Linking genomics and population genetics with R

Emmanuel Paradis, Thierry Gosselin, Jérome Goudet, Thibaut Jombart, Klaus Schliep

July 20, 2016

Linking genomics and population genetics with R

EMMANUEL PARADIS,* THIERRY GOSSELIN,† JÉRÔME GOUDET,‡ THIBAUT JOMBART,§ KLAUS SCHLIEP,¶

**Institut des Sciences de l'Évolution, Université Montpellier - CNRS - IRD - EPHE, Place Eugène Bataillon – CC 065, 34095 Montpellier cédex 05, France*, †*Institut de Biologie Intégrative et des Systèmes (IBIS), Université Laval, Québec, QC G1V 0A6, Canada*, ‡*Department of Ecology and Evolution and Swiss Institute of Bioinformatics, CH-1015 Lausanne, Switzerland*, §*MRC Centre for Outbreak Analysis and Modelling, Department of Infectious Disease Epidemiology, School of Public Health, Imperial College, London W2 1PG, United Kingdom*, ¶*Department of Biology, University of Massachusetts Boston, Boston, Massachusetts 02125, USA*

Email: Emmanuel.Paradis@ird.fr

Word count: 7817 (including title, abstract, keywords, references and table/figure captions)

Abstract

Population genetics and genomics have developed and been treated as independent fields of study despite having common roots. The continuous progress of sequencing technologies is contributing to (re-)connect these two disciplines. We review the challenges faced by data analysts and software developers when handling very big genetic data sets collected on many individuals. We then expose how R, as a computing language and development environment, proposes some solutions to meet these challenges. We focus on some specific issues that are often encountered in practice: handling and analysing SNP data, handling and reading VCF files, analysing haplotypes and linkage disequilibrium, and performing multivariate analyses. We illustrate these implementations with some analyses of three recently published data sets that contain between 60,000 and 1,000,000 loci. We conclude with some perspectives on future developments of R software for population genomics.

Keywords: multivariate analysis, NGS, R, SNP, VCF

Introduction

An exploration into the history of genetics reveals a complex pattern of interconnected concepts, hypotheses, and research programs. A remarkable historical fact is that population geneticists laid the foundations of their discipline several decades before the physical support of heredity was known. Avery *et al.* (1944) demonstrated that DNA was the molecule coding for genes, a fact that was not immediately accepted by their contemporary geneticists for several reasons, perhaps the main one being that, at this time, it was widely accepted that genes most likely consisted of proteins (Lederberg 1994; Deichmann 2004). Population genetics was a mature field at the time of Avery *et al.*'s finding as witnessed by the long-dated publication of seminal books on the subject (Fisher 1930, e.g.,). Thus, population and molecular genetics have had separate histories at some time and this certainly had an impact on how empirical studies were conducted or motivated. During almost half a century, population genetic studies were based on genotyping individuals on a limited number of loci, especially in conservation-oriented research (Ekblom & Wolf 2014). A crucial preliminary of these studies was to find the 'right genetic marker for the study' (Sunnucks 2000).

At the end of the century, genomic data from human populations started to accumulate making possible to address new questions in the emerging field of population genomics, even though at this time we were "a long way from knowing all the SNPs, even in any given population" mainly because "exhaustive typing is currently prohibitive" (Goldstein & Weale 2001). Nevertheless, it appeared clearly that this wealth of human genetic data could not be interpreted without traditional population genetic concepts such as linkage disequilibrium, genetic drift, or coalescent (Jorde *et al.* 2001). Another result that came out at this time was the peculiarity, from an evolutionary point of view, of human populations compared to other species: low overall genetic diversity, considerable continental-level homogeneity, and small effective population size (Jorde *et al.* 2001). These features associated with the relatively simple biology of humans (lack of polyploidy, long lifespan) and the considerable interest in medically oriented applications of genomic research resulted in a lot of developments for handling and analysing population genomic data. However, these tools did not appear flexible

enough to be applied to organisms with more complex biologies.

In the early twenty-first century, next generation sequencing technologies (NGS) have made possible to access the genotype of an individual at a very large number of loci and even its complete genome sequence (Luikart *et al.* 2003). Even though there are still few species with their genome sequenced, several approaches have been proposed to apply NGS to any living species (Ellegren 2014). Furthermore, the decreasing costs of NGS make increasingly easy to generate data on a large number of loci and to apply these technologies in studies on natural populations (Narum *et al.* 2013; Andrews & Luikart 2014). This is a radical change in point of view for population geneticists compared to almost a century of practice of their discipline. In his review, Sunnucks (2000) listed three components in population genetics studies: genetic markers, statistical methods, and computer programs. We believe that the NGS revolution has solved, at least in principle, the problem of genetic markers. The availability of vast amount of data clearly points to new developments in software. Furthermore, the perspective of merging population and molecular genetics calls for a unified approach to software development.

In this paper, we review the current state of progress in the analysis of population genomic data using R. We illustrate some recent developments with three recently published data sets: the human Y chromosome with 62,042 loci for 1233 individuals (The 1000 Genomes Project Consortium 2015), a set of 1,055,818 phased genotypes for 121 fruit flies (*Drosophila melanogaster*, Kao *et al.* 2015ab), and a set of 61,951 SNPs for 113 dolphins (*Lagenorhynchus* spp., Fernández *et al.* 2016ab). In the next section, we summarize the challenges ahead and explain how R can help meeting them. The following two sections explore some specific issues related to handling and analyses of large data sets in the context of population genomics, and how the R packages presented in this paper integrate with other genomic bioinformatics software. We argue that R solves the issue of software in population genetics in the current context of NGS and in the perspective of a constant flux of genomic data. In the last section, we discuss the future developments of statistical methods which will, hopefully, complete the merging of population and molecular genetics.

Challenges

NGS contrast sharply with their predecessor (the Sanger method) in two important aspects: they generate far larger quantities of data, which is clearly a source of practical difficulties for data storage and analysis (Stephens *et al.* 2015), and they evolve much more quickly. The tremendous potential applications of NGS have stimulated a lot of research and development (Erlich 2015). A new technology or platform can be created and made available rapidly, but tests and validation for reproducibility take time. Meanwhile, competing laboratories or companies may initiate another cycle of testing, assessment, and validation for a new method. Furthermore, the commercialisation of a technology and the training of laboratory staff to use this technology on an appropriate scale also take time. All these time-lags result in a form of inertia in the deployment of NGS in research laboratories. A now famous example is provided by the 454 platform which was discontinued in 2013, just six years after being acquired by a large company—some emerging platforms will probably have a shorter lifespan. It appears that we are now in a situation where we can start sampling in the field without knowing exactly what genotyping or sequencing method will be used in the laboratory.

Nowadays, it is possible to genotype a single individual at thousands or even millions of loci, and population genomic studies typically do this for a few individuals (n), at least far less than the number of loci that are analysed (p). This 'small n, large p' problem raises some difficulties, especially in statistical analyses (see section below on multivariate methods). This situation contrasts with traditional population genetic studies which generally used sampling protocols with $n > p$. With complete genome sequencing the value of p is theoretically bounded by the size of the genome; however, the number of variants (the sites in the genome which are variable among individuals in the sample) depends on the sampling protocol. For instance, the 1000 Genomes Project found almost 38,000,000 variants from 1092 individuals sampled from 14 populations (The 1000 Genomes Project Consortium 2012); this number increased to more than 88,000,000 when 2504 individuals were sampled from 26 populations (The 1000 Genomes Project Consortium 2015). With non-model organisms, we certainly have to expect even more variation since most species are distributed among several subspecies over their range.

The fast pace of NGS development is usually accompanied with a rapid development of bioinformatics tools often resulting in the appearance of a wide variety of data formats. A problem is that very different technologies are used by the different NGS platforms, so that raw data and their assembly differ a lot even though the final goal is the same. For instance, though the FASTQ file format has been adopted as the *de facto* standard for storing raw DNA reads, the emerging NGS platforms in the late 2000's used subtle differences in their respective FASTQ files making the development of common bioinformatics tools problematic (see: `https://en.wikipedia.org/wiki/FASTQ_format` for a review of the variation in FASTQ formats). Another difficulty comes from the fact that population genetics software have traditionally been developed independently of each other and often created their own data file formats, resulting in burdensome conversions necessary along bioinformatics workflows (see Lischer & Excoffier 2012, for a tool to work around this problem). Besides, because of the conceptual separation between population and molecular genetics, most software for population genetics has been developed with classical genetic markers and cannot handle large genomic data sets simply because of memory limitations.

