



A. Lancaster
M. P. Nelson
R. M. Single
D. Meyer
G. Thomson

13th IHWG Shared Resources Joint Report

B3.1 Software framework for the Biostatistics Core of the International Histocompatibility Working Group

Authors' addresses

Alex Lancaster¹, Mark P. Nelson¹, Richard M. Single²,
Diogo Meyer¹, Glenys Thomson¹

¹University of California, Berkeley,
Department of Integrative Biology,
3060 Valley Life Sciences
Berkeley CA 94720, USA
²University of Vermont, Department of
Biometry, Burlington VT, USA

Correspondence to:

Alex Lancaster
University of California, Berkeley
Department of Integrative Biology
3060 Valley Life Sciences
Berkeley CA 94720, USA
e-mail: alexl@socrates.berkeley.edu

Acknowledgements

This work has benefited from the support of NIH grant AI49213 (13th IHWG Shared Resources). Thanks to Steve Mack.

Introduction

The PyPop (Python for Population Genetics) analysis package is a suite of population genetic analyses and forms the base of the biostatistics core. It is in an object-oriented framework implemented in the programming language Python. The object-oriented approach allows us to implement individual analysis modules which can be inserted or removed without affecting other modules. Python is a flexible scripting language which allows rapid prototyping of code and has powerful features for interfacing with other languages, such as C (in which we have already implemented many routines and which is particularly suited to computationally intensive tasks). The output of the analyses are stored in the XML format (XML is the eXtensible Markup Language devised by the World Wide Web Consortium, and is a platform-independent, vendor-neutral, non-proprietary, open standard for storing data (2)). These output files can then be transformed using standard tools into many other data formats suitable for machine input (such as PHYLIP (6) or input for spreadsheet programs such as Excel or statistical packages, such as R (8)), plain text, or HTML for "human-readable" format. Storing the output in XML allows the final viewable output format to be redesigned at will, without requiring the (often time-consuming) re-running of the analyses themselves. Data from the anthropology, transplantation, disease and cytokine components have been analyzed using PyPop.

IHWG: International Histocompatibility Working Group

The primary motivation for developing this project was our role as the "Biostatistics Core" for the International Histocompatibility Working Group (IHWG (<http://www.ihwg.org/>)). As part of the Anthropology and Human Diversity component of the IHWG, HLA class I (A, B and C) and class

HLA 2003

Vol. ■: 000–000

Printed in Denmark. All rights reserved

Copyright © Blackwell Munksgaard 2004

HLA 2002



II (DRA1, DRB1, DQA1, DQB1, DPA1 and DPB1) data from populations from upwards of 50 laboratories was made available via a database housed at the Fred Hutchinson Cancer Research Institute (the “Database Core”).

Population genetic statistics

The particular population analyses we wished to address included: (1) conformity to Hardy-Weinberg Proportions (HWP), (2) tests for balancing or directional selection; (3) estimates of haplotype frequencies (and their distributions) and measures and tests of significance for linkage disequilibrium (LD) (among populations); and (4) other tests such as worldwide patterns of genetic differentiation. A description of the methodology for specific tests can be found in the Workshop Statistical Methods paper of the current volume. Section 2.3 briefly outlines the PyPop modules and specific implementation details of population genetic methods.

Requirements

The data sets for analysis were highly heterogeneous. Datasets varied considerably in the number of loci typed (from two to nine), number of individuals sampled (from 12 to 1012), and number of alleles at each locus (from 2 to 55). Given that we had such highly diverse sets of data, it was necessary to generate analyses that could be integrated in a systematic way. This led to the following set of requirements for the software framework.

