Received Date : 04-Apr-2016

Revised Date : 26-May-2016

Accepted Date : 31-May-2016

Article type : Special Issue

Corresponding Author Email ID: **eric.archer@noaa.gov**

strataG: An R package for manipulating, summarizing, and analyzing population genetic data

Frederick I. Archer[1], Paula E. Adams[2], Brita B. Schneiders[3]

[1]Southwest Fisheries Science Center, 8901 La Jolla Shores Drive, La Jolla, CA 92037

[2]University of Alabama, Box 870344, Tuscaloosa, AL 35487

[3]Northwestern University, 2145 Sheridan Road, Evanston, IL 60208

Abstract

We introduce the R package *strataG* as a user-friendly population genetics toolkit. *strataG* provides easy access to a suite of standard genetic summaries as well as the ability to rapidly manipulate stratified genetic data for custom analyses. Tests of population subdivision with most common measures of population subdivision (e.g., F_{ST}, G_{ST}, ϕ_{ST}, χ^2) can be conducted within a single function. The package also provides wrapper functions that allow users to configure and run popular external

programs such as Genepop, STRUCTURE, and fastsimcoal from within R, and smoothly interface with popular R packages *adegenet*, and *pegas*. *strataG* is intended to be an open-source dynamic package that will grow with future needs and user input.

Introduction

The R programming language (R Core Team 2015) has rapidly become a popular platform for analyses of genetic data. Multiple packages have been developed to efficiently store and manipulate genetic data (Jombart *et al.* In Prep; Paradis *et al.* 2004) summarize genetic diversity (Goudet 2005; Jombart 2008; Kamvar *et al.* 2014; Keenan *et al.* 2013; Paradis 2010), and conduct phylogenetic analyses (Paradis *et al.* 2004; Schliep 2011). Still, many population genetics studies that have relatively standard workflows use a mixed array of software and require the reformatting of data or results multiple times in order to move among programs.

In order to alleviate this movement among software platforms and help users create custom analytical workflows within the R environment, we present the *strataG* package, which is designed to be an extensible toolkit for population genetics analyses. With *strataG*, users can easily compile suites of standard genetic summary statistics (e.g., allele frequencies, heterozygosity, proportion of unique alleles, number of private alleles) and conduct common analyses of population structure (e.g., F_{ST}, G_{ST}, ϕ_{ST}, χ^2) with the flexibility of testing multiple stratification schemes. The package also includes wrapper functions to run several popular external analytical programs such as Genepop (Raymond & Rousset 1995) or STRUCTURE (Pritchard *et al.* 2000). The results of many of these external programs are automatically read back into the R environment to facilitate downstream analyses and visualization.

Data input and manipulation

Most of the functions in *strataG* operate on genetic data stored within a `gtypes` object, which is structured as an S4 class with slots for genotypes, stratification

schemes, optional sequences, a description of the data, and optional ancillary information. A `gtypes` object can be created from data originating from a number of standard R data structures, the most common of which is a table-like object (an R `matrix` or `data.frame`) organized where each row is an individual and the columns are its genotypes. Haplotype IDs from multiple DNA loci can also be stored along with their respective sequences.

strataG has also been designed to work with other popular genetics packages. As such, functions (`gtypes2genind`, `gtypes2loci`, and `gtypes2phyDat`) are available to convert between `gtypes` and `genind` objects for the *adegenet* package (Jombart 2008), `loci` objects for the *pegas* package (Paradis 2010), or `phyDat` objects for the *phangorn* package (Schliep 2011). Additionally, functions are available to help convert and prepare data for loading, such as splitting alleles of a locus that are concatenated into a single column, or translating a table of haplotype frequencies to a conformable table of haplotype assignment for individuals.

Within a `gtypes` object, genotypes are stored as an R `data.frame` of factors which makes memory management and calculation of allele frequencies more efficient. DNA sequences are also efficiently stored as a `multidna` object from the `apex` package (Jombart et al. In Prep), simplifying analyses of multiple haploid loci from the same object. A suite of accessor functions is available for `gtypes` objects to extract basic information such as the number of individuals, the current stratification scheme, or the set of associated sequences. Subsetting or indexing of `gtypes` for a specified set of individuals, loci, or strata, uses standard R syntax as in this example based on bottlenose dolphin (*Tursiops truncatus*) microsattelite genotypes as presented in Lowther-Thieleking *et al.* (2015).

```
> # An example microsattelite `gtypes` object
> data(msats.g)
> msats.g


<<< dolphin msats >>>

```

```
Contents: 126 samples, 5 loci, 3 strata

Strata summary:
               num.samples num.missing num.alleles prop.unique.alleles
Coastal                 68         1.2         4.8              0.0857
Offshore.North          40         0.8        12.6              0.2240
Offshore.South          18         0.0        11.0              0.2510
               heterozygosity
Coastal                 0.631
Offshore.North          0.790
Offshore.South          0.867

Locus summary:
      num.genotyped num.alleles prop.unique.alleles obsvd.heterozygosity
D11t           125          12              0.2500                 0.704
EV37           119          22              0.1364                 0.697
EV94           125          15              0.0667                 0.776
Ttr11          125           9              0.2222                 0.704
Ttr34          126          10              0.2000                 0.698

> # Extract the first two loci from the Coastal population
> msats.g[, 1:2, "Coastal"]

<<< dolphin msats >>>

Contents: 68 samples, 2 loci, 1 stratum

Strata summary:
        num.samples num.missing num.alleles prop.unique.alleles
Coastal          68           3           5               0.214
        heterozygosity
Coastal          0.571

Locus summary:
     num.genotyped num.alleles prop.unique.alleles obsvd.heterozygosity
D11t            67           3               0.000                 0.522
EV37            63           7               0.429                 0.619
```