Why R

The quest for standardization is a common and recurrent issue in software development, and applications for data analysis are no exception to this. The 1990's were a key period in this respect: efficient hardware (e.g., 32-bit processors) were widely available, Internet became common, and a new generation of software started to spread. R was developed in this exciting context (Ihaka & Gentleman 1996), and started to be progressively adopted by a wide community of scientists for analysing their data, but also for developing new applications (Vance 2009). Several reasons can explain the success of R: we can cite a few that are relevant for population genetics. R is available for all common platforms and operating systems. Its fast graphical capabilities make it ideal for exploratory analyses. Some generic features of R are extremely useful when manipulating data, such as factors or data structures indexed with names. R is modular so that building new packages or pipelines, or conducting simple analyses is straightforward. R can be easily interfaced with code written in C, C++, and some

other languages, so that computations which are not efficient in R can be recoded in these languages without losing the advantages of the R environment. R has a standard documentation system which makes very easy to find the relevant information when using a new package or a new function (of course, providing the package developers have written the documentation appropriately). R has been widely adopted in statistics courses in many countries so that most students have a fair knowledge of this language. Finally, and not the least, several user interfaces (Emacs+ESS, JGR, R-commander, RKWard, RStudio, Tinn-R, among others) are available which considerably ease the work flow while accomodating each researcher's personal taste.

Naturally, all these features (and others) apply to the analysis of population genetic data which has led to the progressive development of a software environment that is attractive to a wide community of population geneticists. Two remarkable features of this software are worth mentioning here. First, these R packages consider all kinds of genetic markers: DNA sequences, SNP, micro-satellites, allozymes. Second, all situations can be handled: genotypes with single or multiple loci, locus with more than two alleles, all levels of ploidy including cases where ploidy level varies within the same data set, phased and unphased genotypes (Table 1). By contrast to some applications which have been developed specifically for diploid organisms with simple life histories and relatively low polymorphism, R packages are developed to meet the needs of a wide community of molecular ecologists. This underlines the philosophy of R to be inclusive in its developments: from the simplest to the most complex or challenging situations.

Data and files

R has a wide range of tools to handle and analyse DNA sequences (mostly in the package ape; Paradis *et al.* 2004), and traditional allelic data (mostly in the package adegenet; Jombart 2008). In this section, we detail how some other kinds of data that are more specific to NGS can be handled with R.

SNP Data

Molecular genetic studies have revealed that variation in genetic materials can take many forms. In humans, among 88,332,015 genetic variants identified in a sample of 2504 individuals, 95.53% were biallelic single nucleotide polymorphism (SNP), 4.07% insertions–deletions (indels), 0.33% multiallelic SNPs, and 0.07% structural variants (The 1000 Genomes Project Consortium 2015). Thus, biallelic SNPs are by far the most common form of genetic variation in natural populations (see also the examples below). Among the technologies developed to acquire this kind of variants, two are most frequently used: sequencing and genotyping arrays, the second one being able to acquire only biallelic SNPs (which are often called 'SNPs' in short). The approach based on genotyping SNPs can be seen as an intermediate stage between the use of traditional population genetic markers and the more modern genomic sequencing-based approaches which consider the whole set of genetic variation in populations.

Typically, SNP data files store genotypes of individuals at many loci in a matrix form with additional information for each locus such as its position along the chromosome. A difficulty with these files is that there is no widely accepted standard format. For instance, a genotype can be coded in the usual form (e.g., A/A, A/T, or T/T), or as the number of minor alleles in the genotype (0, 1, or 2, if T is the minor allele). However, it appears that the VCF format has recently been adopted as a standard for all kinds of SNP data files (see the example with the dolphin data).

The package `adegenet` provides several tools to read and handle SNP data (Jombart & Ahmed 2011). The core of these tools is the class `"genlight"` designed to store SNP data in a compact way: for example, such an object with one million SNPs for 100 individuals uses about 16 Mb of RAM (or active memory) which is less than 1% of the available RAM on most modern computers. Besides, `adegenet` has several functions specially designed for the analysis of `"genlight"` data, such as `glPca` which performs a principal components analysis (PCA; see below), `glPlot` which plots a set of SNPs, or `glSim` which simulates efficiently SNP data. The function `read.PLINK` in the same package can read some SNP files. More generally, SNPs are stored in plain-text files in tabular form, so standard R functions (e.g., `read.table`) can be used to parse such files.

VCF files

Over the last few years, the Variant Call Format (VCF) has appeared as a standard for storing population genetic data from NGS sequencing or genotyping (Danecek *et al.* 2011). By contrast to SNP data files, VCF files can contain data on all types of genetic variation. VCF files are text files storing a lot of information, including on the process used to create it from the raw sequencing reads (variant calling). The genetic data are stored in a matrix-like layout where the individuals are the columns and the loci are the rows.

Several R packages have tools to analyse VCF files: two of them, stacksr and vcfR, are described in this issue. Bioconductor has the package VariantAnnotation (Obenchain *et al.* 2014) for annotating VCF files. We focus here on the functions in pegas because this package is dedicated to population genetics (Paradis 2010). vcfR provides functions to assess the quality of the VCF file (Knaus & Grünwald 2017) together with functions for data conversion (Table 2).

VCF files can be very big, so a strategy is to first scan the files before reading more detailed information such as the genotypes. pegas follows this strategy by providing several functions: the main ones are `VCFloci` and `read.vcf`. The first function scans a VCF file and extracts the information on the loci; the second one reads the individual genotypes with the possibility to select a subset. `VCFloci` uses a specific mechanism to read big files: the file is scanned sequentially by chunks of one billion bytes (1 Gb; the chunk size can be modified by the user). The function then stores a small table with the name of the file, the size of each chunk adjusted to meet the nearest end of a line, and the number of loci in each chunk. With this mechanism, not more than 1 Gb of RAM is used at the same time whatever the size of the file. The running time increases roughly linearly with the file size: it takes about 20 sec to scan an 8 Gb file (compressed or not) with about 500,000 loci. For smaller files, the running time depends on whether the file is compressed or not: for a 200 Mb file with 62,000 loci, this takes 1 sec or 0.2 sec, respectively.

The output of `VCFloci` is a data frame (R's usual data table) with nine columns where each row is a locus and the columns are the information for each locus from the

VCF file. The information correspond to the mandatory VCF header fields (see Danecek *et al.* 2011): chromosome (CHROM), position (POS), unique identifier (ID), reference and alternative alleles (REF and ALT), phred-scaled quality score (QUAL), site-filtering information (FILTER), a semicolon-separated list of additional annotation metadata (INFO) and the FORMAT field containing genotype data information (often a semicolon-separated list). The data frame can be manipulated to extract information with standard R functions (Box 1). The function `VCFheader` prints the header of a VCF file which, usually, contains details on these different fields.

The genotypes are then read by `read.vcf`. At this stage, it is good to think if it is interesting or feasible to read a complete VCF file. `pegas` takes roughly four bytes of RAM to store a single genotype; so (using the above notation), about $4np$ bytes are needed for a given genetic data set. With the rule of thumb that data in R should not use more than 25% of the available RAM to perform "comfortable" analyses, one million loci for 1000 individuals appears as a reasonable upper limit on a computer with 16 Gb of RAM. Furthermore, the user may be interested in only some loci depending on the type of genetic variant, position on the chromosome, or else. A strategy is to locate these loci by finding the rows of the output of `VCFloci` using standard R data manipulation (Box 1). Similar operations can be done on the field `"CHROM"` (if loci from several chromosomes are in the same file), `"QUAL"`, or `"FILTER"`. The field `"INFO"` requires a special attention because it includes detailed information on each locus. To make this information easier to digest, `pegas` provides the function `getINFO` to extract a specific information from this character string; by default this is the sequencing depth (DP), but the option `what` makes possible to extract another information, for instance, the variant type (Box 2).

For small VCF files, `read.vcf` can be used directly without calling `VCFloci` first, but the latter is still needed to get the information on the loci.