- **Modular.** Analyses (e.g. tests of HWP) should be implemented as separate modules to allow for the output from each test to be self-contained, and tests added in the future to be integrated into the package seamlessly without affecting existing tests.
- **Configurable.** Analyses should be able to be switched on or off as required by the user in a simple configuration file and should be able to be run stand-alone or as part of a battery of tests.
- **Machine-readable output.** Output from the analyses should be available as input to other programs, so successive stages in the output can be made available as input to later stages.
- **Standardized output.** Output should be generated in the open standard format XML, which is both machine readable and easily translated into human-readable format (such as text).
- **Cross-platform.** Software should be cross platform, and not be tied to proprietary features. It should be available to run on high-performance UNIX platforms such as GNU/

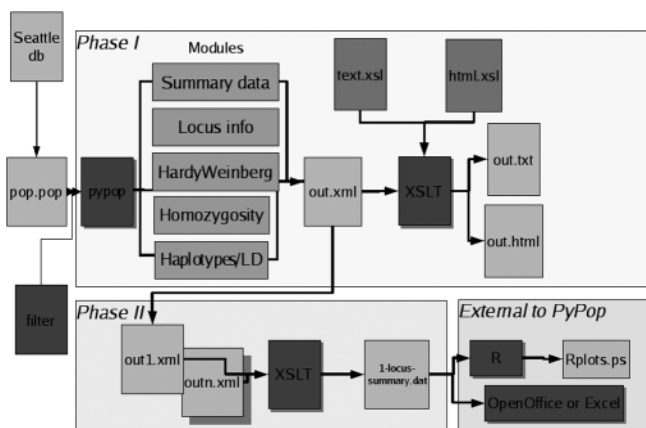


Figure 1. Overview of PyPop dataflow.

Linux or Solaris as well as Windows and Macintosh platforms.

- **Integrating.** The platform should allow simple integration of modules written in other languages (e.g., C, such as *emhaplofreq*, a program for haplotype frequency estimation for the highly polymorphic HLA loci) and/or third party software, packages that deal some aspects of population genetic statistics such as *Arlequin* (10), *Genepop* (9), and *PHYLIP* (6).

Method

Overall design

To integrate the data analyses for multiple populations, the analysis pipeline for PyPop has been designed in two major phases. The first phase encompasses the population genetic analyses, and the second integrates the results of these analyses across multiple populations to generate output that can be read by a statistical package or spreadsheet program. The overall data and workflow is shown in Fig. 1.

Implementation

We implemented PyPop in the object-oriented scripting language Python (14). It is an interpreted language allowing for rapid prototyping of modules and has a convenient standard library of functions. Through use of the the Simple Wrapper Interface Generator (SWIG (1)), it is also straightforward to “wrap” existing code in C, C++ or Java and call it from Python. We leveraged other open-source projects: Numeric Python (4) project, which provides efficient data structures for holding large arrays of data. As our XSLT parser, we chose the fast C-based *libxslt* (<http://xmlsoft.org/>) from the GNOME (<http://www.gnome.org/>) project.

PyPop modules

PyPop groups analyses into separate modules, represented in the data flow by the grey boxes in the group HardyWeinberg, Homozygosity and Haplotypes/LD in Fig. 1. These modules are outlined below.

Population summary data. This module extracts the following information from the input data files: labcode, typing method, ethnic group, continent of origin, collection site, latitude, longitude, total sample size, and number of loci typed.

Locus-specific information. Population genetics statistics at the single-locus level are tabulated in this module including locus name, sample size, number of distinct alleles observed, and the counts and frequencies of all alleles observed in the sample (ordered by frequency and then by allele name).

Hardy-Weinberg proportions. This module allows several implementations of tests for HWP. In addition to the traditional chi-square test of HWP, we performed the overall exact test using Arlequin's implementation of Guo & Thompson's (7) method. The PyPop framework has a facility to generate Arlequin format project (.arp) and settings (.ars) files, execute the batch version of the program, and re-import the results back into the framework for further downstream processing.

Ewens-Watterson test of neutrality. For the Ewens-Watterson (5, 13) homozygosity test of neutrality we created an extension module in PyPop, that uses a C implementation of a Monte-Carlo approximation to the exact test kindly provided by Montgomery Slatkin (11, 12). We modified these programs to additionally report the normalized deviate of homozygosity (Fnd) and the variance.

