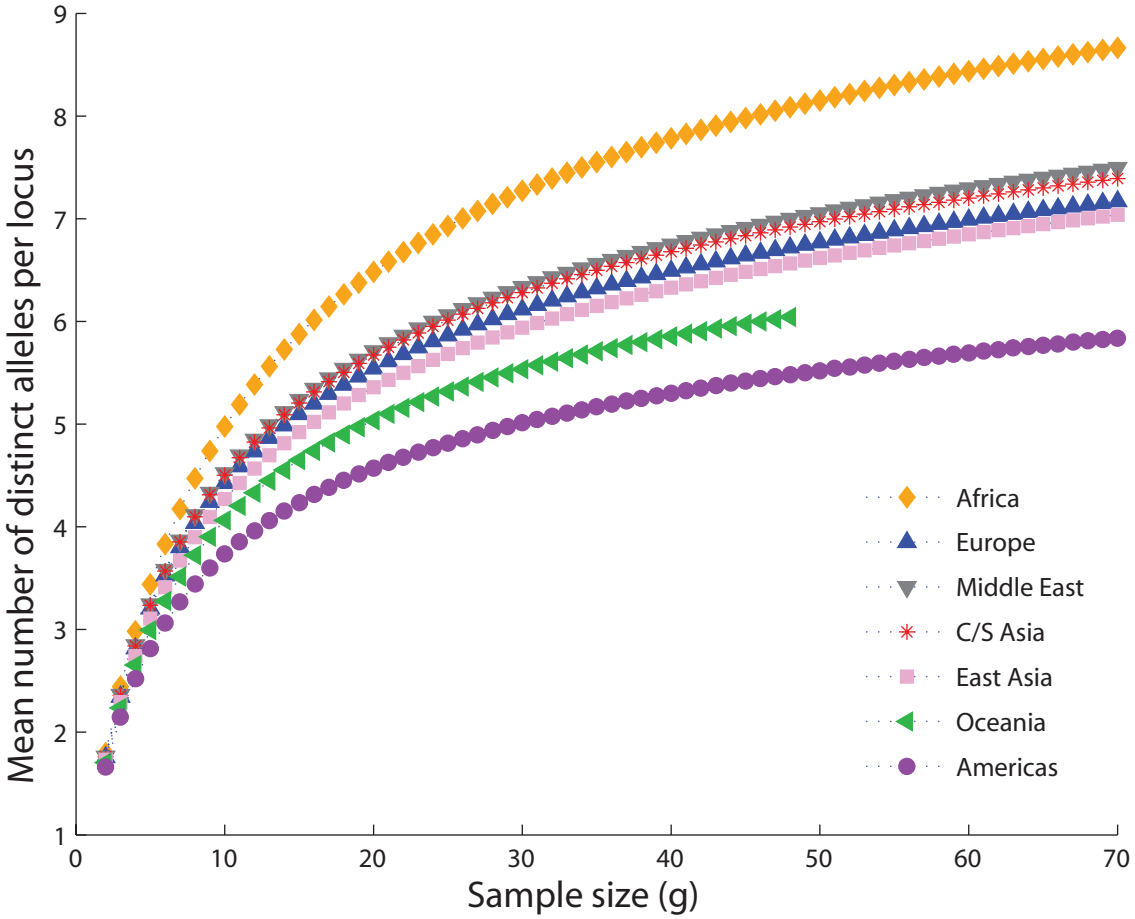

Figure 4: The mean number of distinct alleles per locus as a function of standardized sample size for seven major geographic regions.