Analysis of population genomic data with R

Haplotypes and Linkage Disequilibrium

The possibility to investigate the genetics of populations at many loci of individuals brings the perspective to study the dynamics of genomes in populations. An exciting opportunity is to identify parts of species genomes which evolve as a single unit (Andrews & Luikart 2014). Several approaches with NGS allow one to infer genotype phasing for diploid or polyploid organisms, either directly by sequencing long fragments of DNA (e.g., McCoy *et al.* 2014), or with high coverage sequencing of shorter fragments (Nielsen *et al.* 2011). With phased genotypes, it is straightforward to reconstruct the haplotypes of each individual. `pegas` has the generic function `haplotype` which does this operation (it also works with DNA sequences). Linkage disequilibrium (LD) is the most widely used statistical method to identify alleles at different loci that are statistically associated. It is also of central importance in association studies of genetic diseases (Goldstein & Weale 2001; Jorde *et al.* 2001). The function `LDscan` permits to analyse LD by calculating pairwise correlation coefficients (r^2) for a series of biallelic loci. The output of this function can be plotted with `LDmap`; this function has an option `POS` to plot the correlation coefficients together with the positions of the loci on the chromosome. Figure 1 shows the LD at two scales for the chromosome 2L of the fruit fly data: for the first 100 loci and for 100 loci equally spaced on the chromosome (Box 3). Two other functions analyse LD for a pair of loci with any number of alleles: `LD` and `LD2` for phased and unphased genotypes, respectively. These two functions return detailed statistics and tests (Schaid 2004; Zaykin *et al.* 2008).

Multivariate Methods

Multivariate methods are common when analysing big data sets because of their abilities to provide dimension reduction, that is to provide a summary of many variables into a few, usually linear, combinations of them. Thus, population genetics has used these methods for some time (Westfall & Conkle 1992; Guinand 1996; Moazami-Goudarzi & Laloë 2002). The package `adegenet` implements a lot of tools for the multivariate analysis of genetic data (Jombart *et al.* 2009). Some of the methods of particular interest here

include the spatial principal components analysis (sPCA; Jombart *et al.* 2008) and the discriminant analysis of principal components (DAPC; Jombart *et al.* 2010) which is illustrated below.

In spite of their efficiency with large tables, multivariate methods are penalized when analysing very large data sets, for instance, when n and p are larger than 10,000. Some approaches have been developed to decompose very large matrices without using traditional methods such as eigen or singular value decomposition (Halko *et al.* 2011). These approaches are based on random matrices and aim to approximate the first few principal components. Abraham & Inouye (2014) implemented this approach to perform a PCA on genomic microarray data collected on a large sample of individuals ($n >$ 10,000). Their implementation is available as the R package flashpca distributed on GitHub (https://github.com/gabraham/flashpca). This approach is quite straightforward to code in R and is available in several forms with more or less accuracy (Halko *et al.* 2011).

Discriminant analysis (DA) is a multivariate method which searches for linear combinations of variables resulting in the strongest discrimination of groups identified from the data (Fisher 1936). Under some assumptions, it is possible to quantify the reliability of the inferred assignments. Thus, DA offers a powerful approach to assess the presence of structure in genetic data with many loci. However, because it is based on matrix decomposition algorithms, it suffers from the same limitations than PCA with very big data sets, and these methods need to be adapted to be used on genomic data. The DAPC gives a solution to this problem: it can be used to assess group structure in genetic diversity with large datasets, and is also implemented separately for `"genlight"` objects (Jombart & Ahmed 2011). The main idea behind the DAPC is to first perform a data reduction using a PCA. Typically, less than 100 principal axes are sufficient. If group priors are missing, a sequential k-means clustering with model selection based on the Bayesian information criterion (BIC) is done to infer prior clusters. The next step is to perform a discriminant analysis in the reduced data space using the prior clusters returned by the k-means clustering (or the cluster definition provided by the user if available). The results can be graphically visualized in two ways: by plotting the projections of the individuals on discriminant axes (as usual in multivariate analyses), or

by examining the relative posterior probabilities to the different clusters inferred from the discriminant analysis (as usual in stochastic assignment methods).

We consider here the dolphin data and try to show evidence for geographical structure (Fernández *et al.* 2016a). The data are provided as a VCF file (simply named 'vcf') available from Dryad (Fernández *et al.* 2016b). Because the multivariate methods are implemented in `adegenet`, we have to convert the data read by `pegas` into the appropriate class (Box 4). Remarkably, none of these analyses take significant running time (in fact, the longest operation is to convert from the class `"loci"` to the class `"genind"`).

Population Genetics and Genomic Bioinformatics

The main strength of R, and maybe the most attractive one for data analysis, is the possibility to integrate a great variety of methods and tools while using the same programming language. In the past few years, the analysis of genetic data has changed quite radically. Until the early twenty-first century, population geneticists used to get their data from the laboratory on their physical support (e.g., electrophoresis gels), then input the data in the computer, and analyse them with specialised software. With the advent of NGS, genetic data are directly acquired on the computer, but an additional step is required: the identification of alleles from the many (hundreds of millions) sequencing reads. This step is called 'variant calling' and requires highly specialised software. Because NGS technologies are evolving very fast (see above), this software is also evolving very fast. Fortunately, the past few years have seen the development of more integrated bioinformatics tools to accomplish such tasks. The R package `poRe` has been recently released to analyse data from the the mobile nanopore sequencer MinION (Watson *et al.* 2015). This package takes profit of R's tools such as scanning a directory with many files, searching for patterns in file names, and reading files in a specific format (here HDF5), to extract and plot summary statistics from the sequencing runs, as well as export data into FASTQ or FASTA format.

Bioconductor is a long-term effort to produce R packages for the analysis of large-scale genomic data sets (Gentleman *et al.* 2005; Huber *et al.* 2015). Bioconductor's website (`www.bioconductor.org`) hosts a suite of integrated R packages mostly dedicated to the analysis of expression data. Several packages have been released for the

analysis of DNA microarray data (e.g., Dunning *et al.* 2007; Ritchie *et al.* 2009; Morgan 2015). The package ape has functions to convert DNA data from Bioconductor classes into the "DNAbin" class from which population and evolutionary analyses can be performed.

A crucial aspect of R packages which permits the integration of different tasks is the stability of the packages themselves and especially of the data classes defined therein (Tables 1, 3). Several R packages for evolutionary genetic analyses are now more than one decade-old and have been widely used and tested, so that their contents are now reliable for a wide range of applications, and package developers have the possibility to work with them to develop new tools and methods (e.g., Paradis 2012; Lawrence & Morgan 2014).

Perspectives

Remarkable progress has been accomplished in the development of software for population genetics in the context of analysing data sets from NGS. There is no doubt that this will continue in the years to come where population genetic data will become more and more important to assess biodiversity dynamics and adaptations of species in a changing environment. We discuss below three specific areas where significant progress is likely to be achieved in the coming years.

As apparent from the contributions to this Special Issue, a lot of effort has been devoted to improve the overall quality of the software dedicated to population genetics. On the other hand, hardware has continued to improve. For instance, multi-core processors are now the rule rather than the exception, so that parallel computing can be accomplished on almost all computers. The adaptation of genetic software to this has been uneven. Lawrence & Morgan (2014) give some examples using Bioconductor, illustrating the issue that implementing a parallel algorithm to accomplish a given task may not always be beneficial. R has a built-in capacity to parallelize computations using the package parallel. The functions in parallel can be used with most R packages described in this article, for instance, by distributing a calculation to several cores of a processor. However, there is certainly some improvements possible since a lot of the computations

used in population genetics implies repetitive and independent calculations (e.g., for allele frequencies by population or for LD).

NGS raw data are more or less noisy because they are generated through some biological and electronical processes which may be affected by random variation. Besides these measurement errors, intrinsic biological processes induce another form of variation, for instance because of cell-specific genetic changes. Lynch (2008) addressed the issue of inferring population genetic parameters using NGS data (see Korneliussen *et al.* 2013, for a more recent contribution). Blischak *et al.* (2016) developed a hierarchical Bayesian model to assess genotype uncertainty in autopolyploids using raw read counts. They implemented their method in the R package polyfreqs (available on CRAN). We take this opportunity to emphasize the importance of open source software for the future of population genomics. O'Rawe *et al.* (2015) warned about the risk that commercial NGS applications tend to ignore uncertainty in genotyping inference. As a sign of this risk, these authors mention that several applications for haplotype or variant call are proprietary and closed-source. Considering the continuously increasing diversity of NGS technologies and the increasing range of their applications, a special attention to uncertainty in statistical inference using NGS data will be crucial.

Environmental monitoring already allows acquisition of data in real time, and recent developments in NGS (e.g., nanopore sequencing) strongly suggest that similar things will soon be accomplished with population genetic data. This is a crucial challenge for developing efficient software and making sense of the results in real time. Facing this forthcoming challenge will mean having to combine data analysis with theoretical population genetics in order to help us meet issues related to global change, biodiversity conservation, and pathogen dynamics. It will surely be useful to remember the words of caution—if not wisdom—from Gower (2008): "Automatic instrumentation ensures that there is no lack of large data sets. This is something new but I sometimes think that we are getting perilously close to the search for the philosophers stone. If only we could find the right recipe, great truths would be revealed by analysing vast masses of data."