Haplotype frequency estimation and linkage disequilibrium. Haplotype frequencies (estimated using the expectation-maximization (EM) algorithm) and LD measures were calculated using emhaplofreq, a C extension module that we wrote as part of the framework.

With the exception of the Arlequin implementation of the Guo & Thompson HWP exact test and Slatkin's Monte-Carlo implementation of the Ewens-Watterson test, new programs were developed to handle the highly polymorphic HLA loci. However, they are not limited to use on HLA data, but are appropriate for analysis of any loci with less polymorphism than HLA. We designed PyPop to handle not only the high level of polymorphism, but also large sample sizes (e.g., sample sizes of over 1,000 for some anthropology data and 2,000

for the overall transplant data) which can be problematic for methods using resampling and/or iterative procedures, such as the HWP exact test, haplotype estimation, and LD significance testing.

Data flow

The input to PyPop is a text file with the .pop extension, containing the multi-locus genotype data for each population. The file format is described in Appendix A. Note that either allele names or sequence data can be used as input, PyPop does not distinguish between allele calls or sequence data and can transparently handle both.

Before running data through PyPop, ambiguities in typing should be resolved (in our case this has been done by Steve Mack for the anthropology data, by Effie Petersdorf and Mari Malkki for the transplantation component, and by the respective disease chairs in the disease component).

Filtering

Before phase I analysis begins, the .pop file can be passed through a filter module for data validation (this module can be switched off if it is not needed). The filter module is a set of rules that ensures that allele names are valid. (An IHWG-specific filter was implemented as part of PyPop using the IMGT/HLA database (<http://www.ebi.ac.uk/imgt/hla/>), a database of HLA alleles maintained by the Anthony Nolan Trust in the UK), see section C2 in this volume for further details.

Phase I

Phase I analysis consists of a run of a single population data file run through the set of PyPop modules (using the **pypop** script). (For the IHWG data set, once each single population analysis was complete, an output file of results was forwarded to the contributing laboratory for review and appropriate revision of data. If the review of data required revisions, Phase I was repeated). The results of the Phase I analyses are stored in an XML file (one per population) with the format as shown in Fig. 2.

XML was chosen as the output format because: 1) it can be read as input by other programs and; 2) it is readily transformable into human-readable form (e.g., a text file) or web-form (e.g., HTML) via XSLT (eXtensible Stylesheet Language for Transformations) (3).

Phase II

Phase II, integrates the results of the analyses of the individual data files from Phase I (using the **popmeta** script). At this stage the benefits of using XML as the storage and exchange

```

<dataanalysis date="2002-07-09-05-01-18">
  <filename>ukimid_nireland.pop</filename>
  <pypop-version>DEVEL_VERSION</pypop-version>
  <populationdata>
    <labcode>UKIMID</labcode>
    <method>SSOP</method>
    <ethnic>Irish</ethnic>
    <contin>Europe</contin>
    <collect>Northern Ireland</collect>
    <latit>54 degrees 40 minutes north</latit>
    <longit>6 degrees 45 minutes west</longit>
    <complex>3</complex>
    <popname>NIreland</popname>
    <totals>
      <indivcount>1000</indivcount>
      <allelecount>2000</allelecount>
      <locuscount>9</locuscount>
      <lociWithDataCount>4</lociWithDataCount>
    </totals>
  </populationdata>
  <locus name="A">
    <allelecounts>
      <untypedindividuals>0</untypedindividuals>
      <indivcount>1000</indivcount>
      <allelecount>2000</allelecount>
      <distinctalleles>26</distinctalleles>
      <allele name="0101">
        <frequency>0.20200 </frequency><count>404</count>
      </allele>
    </allelecounts>
  </locus>
</dataanalysis>

```

Figure 2. Extract from sample XML output file.

format are realized. There are two major benefits: one relevant for displaying results of each individual run, the second, far more powerful benefit, for aggregating the data for cross-population meta-analyses and transforming it as input for third party packages.