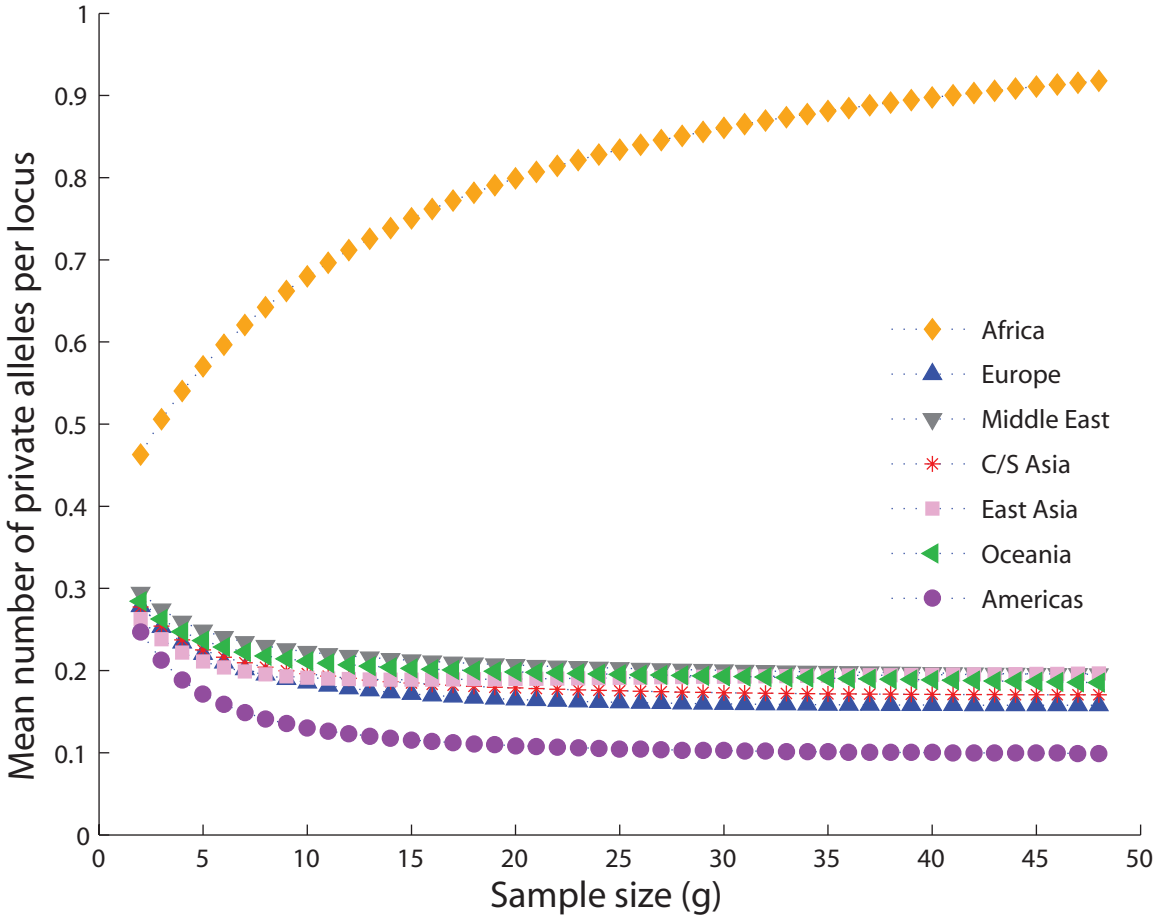

Figure 5: The mean number of private alleles per locus as a function of standardized sample size for seven major geographic regions.