Acknowledgements

We are grateful to David Winter and two anonymous reviewers for their constructive comments on a previous version of this paper. The resource reported in this paper started at the Population Genetics in R Hackathon, which was held in March 2015 at the National Evolutionary Synthesis Center (NESCent) in Durham, NC, with the goal of addressing interoperability, scalability, and workflow building challenges for the population genetics package ecosystem in R. The authors were participants in the hackathon, and are indebted to NESCent (NSF #EF-0905606) for hosting and supporting the event. Many thanks to all participants of the hackathon for their feedback and enthusiasm during and after the event. TJ is funded by the Medical Research Council Centre for Outbreak Analysis and the National Institute for Health Research - Health Protection Research Unit (NIHR HPRU) in Modelling Methodology at Imperial College London in partnership with Public Health England (PHE). KS is funded by a grant from the National Science Foundation (DEB 1350474). This is publication ISEM 2016-150.

References

Abraham G, Inouye M (2014) Fast principal component analysis of large-scale genome-wide data. *PLoS ONE*, **9**, e93766.

Andrews KR, Luikart G (2014) Recent novel approaches for population genomics data analysis. *Molecular Ecology*, **23**, 1661–1667.

Avery OT, Macleod CM, Mccarty M (1944) Studies on the chemical nature of the substance inducing transformation of pneumococcal types. Induction of transformation by a desoxyribonucleic acid fraction isolated from Pneumococcus type III. *Journal of Experimental Medicine*, **79**, 137–158.

Blischak PD, Kubatko LS, Wolfe AD (2016) Accounting for genotype uncertainty in the estimation of allele frequencies in autopolyploids. *Molecular Ecology Resources*, **16**, 742–754.

Danecek P, Auton A, Abecasis G, *et al.* (2011) The variant call format and VCFtools. *Bioinformatics*, **27**, 2156–2158.

Deichmann U (2004) Early responses to Avery et al.'s paper on DNA as hereditary material. *Historical Studies in the Physical and Biological Sciences*, **34**, 207–232.

Dunning MJ, Smith ML, Ritchie ME, Tavaré S (2007) beadarray: R classes and methods for Illumina bead-based data. *Bioinformatics*, **23**, 2183–2184.

Ekblom R, Wolf JBW (2014) A field guide to whole-genome sequencing, assembly and annotation. *Evolutionary Applications*, **7**, 1026–1042.

Ellegren H (2014) Genome sequencing and population genomics in non-model organisms. *Trends in Ecology & Evolution*, **29**, 51–63.

Erlich Y (2015) A vision for ubiquitous sequencing. *Genome Research*, **25**, 1411–1416.

Fernández R, Schubert M, Vargas-Velázquez AM, *et al.* (2016a) A genomewide catalogue of single nucleotide polymorphisms in white-beaked and Atlantic white-sided dolphins. *Molecular Ecology Resources*, **16**, 266–276.

Fernández R, Schubert M, Vargas-Velázquez AM, *et al.* (2016b) Data from: A genomewide catalogue of single nucleotide polymorphisms in white-beaked and Atlantic white-sided dolphins. Dryad Digital Repository. `http://dx.doi.org/10.5061/dryad.sd28r`.

Fisher RA (1930) *The genetical theory of natural selection (a complete variorum edition, 1999)*. Oxford University Press, Oxford.

Fisher RA (1936) The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, **7**, 179–188.

Gentleman R, Carey VJ, Huber W, Irizarry RA, Dudoit S, eds. (2005) *Bioinformatics and computational biology solutions using R and Bioconductor*. Springer, New York.

Goldstein DB, Weale ME (2001) Population genomics: linkage disequilibrium holds the key. *Current Biology*, **11**, R576–R579.

Goudet J (2005) Hierfstat, a package for R to compute and test variance components and F-statistics. *Molecular Ecology Notes*, **5**, 184–186.

Gower JC (2008) The biological stimulus to multidimensional data analysis. *Electronic Journal for History of Probability and Statistics*, **4**. `http://www.jehps.net/Decembre2008/Gower.pdf`.

Guinand B (1996) Use of a multivariate model using allele frequency distributions to analyse patterns of genetic differentiation among populations. *Biological Journal of*

the Linnean Society, **58**, 173–195.

Halko N, Martinsson PG, Tropp JA (2011) Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, **53**, 217–288.

Huber W, Carey VJ, Gentleman R, *et al.* (2015) Orchestrating high-throughput genomic analysis with Bioconductor. *Nature Methods*, **12**, 115–121.

Ihaka R, Gentleman R (1996) R: a language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, **5**, 299–314.

Jombart T (2008) *adegenet*: a R package for the multivariate analysis of genetic markers. *Bioinformatics*, **24**, 1403–1405.

Jombart T, Ahmed I (2011) *adegenet 1.3-1*: new tools for the analysis of genome-wide SNP data. *Bioinformatics*, **27**, 3070–3071.

Jombart T, Devillard S, Balloux F (2010) Discriminant analysis of principal components: a new method for the analysis of genetically structured populations. *BMC Genetics*, **11**, 94.

Jombart T, Devillard S, Dufour AB, Pontier D (2008) Revealing cryptic spatial patterns in genetic variability by a new multivariate method. *Heredity*, **101**, 92–103.

Jombart T, Pontier D, Dufour AB (2009) Genetic markers in the playground of multivariate analysis. *Heredity*, **102**, 330–341.

Jorde LB, Watkins WS, Bamshad MJ (2001) Population genomics: a bridge from evolutionary history to genetic medicine. *Human Molecular Genetics*, **10**, 2199–2207.

Kamvar ZN, Tabima JF, Grünwald NJ (2014) Poppr: an R package for genetic analysis of populations with clonal, partially clonal, and/or sexual reproduction. *PeerJ*, **2**, e281.

Kao JY, Zubair A, Salomon MP, Nuzhdin SV, Campo D (2015a) Data from: Population genomic analysis uncovers African and European admixture in *Drosophila melanogaster* populations from the south-eastern United States and Caribbean Islands. Dryad Digital Repository. `http://dx.doi.org/10.5061/dryad.446sv.2`.

Kao JY, Zubair A, Salomon MP, Nuzhdin SV, Campo D (2015b) Population genomic analysis uncovers African and European admixture in *Drosophila melanogaster* populations from the south-eastern United States and Caribbean Islands. *Molecular Ecology*, **24**, 1499–1509.

Knaus BJ, Grünwald NJ (2017) VcfR: an R package to manipulate and visualize VCF format data. *Molecular Ecology Resources*, **17**, 000–000.

Korneliussen TS, Moltke I, Albrechtsen A, Nielsen R (2013) Calculation of Tajima's D and other neutrality test statistics from low depth next-generation sequencing data. *BMC Bioinformatics*, **14**, 289.

Lawrence M, Morgan M (2014) Scalable genomics with R and Bioconductor. *Statistical Science*, **29**, 214–226.

Lederberg J (1994) The transformation of genetics by DNA: an anniversary celebration of Avery, MacLeod and McCarty (1944). *Genetics*, **136**, 423–426.

Lischer HEL, Excoffier L (2012) PGDSpider: an automated data conversion tool for connecting population genetics and genomics programs. *Bioinformatics*, **28**, 298–299.

Luikart G, England PR, Tallmon D, Jordan S, Taberlet P (2003) The power and promise of population genomics: from genotyping to genome typing. *Nature Reviews Genetics*, **4**, 981–994.

Lynch M (2008) Estimation of nucleotide diversity, disequilibrium coefficients, and mutation rates from high-coverage genome-sequencing projects. *Molecular Biology and Evolution*, **25**, 2409–2419.

McCoy RC, Taylor RW, Blauwkamp TA, *et al.* (2014) Illumina TruSeq synthetic long-reads empower *de novo* assembly and resolve complex, highly-repetitive transposable elements. *PLoS ONE*, **9**, e106689.

Moazami-Goudarzi K, Laloë D (2002) Is a multivariate consensus representation of genetic relationships among populations always meaningful? *Genetics*, **162**, 473–484.

Morgan AP (2015) argyle: a R package for analysis of Illumina genotyping arrays. *Genes Genomes Genetics*, **6**, 281–6.

Narum SR, Buerkle CA, Davey JW, Miller MR, Hohenlohe PA (2013) Genotyping-by-sequencing in ecological and conservation genomics. *Molecular Ecology*, **22**, 2841–2847.

Nielsen R, Paul JS, Albrechtsen A, Song YS (2011) Genotype and SNP calling from next-generation sequencing data. *Nature Reviews Genetics*, **12**, 433–441.