Generating human-readable output. First, since many of the analyses (notably estimating the significance of all pairwise LD) can take a considerable amount of time, especially for large population samples with highly polymorphic loci, the analysis generation stage is decoupled from the analysis presentation stage. This enables tweaking of the human-readable text output of the individual files of the presentation without completely re-running the analyses. A sample text output is shown in Fig. 3. The same XML content can also be used to generate a web version (HTML) of the same data set. These are the files that were provided to individual labs.

Combining results across populations. Second, the results of Phase I can be aggregated across populations. As indicated in

the “Phase II” box in Fig. 1, the individual XML files (indicated by out1.xml to outn.xml in the diagram) are transformed using an XSLT stylesheet into tables of data that can be read by any spreadsheet or statistical package such as R. The tables generated are all files of tab-separated values (TSV) that include:

- 1-locus-summary.dat: consists of a line for population and locus, with fields for number of gametes number of distinct alleles, HWP p-value for the Chi-square test and all other single locus statistics.
- 1-locus-allele.dat: consists of a line for each combination of population, locus and allele. The line of data contains the allele frequency (allele.freq) and count (allele.count) a short example of this file is shown in Fig. 5 below.
- 2-,3-, 4-locus-summary.dat: consists of a line for each combination of population, and locus group (with separate files for pairs, three or four locus). The line of data contains LD measure using both D' and Wn measures and other overall tests of LD. Separate files were used for two, three or four locus haplotypes.

Performed on the 'ukimid_nireland.pop' file at: 2002-07-09-05-01-18

Figure 3. Extract from sample plain text output generated from XML data.

Population Summary
=====

Lab code: UKIMID
Typing method: SSOP
Ethnicity: Irish
Continent: Europe
Collection site: Northern Ireland
Latitude: 54 degrees 40 minutes north
Longitude: 6 degrees 45 minutes west
Population Name: NIreland
[...]

1.1. Allele Counts [A]

Untyped individuals: 0
Sample Size (n): 1000
Allele Count (2n): 2000
Distinct alleles (k): 26

Counts ordered by frequency			Counts ordered by name		
Name	Frequency	(Count)	Name	Frequency	(Count)
0201	0.27400	548	0101	0.20200	404
0101	0.20200	404	0201	0.27400	548

- 2-,3-, 4-locus-haplo.dat: this is analogue of the 1-locus-allele.dat, except with information for each population's haplotype, such as the estimated haplotype count and frequency.
- To illustrate this process, short excerpts from the XML files (which include allele frequencies for the same locus, "A") of two individual population runs Bugawan-Filipino-out.xml and CZEIVS-Czech-out.xml are shown below:

The phase II part of PyPop merges the allele frequencies (noted in **bold**) of the separate XML files into a single text file, 1-locus-allele.dat which contains all the allele frequencies

listed by population and by locus. Here is an extract of the file showing just two populations containing the same allele (note that some columns have been omitted for clarity):

1. Allele frequency of 0101 allele in Filipino population
2. Allele frequency of 0101 allele in Czech population

The PyPop application concludes with the generation of the *.dat files. The next section deals with various possible uses of the data that PyPop generates after both Phase I and Phase II are completed, but it is important to note that they are not part of the framework.

Excerpt from Bugawan-Filipino-out.xml

```
...
<locus name="A">
<allelecounts>
<untypedindividuals>0</untypedindividuals>
<indivcount>94</indivcount>
<allelecount>188</allelecount>
<distinctalleles>16</distinctalleles>
>
<allele name="0101">
<frequency>0.00532</frequency>
<count>1</count>
</allele>
```

Excerpt from CZEIVS-Czech-out.xml

```
...
<locus name="A">
<allelecounts>
<untypedindividuals>1</untypedindividuals>
<indivcount>105</indivcount>
<allelecount>210</allelecount>
<distinctalleles>22</distinctalleles>
>
<allele name="0101">
<frequency>0.12857 </frequency>
<count>27</count>
</allele>
```

Figure 4. Excerpts from Bugawan-Filipino and CZEIVS-Czech output.

pop	ethnic	region	locus	allele	allele.freq	allele.count
Filipino	Filipino	06.Oceania	A	0101	0.00532	1
...						
Czech	Czech	03.Europe	A	0101	0.12857	27

Figure 5. Extract from 1-locus-allele.dat.