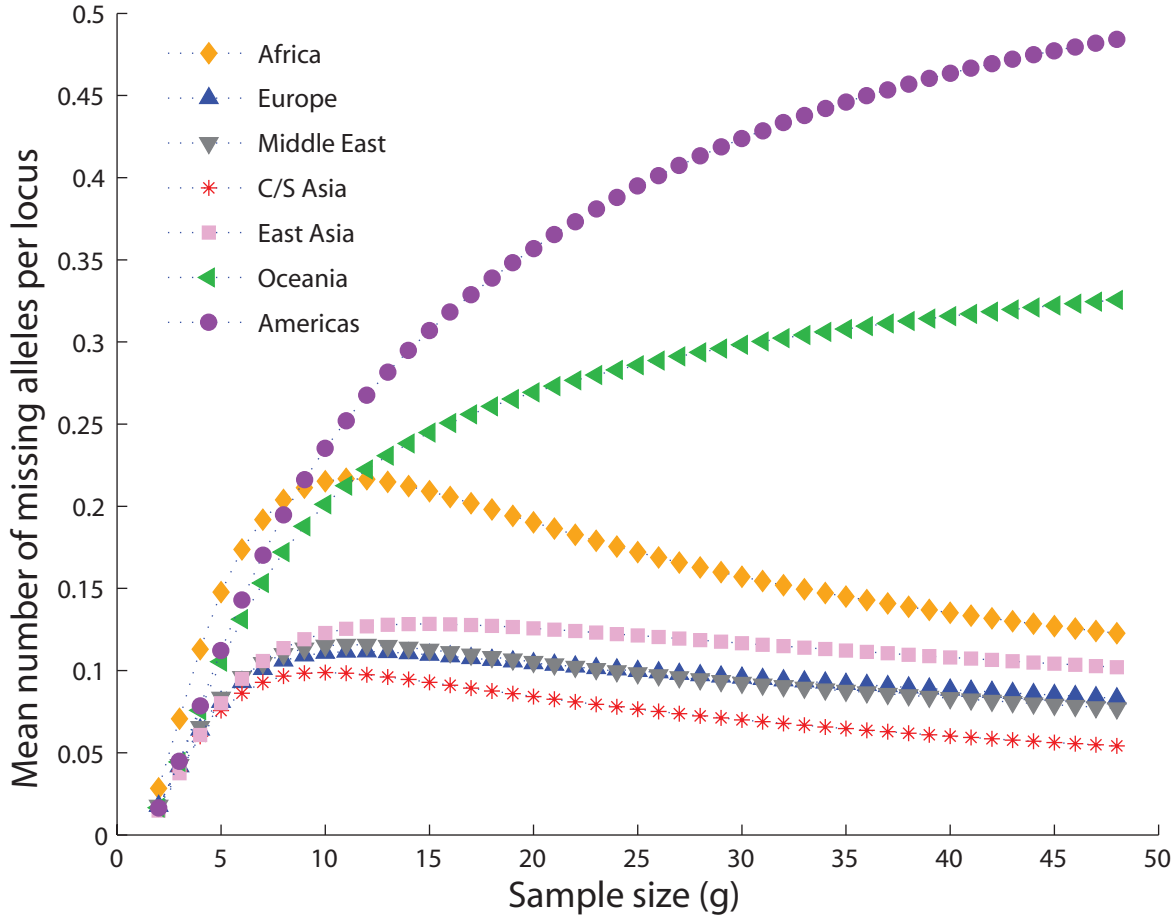

Figure 6: The mean number of missing alleles per locus for seven major geographic regions (private to combinations of six of seven major geographic regions) as a function of standardized sample size.

References

Hurlbert, S. H. 1971. Nonconcept of species diversity: A critique and alternative parameters, *Ecology* **4**, 577–586.

Kalinowski, S. T. 2004. Counting alleles with rarefaction: private alleles and hierarchical sampling designs, *Conservation Genetics* **5**, 539–543.

Kalinowski, S. T. 2005. HP-RARE 1.0: a computer program for performaing rarefaction on measures of allelic richness, *Molecular Ecology Notes* **5**, 187–189.

Petit, R., Mousadik, A. E., and Pons, O. 1998. Identifying populations for conservation on the basis of genetic markers, *Conservation Biology* **12**, 844–855.

Rosenberg, N. A., Pritchard, J. K., Weber, J. L., Cann, H. M., Kidd, K. K., Zhivotovsky, L. A., and Feldman, M. W. 2002. Genetic structure of human populations, *Science* **298**, 2381–2385.