Obenchain V, Lawrence M, Carey V, Gogarten S, Shannon P, Morgan M (2014) VariantAnnotation: a Bioconductor package for exploration and annotation of genetic

variants. *Bioinformatics*, **30**, 2076–2078.

O'Rawe JA, Ferson S, Lyon GJ (2015) Accounting for uncertainty in DNA sequencing data. *Trends in Genetics*, **31**, 61–66.

Paradis E (2010) pegas: an R package for population genetics with an integrated–modular approach. *Bioinformatics*, **26**, 419–420.

Paradis E (2012) *Analysis of Phylogenetics and Evolution with R (Second Edition)*. Springer, New York.

Paradis E, Claude J, Strimmer K (2004) APE: analyses of phylogenetics and evolution in R language. *Bioinformatics*, **20**, 289–290.

Ritchie ME, Carvalho BS, Hetrick KN, Tavaré S, Irizarry RA (2009) R/Bioconductor software for Illumina's Infinium whole-genome genotyping BeadChips. *Bioinformatics*, **25**, 2621–2623.

Schaid DJ (2004) Linkage disequilibrium testing when linkage phase is unknown. *Genetics*, **166**, 505–512.

Stephens ZD, Lee SY, Faghri F, *et al.* (2015) Big data: astronomical or genomical? *PLoS Biology*, **13**, e1002195.

Sunnucks P (2000) Efficient genetic markers for population biology. *Trends in Ecology & Evolution*, **15**, 199–203.

The 1000 Genomes Project Consortium (2012) An integrated map of genetic variation from 1,092 human genomes. *Nature*, **491**, 56–65.

The 1000 Genomes Project Consortium (2015) A global reference for human genetic variation. *Nature*, **526**, 68–74.

Vance A (2009) Data analysts captivated by R's power. *New York Times*. January 6, 2009.

Watson M, Thomson M, Risse J, *et al.* (2015) poRe: an R package for the visualization and analysis of nanopore sequencing data. *Bioinformatics*, **31**, 114–115.

Westfall RD, Conkle MT (1992) Allozyme markers in breeding zone designation. *New Forests*, **6**, 279–309.

Winter DJ (2012) mmod: an R library for the calculation of population differentiation statistics. *Molecular Ecology Resources*, **12**, 1158–1160.

Zaykin DV, Pudovkin A, Weir BS (2008) Correlation-based inference for linkage

546 disequilibrium with multiple alleles. *Genetics*, **180**, 533–545.

Box 1 *Scanning and reading VCF files*

First we scan the human Y chromosome data with `VCFloci`, print the number of loci and the labels of the columns with `names`:

```
> info.Y <- VCFloci("chrY.vcf.gz")
Scanning file chrY.vcf.gz
 171.6615 Mb
Done.
> nrow(info.Y)
[1] 62042
> names(info.Y)
[1] "CHROM"  "POS"    "ID"     "REF"    "ALT"    "QUAL"
[7] "FILTER" "INFO"   "FORMAT"
```

The number of individuals in the VCF file are obtained with the function `VCFlabels` which extracts their labels (or identifiers):

```
> labs <- VCFlabels("chrY.vcf.gz")
> length(labs)
[1] 1233
> head(labs)
[1] "HG00096" "HG00101" "HG00103" "HG00105" "HG00107"
[6] "HG00108"
```

Then we read the genotypes with `read.vcf`. Here, we select the loci in a 1 Mb region of the Y chromosome between positions 5,000,000 and 6,000,000 which can be found with:

```
> sel <- which(info.Y$POS > 5e6 & info.Y$POS <= 6e6)
> length(sel)
[1] 5
```

`read.vcf` has two ways to specify which loci to read: either using the options `from` and `to` (which are set to 1 and 10,000 by default), or using the option `which.loci` which requires a vector of integers specifying the indices of the loci to read:

```
> x <- read.vcf("chrY.vcf.gz", which.loci = sel)
Reading 5 / 5 loci.
Done.
> x
Allelic data frame: 1233 individuals
                    5 loci
```

If the user wants to read all loci with a single command, the following can be done:

```
> x <- read.vcf("chrY.vcf.gz", to = nrow(info.Y))
```

Box 2 *Going further with the INFO field in VCF files*

Here we use the information returned by `VCFloci`, such as the field CHROM (if loci from several chromosomes are in the same file), to select some loci to be read. The field INFO requires a special attention because it includes detailed information on each locus:

```
> info.Y$INFO[1]
[1] "AA=G;AC=22;AF=0.0178427;AN=1233;DP=84761;NS=1233;\
AMR_AF=0.0000;AFR_AF=0.0000;EUR_AF=0.0000;SAS_AF=0.0000;\
EAS_AF=0.0451;VT=SNP;EX_TARGET"
```

To make this information easier to digest, the function `getINFO` helps to extract a specific information from this character string; by default this is the sequencing depth (DP), but the option `what` makes possible to extract any other information, for instance, the variant type:

```
> VT.Y <- getINFO(info.Y, what = "VT")
> table(VT.Y)

INDEL   MNP   SNP    SV
 1314   113 60505   110
```

The function `VCFheader` reads the header of the VCF file where details on the abbreviations used can be found:

```
> cat(VCFheader("chrY.vcf.gz"))
....
##INFO=<ID=VT,Number=.,Type=String,Description="indicates what type\
of variant the line represents"
....
```

Thus, we can now locate which loci are SNPs:

```
> sel.snp <- which(VT.Y == "SNP")
> length(sel.snp)
[1] 60505
```

As usual with R, it is possible to combine these logical comparisons with the AND operator (`&`), for instance, to locate the SNPs within a 2 Mb region:

```
> sel <- which(VT.Y == "SNP" & info.Y$POS > 1e6 & info.Y$POS <= 3e6)
> length(sel)
[1] 1231
```

To illustrate how to use other information returned by `VCFloci`, we scan the fruitfly data (Kao *et al.* 2015a) which include genotypes from all chromosomes in a single file:

```
> fl <- "global.pop.GATK.SNP.hard.filters.V3.phased_all.pop.maf.05.recode.vcf.gz"
> info.fly <- VCFloci(fl)
> table(info.fly$CHROM)

    2L     2R     3L     3R      R      X
224253 193675 214235 270619      1 153035
```

Box 3 *Analysis of linkage disequilibrium*

We read the fruitfly data and call the two functions `LDscan` and `LDmap` on the first 100 loci (we could use the option `to` as above to read only 100 loci, but this gives us the opportunity to show how to subset a data frame of genotypes). The positions of the loci are stored in the column `POS` output by `VCFloci` (Box 2), so we extract the positions for these 100 loci and pass them as the (optional) second argument of `LDmap` (Fig. 1a):

```
> x <- read.vcf(fl)
> res <- LDscan(x[, 1:100])
> LDmap(res, info.fly$POS[1:100])
```

We now select 100 loci regularly spaced from 1 to 224,253 (this value comes from the last tabulation in Box 2) and using `ceiling` to ensure we have only integers:

```
> s <- ceiling(seq(1, 224253, length.out = 100))
```

We then repeat the LD analysis using the appropriate POS information (Fig. 1b):

```
> xs <- read.vcf(fl, which.loci = s)
> res2 <- LDscan(xs)
> LDmap(res2, info.fly$POS[s])
```

Box 4 *DAPC with the dolphin data*

We read 1000 loci and convert them to the class `"genind"`:

```
> x2 <- read.vcf("vcf", to = 1000)
> g2 <- loci2genind(x2)
```

We can now call the method implemented in the function `find.clusters`:

```
> o2 <- find.clusters(g2)
```

When called with no other option, the function displays two graphs (Fig. 2) and asks the user to input the number of principal components (PCs) and the number of clusters to retain. The first graph shows that 40 PCs retain about 80% of the variance, so we select this number. The second graph shows a lowest value of BIC with seven clusters. However, the present data has eight populations (see below) so we select this number of clusters (K) for this analysis (we could use simply the original populations).
The output is a list with several elements, including `grp` giving the cluster assignment of the individuals. The labels of the individuals include the geographical origin of the samples (as often the case with this kind of data set). So it is simple to extract this information, and cross-tabulate it with the cluster assignment done by `find.clusters`:

```
> head(names(o2$grp))
[1] "FAE_2"   "FAE_609" "FAE_660" "FAE_667" "FAE_668" "FAE_671"
> pop <- gsub("_.*", "", names(o2$grp))
> cluster <- o2$grp
> table(pop, cluster)
     cluster
pop    1  2  3  4  5  6  7  8
  DEN  0  0  0  0  4  2  0  0
  FAE  0  9  1  3  2  0  2  1
  FRA  0  0  0  0  1  0  1  0
  GER  3  0  0  0  5  3  1  0
  ICE  0  0  0  0 15  2  0  0
  IRE  0  9  0  0  2  2  3  0
  NOR  0  0  0  0  5  2  0  0
  SCO  0  5  0  0 24  1  2  0
```