After PyPop: using the output

For the individual user, analyses may end with the generation of a text file of results for a single population. When multiple populations are involved the user may want to use what we refer to as the phase II features. The final output of this part of the PyPop package consists of both the .dat output tables and the individual XML files for each population included in the analysis. The former are text files consisting of tab-separated values (TSV) available for direct import to many statistical packages. The latter are XML files which contain more information than the *.dat (such as numbers of steps required for convergence for the EM algorithm) and can like be used, as input for alternative analyses. We discuss each of these below.

Using the tables

Packages to read the tables in the *.dat files can be divided into two kinds: graphical spreadsheet-style, which can generate a wide range of charts, graphs and summaries, and statistical packages which offer a more flexible and powerful alternative, but often require greater computer expertise. The output that PyPop generates can be handled easily by both. A non-exhaustive list of options includes:

- spreadsheet packages. OpenOffice.org Calc, part of the open-source multi-platform Office package OpenOffice.org (<http://www.openoffice.org/>), the Gnumeric spreadsheet package available widely on UNIX and GNU/Linux systems, and proprietary alternatives such as Microsoft Excel on Windows and Macintosh platforms can all easily handle TSV files.
- statistical packages. R, Splus, SAS and other programs can all handle TSV text files. For, example in the case of the graphical output for the IHWG the described in the next section, we wrote custom R scripts (not currently part of PyPop) highly specific to HLA data to generate various summaries and plots from the raw table output.

Advanced usage

The XML output (both the individual population-level data files, and the aggregated multi-population data) can also be

transformed (via XSLT) into formats suitable for input to other programs. Currently we have prototype modules for generating input for PHYLIP and Arlequin. We ultimately wish to develop an extensible “plug-in” XSLT architecture so that the output can be transformed into formats suitable for other programs and packages in such a way that doesn’t require the user to do their own programming.

Finally, the raw XML files offer the opportunity for those who are so inclined, to write custom XSLT stylesheets which could parse the XML and transform the data in many ways. Alternatively it is possible to load the XML into a database and query the data using SQL.

Results and sample output

Analyzing IHWG workshop data

The PyPop framework was used to analyze the full set of IHWG data from both the 12th and 13th workshops. With over 140 separate data-sets (some of which individually required a long time to complete the basic per-population analyses), it was straightforward to set up a batch program (calling the **pypop** script on each .pop file) to generate the individual output analyses. For each file output, an individual XML file was generated. From these individual XML files, input data files for the statistical package R (the open-source implementation of the programming language S) were generated using XSLT stylesheets as described above in Phase II of PyPop (invoked by the **popmeta** script). The R code was set up to generate overall graphical and numeric summaries for many population genetic parameters of interest. In particular, the analysis allows “slicing” of the data along a number of axes. As an example, we can view proportion of populations with significant deviation from HWP, for all geographic regions at a given locus, or view all loci for a given region (data not shown). As another example, we can view the number of populations that have data for a given region (Fig. 6), or number of populations for which data was provided for a given locus (Fig. 7).