There does not seem to be a very good match between the originally sampled populations and the cluster assignments. Thus, we repeat the analysis with a smaller number of clusters ($K = 4$) in a non-interactive way by specifying the required options:

```
> o2b <- find.clusters(g2, n.pca = 40, n.clust = 4)
```

As above, we extract the 'pop' and 'cluster' information (not shown):

```
> table(pop, cluster)
     cluster
pop    1  2  3  4
  DEN  0  6  0  0
  FAE  0  6  4  8
  FRA  0  1  0  1
  GER  6  5  0  1
  ICE  2 15  0  0
  IRE  2  4  0 10
  NOR  2  5  0  0
  SCO  0 26  0  6
```

We now perform the DAPC with 4 groups asking to output two discriminant axes:

```
> d2b <- dapc(g2, cluster, n.pca = 40, n.d = 2)
```

The output contains several elements, including the coordinates of the individuals and the group means. The plots can now be done with R's standard functions (see Supplementary Information) or with `adegenet`'s functions (Figs. 3, 4):

```
> col <- c("red", "blue", "gold", "darkgreen")
> scatter(d2b, posi.da = "topright", col = col)
> compoplot(d2b, legend = FALSE, col = col)
```

Table 1: Major R data classes for population genetics and genomics. The package in parentheses is where the class is defined and which contains the main functions for data manipulation and visualization.

Class	Data type	Main features	Missing data
DNAbin (ape)	DNA sequences	Single or multiples sequences, aligned or not	Ambiguity code + ?*
genind (adegenet)	Allelic (allele-centered)	Any level of ploidy; ploidy may vary among loci and among individuals	NA†; null allele(s) can be specified at input
genlight (adegenet)	biallelic SNP	Any level of ploidy	Not supported
loci (pegas)	Allelic (locus-centered)	Any level of ploidy; ploidy may vary among loci and among individuals; genotypes may be phased or not	NA†; null allele(s) can be specified from R

*The question mark (?) is not part of the IUPAC ambiguity code.
†Standard missing value in R.

Table 2: Data inter-operability.

Class	File		Class conversion	
	Input	Output	From	To
DNAbin	FASTA Phylip Clustal	FASTA Phylip	Bioconductor*	genlight
genind	Genetix Structure Fstat Genepop GenAlex		loci	loci
genlight	PLINK		DNAbin	
loci	Tabular files† Genetix VCF	Tabular files†	genind	genind
vcfR	VCF	VCF		DNAbin genind loci

*Several classes are supported.
†Tab-delimined, CSV, and similar formats.

Table 3: Main R packages for population genetics and genomics. The packages not cited in the text are hierfstat (Goudet 2005), poppr (Kamvar *et al.* 2014), and mmod (Winter 2012).

Package	Main methods implemented
adegenet*	Multivariate methods (sPCA, DAPC)
ape	Evolutionary distances, distance-based phylogenetics, phylogenetic bootstrap
hierfstat*	Hierarchical F-statistics
pegas	Population differentiation (HWE,, AMOVA), haplotype networks, linkage maps
poppr*	Applications to clonal organisms, distances for microsatellites
mmod*	Population differentiation (F_{ST}, G_{ST}, Φ_{ST}, D)
stacksr	Interface to STACKS
vcfR	Tools for FASTA, GFF, and VCF files

*These packages use the class `"genind"`.

Fig. 1 LD maps at two different genomic scales for the fruit fly data: (a) for the 100 first loci on the chromosome 2L, and (b) for 100 loci regularly spaced along this chromosome. The horizontal axis indicates the position of the loci on the chromosome. The linkage coefficients (r^2) between each pair of loci are indicated as coloured squares: the squares at the bottom of the triangle are for nearby loci, whereas the square at the top is for the two most distant loci.

Fig. 2 Results of finding clusters with the dolphin data. (a) Cumulated variance explained by the principal components (PCs). (b) Bayesian information criterion (BIC) for the different values of K (number of clusters).

Fig. 3 Results of the DAPC with the dolphin data.

Fig. 4 Posterior probabilities of group membership from the DAPC with the dolphin data. The color scheme is the same than in Fig 3.

Figure 1:

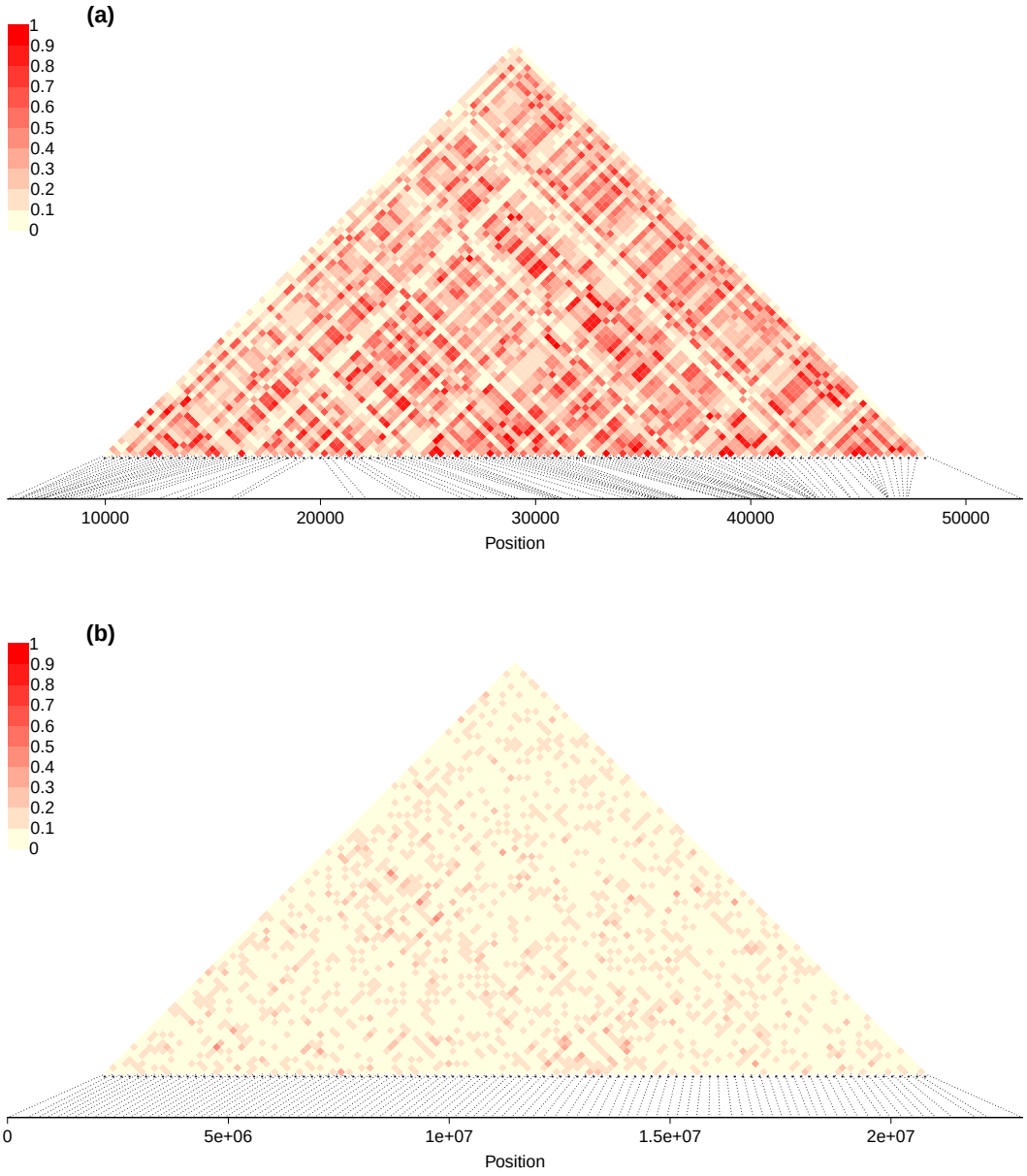

(a)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
10000
20000
30000
40000
50000
Position
(b)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
5e+06
1e+07
1.5e+07
2e+07
Position

Figure 2:

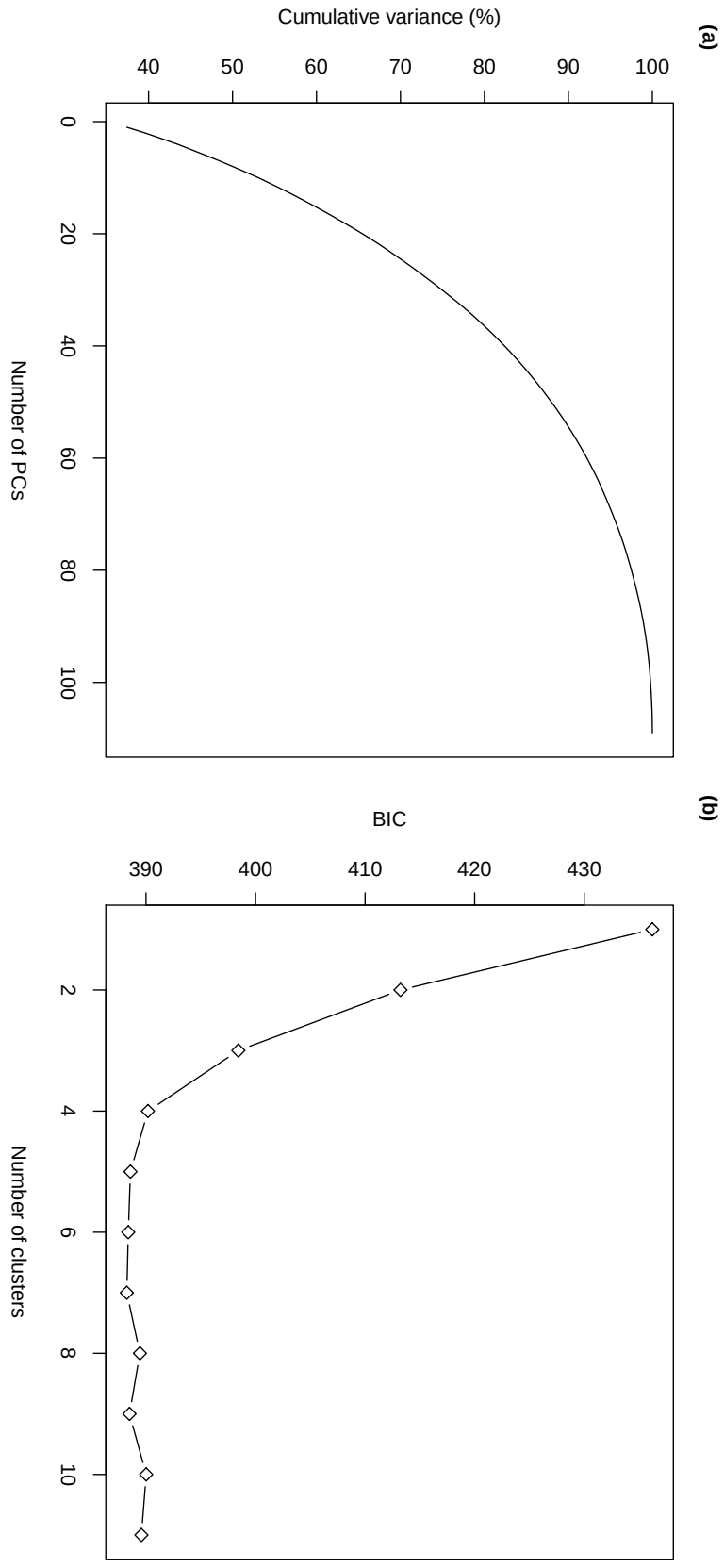

(a)
Cumulative variance (%)
40
50
60
70
80
90
100
Number of PCs
0
20
40
60
80
100
(b)
BIC
390
400
410
420
430
Number of clusters
2
4
6
8
10

Figure 3:

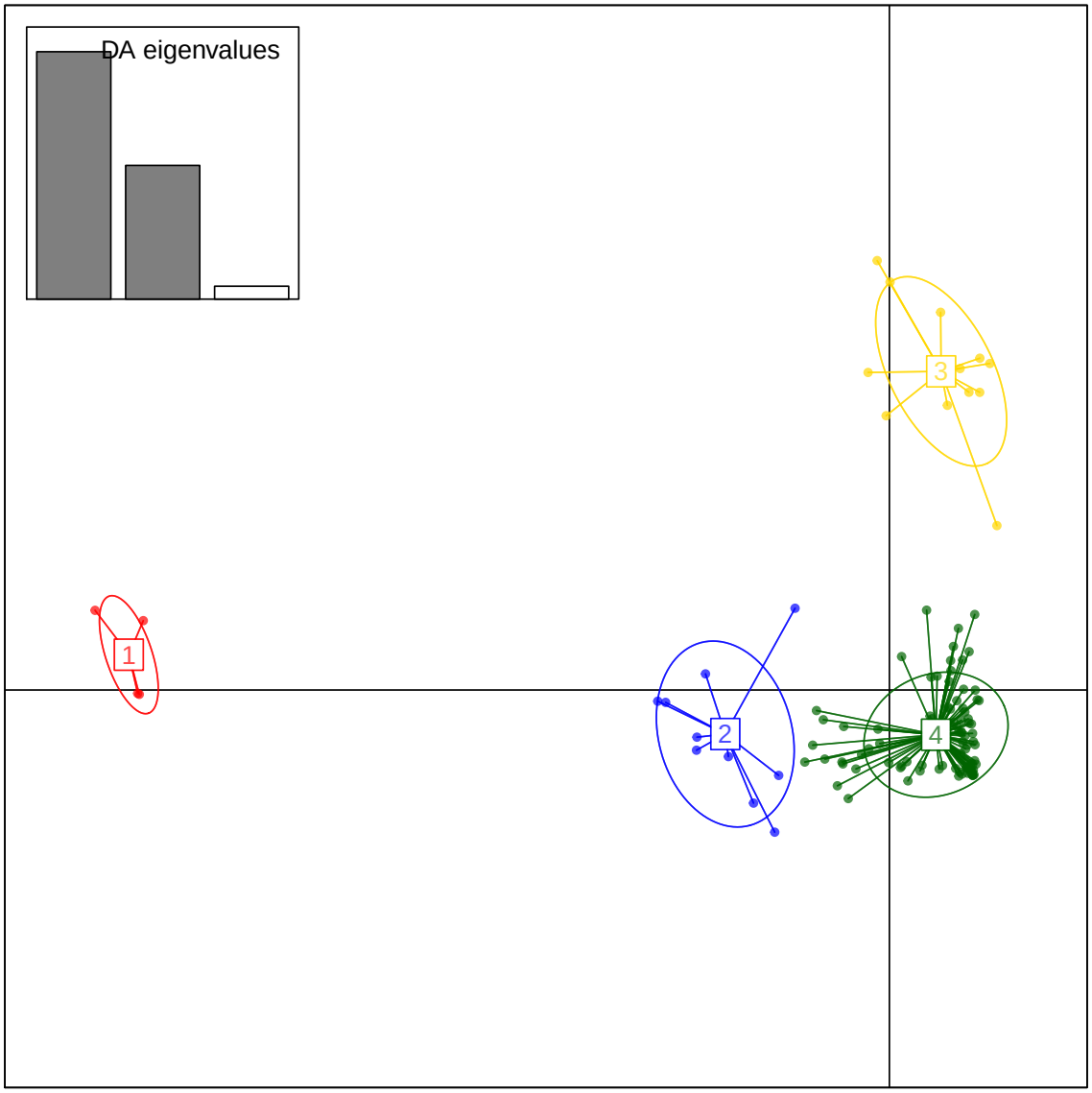
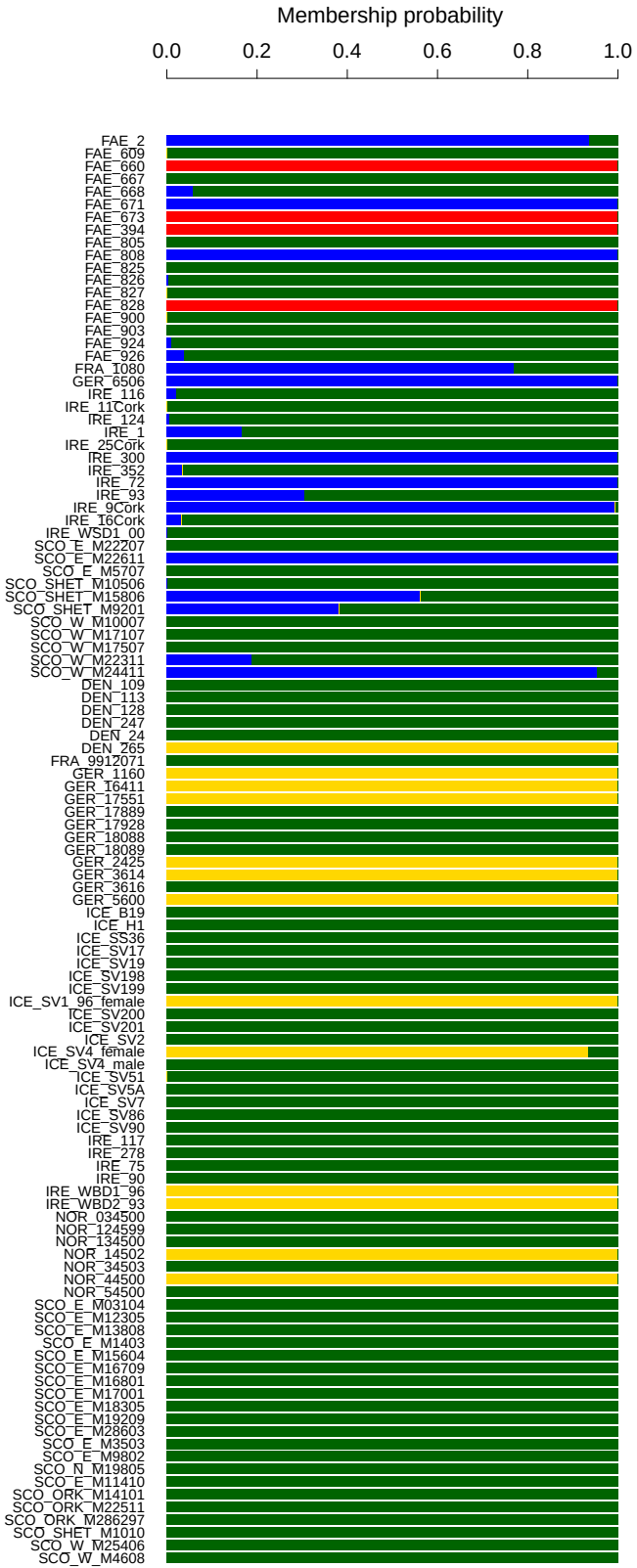