For the IHWG analyses, the complete pipeline, from analyzing the individual data files, to the generation of the meta-data *.dat files can be completely automated from the command-line. Furthermore, the flexibility of the XML format,

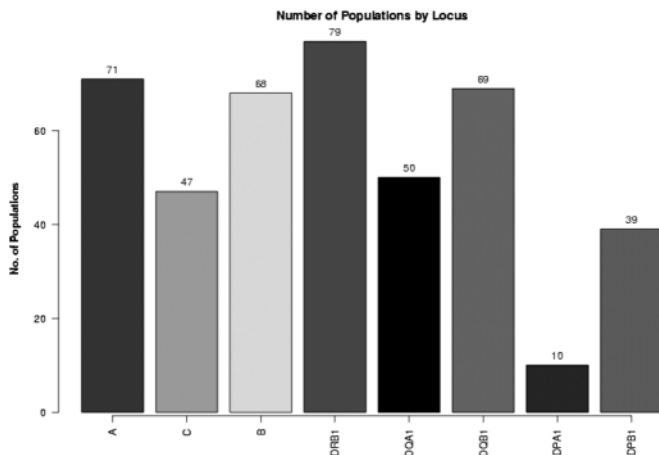


Figure 6. Sample metadata output: viewing number of populations by locus.

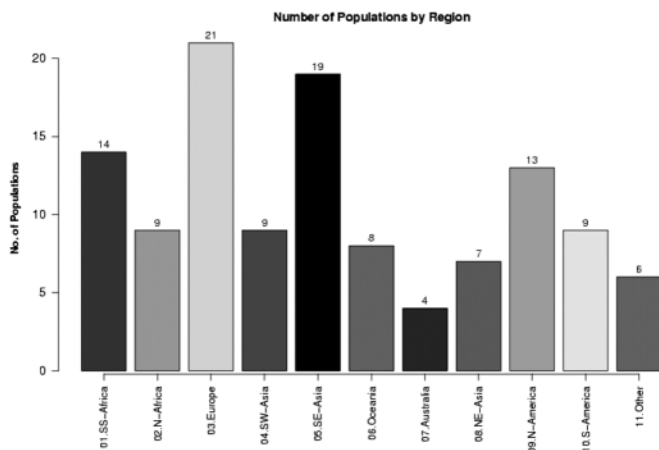


Figure 7. Sample metadata output, a second way to view data: number of populations within each region.

allows us to extract and output data easily for future data analyses without requiring regeneration of the basic population genetics statistics.

Role and significance

It is important to note that PyPop is not attempting to supplant or replace other, more established, population genetics analysis packages. Many of the basic population genetic statistics computed by PyPop are not new in and of themselves, nor is the ability to perform analyses on multiple populations (e.g., Arlequin can currently do this). Indeed, PyPop can use existing packages to calculate them. However the approach of integrating information in highly heterogeneous datasets on a large scale is new, and not currently available in the evolutionary genetics community. In addition to adding some tests

that are unavailable in existing projects, PyPop is unique in that it is intended to be a high-throughput system that enables population genetics to join the realm of genomics.

Future directions

In the future we plan to continue development on the modules that can inter-operate with PHYLIP and Arlequin. We also plan to develop a module that can associate each allele name to an underlying amino-acid sequence (for residues that are known). Analyses may then be run by considering each amino acid site as a genetic locus, rather than the entire allele, as is done currently. This will result in a useful module for those wishing to analyze allele data at the sequence level. Longer-term possibilities include developing a graphical front-end, possibly web-based, integration with data mining tools such as clustering analysis, and integration with the NCBI's dbMHC (<http://www.ncbi.nlm.nih.gov/mhc/>) database. The complete source code to PyPop has been released under the GNU General Public License (GPL (<http://www.gnu.org/licenses/gpl.html>)). Details are available at the Thomson lab website: <http://allele5.bi-ol.berkeley.edu/>, and will also be made available at the IHWG website: <http://www.ihwg.org/>.

Appendix. Data format

The format for the input files for PyPop is plain ASCII, with fields delimited by tabstops, not spaces. Fields are separated by tabs so that spaces are permitted within a field. The file suffix should be .txt or .pop. No field may be left blank. Four asterisks should be used to designate untyped locus data or unknown descriptive data. There should be 3 lines of header information on a population file.