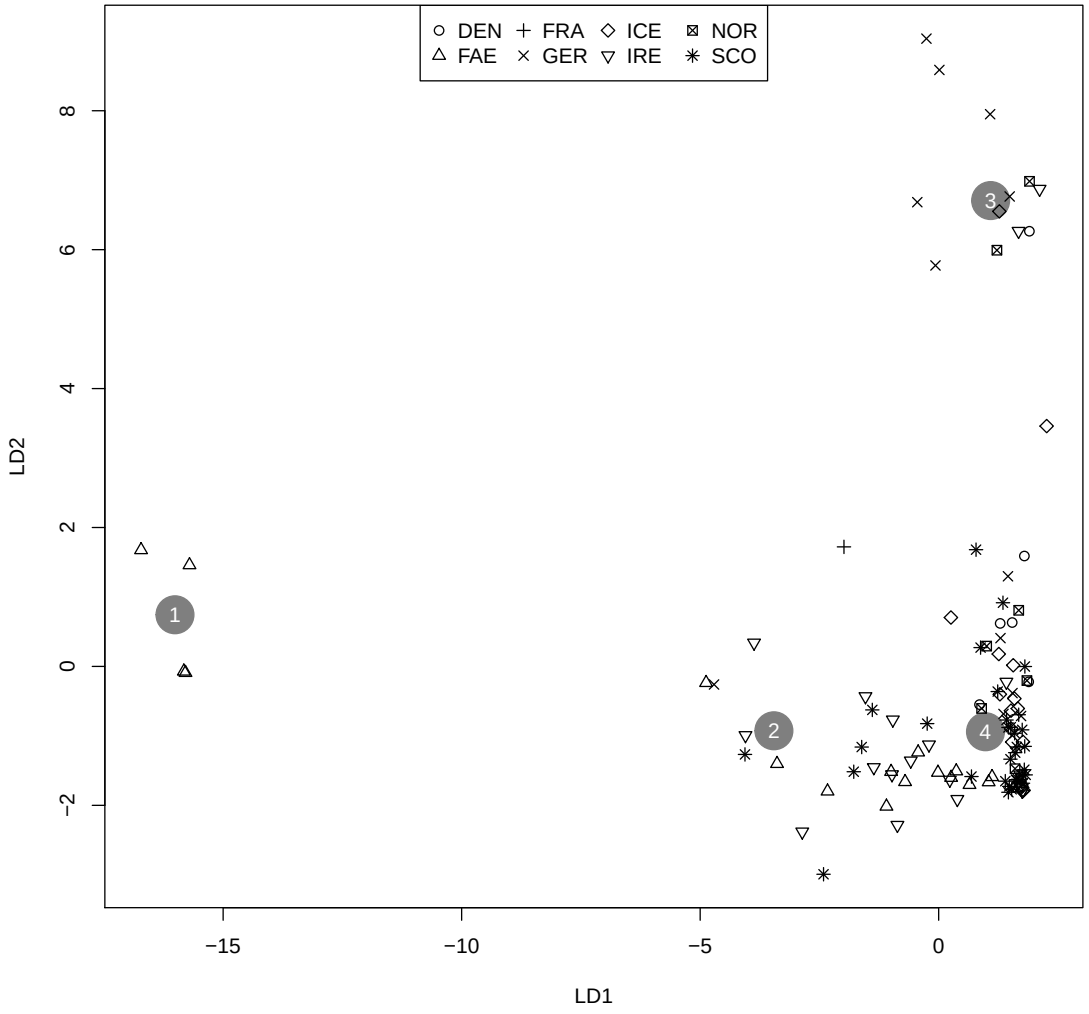
Figure 4:


Membership probability
0.0
0.2
0.4
0.6
0.8
1.0
FAE_2
FAE_609
FAE_660
FAE_667
FAE_668
FAE_671
FAE_673
FAE_394
FAE_805
FAE_808
FAE_825
FAE_826
FAE_827
FAE_828
FAE_900
FAE_903
FAE_924
FAE_926
FRA_1080
GER_6506
IRE_116
IRE_11Cork
IRE_124
IRE_1
IRE_25Cork
IRE_300
IRE_352
IRE_72
IRE_93
IRE_9Cork
IRE_16Cork
IRE_WSD1_00
SCO_E_M22207
SCO_E_M22611
SCO_E_M5707
SCO_SHET_M10506
SCO_SHET_M15806
SCO_SHET_M9201
SCO_W_M10007
SCO_W_M17107
SCO_W_M17507
SCO_W_M22311
SCO_W_M24411
DEN_109
DEN_113
DEN_128
DEN_247
DEN_24
DEN_265
FRA_9912071
GER_1160
GER_16411
GER_17551
GER_17889
GER_17928
GER_18088
GER_18089
GER_2425
GER_3614
GER_3616
GER_5600
ICE_B19
ICE_H1
ICE_SS36
ICE_SV17
ICE_SV19
ICE_SV198
ICE_SV199
ICE_SV1_96_female
ICE_SV200
ICE_SV201
ICE_SV2
ICE_SV4_female
ICE_SV4_male
ICE_SV51
ICE_SV5A
ICE_SV7
ICE_SV86
ICE_SV90
IRE_117
IRE_278
IRE_75
IRE_90
IRE_WBD1_96
IRE_WBD2_93
NOR_034500
NOR_124599
NOR_134500
NOR_14502
NOR_34503
NOR_44500
NOR_54500
SCO_E_M03104
SCO_E_M12305
SCO_E_M13808
SCO_E_M1403
SCO_E_M15604
SCO_E_M16709
SCO_E_M16801
SCO_E_M17001
SCO_E_M18305
SCO_E_M19209
SCO_E_M28603
SCO_E_M3503
SCO_E_M9802
SCO_N_M19805
SCO_E_M11410
SCO_ORK_M14101
SCO_ORK_M22511
SCO_ORK_M286297
SCO_SHET_M1010
SCO_W_M25406
SCO_W_M4608

Supplementary Information

R code for custom plot of the results from the DAPC on the dolphin data. This example shows how to extract the coordinates of the individuals and plot them with a symbol representing the original populations (Fig. S1):

```
plot(d2b$ind.coord, type = "n")
points(d2b$grp.coord, pch = 19, col = "grey50", cex = 4)
text(d2b$grp.coord, labels = 1:4, col = "white")
points(d2b$ind.coord, pch = (1:8)[pop])
legend("top", legend = levels(pop), pch = 1:8, ncol = 4)
```



573 Fig. S1: Results of DAPC with the dolphin data. The group means are shown with white numbers on grey.