The three lines are as follows:

```
labcode  method  ethnic  contin  collect  latit   longit   complex ①
USAERL  SSOP      Irish   Europe  Dublin   30 deg  350 deg  1 ②
populatid a_1 a_2 c_1 c_2 b_1 b_2 tap1_1 tap1_2 ③
```

① **Line 1.** Column headers

② **Line 2.** Data under headings in line 1 are filled in on line 2 as follows:

The method should be one of the following:

```
SSOP
RLS
SBT
SSP
RSCA
```

MS (this is for PCR/gel sizing for microsatellites)
 RFLP
 SSCP
 Other

The ethnic field may contain practically anything, but the continent field should contain one of the following geographic regions:

North-Africa
 Sub-Saharan-Africa
 North-East-Asia
 South-East-Asia
 South-West-Asia
 Australia
 Europe
 North-America
 Oceania
 South-America
 Other

The collect field is for where the data was collected, latit and

longit are for the latitude and longitude (approximately) of origin, and complex should be entered as a zero.

③ **Line 3.** There is one more line which contains the headers for the columns of genotype data.

populat should contain a name for the population and id should contain a sample id.

There should be two header labels for each locus, giving the name of the locus, in lower case please, followed by an underscore and a one (1) or a two (2), so that each column name is unique.

From line 4 onwards, the file should contain genotype data, all tab-separated, with no blanks and a string of four asterisks indicating untyped loci.

NOTE, a homozygote should include entries for both alleles i.e., DRB1 0407 should have 0407 entered twice, the other allele should never be left blank, indicated by a dash, or in any way assumed, otherwise they cannot be differentiated from ambiguous typing results.

There should be no trailing blank lines at the end of a file.

References

1. Beazley DM. SWIG: an easy to use tool for integrating scripting languages with C and C++, available at <http://www.swig.org/>, 4th Annual Tcl/Tk Workshop '96, USENIX, July 10–13, 129–139, 1996.
2. Bray T, Paoli J, Sperberg-McQueen CM, Maler E. Extensible Markup Language 1.0 (XML) (<http://www.w3.org/TR/REC-xml/>), Second Edition, World Wide Web Consortium, 2000.
3. Edited by Clark J. Extensible Stylesheet Language Transformations 1.0 (XSLT) (<http://www.w3.org/TR/xslt/>), World Wide Web Consortium, 1999.
4. Dubois PF, Hinsin K, Hugunin J. Numerical Python, available at <http://numpy.sf.net/>. *Computers in Physics* **10**, 262–267, 1996.
5. Ewens W. The sampling theory of selectively neutral alleles. *Theor Pop Biol* **3**, 87–112, 1972.
6. Felsenstein J. PHYLIP Phylogeny Inference Package (Version 3.2) <http://evolution.genetics.washington.edu/phylip.html>. *Cladistics* **5**, 164–166, 1989.
7. Guo SW, Thompson EA. Performing the exact test of Hardy-Weinberg proportion for multiple alleles. *Biometrics* **48**, 361–372, 1992.
8. Ihaka R, Gentleman R. R: A Language for Data Analysis and Graphics <http://www.r-project.org/>. *Journal of Computational and Graphical Statistics* **5**, 299–314, 1996.
9. Raymond M, Rousset F. GENEPOP (version 1.2): population genetics software for exact tests and ecumenicism, available from <ftp://ftp.cfe.cnrs-mop.fr/genepop>. *J Hered* **86**, 248–249, 1995.
10. Schneider S, Roessli D, Excoffier L. Arlequin: A software for population genetics data analysis. <http://lgb.unige.ch/arlequin/>. Ver 2.000. Genetics and Biometry Lab, Dept. of Anthropology, University of Geneva, 2000.
11. Slatkin M. An exact test for neutrality based on the Ewens sampling distribution. *Genetical Research* **64**, 71–74, 1994.
12. Slatkin M. A correction to the exact test based on the Ewens sampling distribution. *Genetical Research* **68**, 259–260, 1996.
13. Watterson G. The homozygosity test of neutrality. *Genetics* **88**, 405–417, 1978.
14. Edited by van Rossum G, Drake FL. Python Reference Manual, available at <http://www.python.org/>, PythonLabs, 2001.