A gtypes object can also contain alternative stratification schemes for individuals, stored as an R data.frame. Each column is a unique stratification scheme, with individuals in the rows being assigned to a stratum within that scheme. Individuals can be excluded from a stratification within a scheme by assigning them the value NA. Stratification schemes can then be easily changed for different analyses using the stratify function. For instance, in the following example, the object is restratified according to the broad scheme which is a stratification column in the schemes slot of msats.g.

```
> # Restratify based on "broad" scheme
> msats.broad <- stratify(msats.g, "broad")
> msats.broad

<<< dolphin msats >>>

Contents: 126 samples, 5 loci, 2 strata

Strata summary:
         num.samples num.missing num.alleles prop.unique.alleles
Coastal           68         1.2         4.8              0.0857
Offshore          58         0.8        13.6              0.1751
         heterozygosity
Coastal           0.631
Offshore          0.814

Locus summary:
     num.genotyped num.alleles prop.unique.alleles obsvd.heterozygosity
D11t           125          12              0.2500                 0.704
EV37           119          22              0.1364                 0.697
EV94           125          15              0.0667                 0.776
Ttr11          125           9              0.2222                 0.704
Ttr34          126          10              0.2000                 0.698
```

Using standard R functions such as `sapply` or `lapply`, the same analyses can be conducted for a suite of stratification schemes with the results for all combined into single object for further processing or summary.

Summaries

The standard display of a `gtypes` object provides information about the size and contents of the object along with many commonly used summary statistics (e.g., the number of alleles, observed and expected heterozygosity, allelic richness) for each stratum as well as each locus, and differ for genotype and sequence data. If saved to an R object, the result of the `summary` function contains information about haplotype or allele frequencies. Each of these summary statistics is also available as individual functions if only certain ones are desired for a particular application. A pre-defined set of by-locus summaries are available from the `summarizeLoci` function.

The package also includes summary functions for sequence data, stored either as a `gtypes`, `DNAbin` (Paradis *et al.* 2004) or `multidna` (Jombart et al. In Prep) object. These include functions for calculating transition / transversion ratios (Ti/Tv), identifying fixed and variable sites, and calculating estimates of selective pressure such as Tajima's D and Fu's F_s (Tajima 1989; Fu 1997). A set of sequence-specific summaries, such as length distributions and base frequencies are available from the `summarizeSeqs` function.

```
> # Reading a fasta file of aligned mitochondrial control region
sequences from Lowther et al (2012) to a DNAbin object
> fname <- system.file("extdata/dolph.seqs.fasta", package = "strataG")
> x <- read.fasta(fname) # one can also use the function
read.dna(fname, type = "fasta") in the ape package
> x

126 DNA sequences in binary format stored in a list.

All sequences of same length: 402

```

```
Labels: 4495 4496 4498 5814 5815 5816 ...

Base composition:
    a     c     g     t
0.301 0.229 0.129 0.341

> # Summarize sequences
> head(summarizeSeqs(x))
     start end length num.ns num.indels
4495     1 402    402      0          2
4496     1 402    402      0          2
4498     1 402    402      0          1
5814     1 402    402      0          2
5815     1 402    402      0          2
5816     1 402    402      0          2

> # Calculate transition/transversion ratio
> TiTvRatio(x)
        Ti          Tv Ti.Tv.ratio
      41.0         4.0        10.2

> # Estimate Tajima's D test of selective neutrality and p-value
> tajimasD(x)
          D p.value
gene.1 -0.506   0.328

> # For comparison, here is Fu's Fs statistic
> fusFs(x)

gene.1
 -7.61
```

Quality control checks

In order to facilitate the use of routine error checks and quality control analyses as part of all analytical workflows, we have provided a suite of functions for quality assurance / quality control (QA/QC) checks. As an example, there is a function that will identify potential duplicate genotypes by reporting all individuals that share genotypes across a specified number or fraction of loci (`dupGenotypes`). The `jackHWE` function will identify homozygotes that occur at an unusually low frequency, implementing the Hardy-Weinberg (HW) jackknife procedure described in Morin *et al.* (2010). The result of this function identifies all individuals that, when removed from the data, will cause a locus that was previously out of HW equilibrium (HWE) to be in HWE. Most by-individual and by-locus summaries and QA/QC checks have also been bundled into a single function (`qaqc`) that conducts all tests and will optionally write the resulting summaries to comma-delimited (.csv) text files, as illustrated in the following example:

```
> # Example of checks/summaries of microsatellite data:
> checks <- qaqc(msats.g)


2016-05-25 13:14:26 : Individual summaries
2016-05-25 13:14:27 : Locus summaries
2016-05-25 13:14:27 : Duplicate genotypes
2016-05-25 13:14:30 : Writing files

> #   By-sample summaries
> head(checks$by.sample)

  sample num.loci.missing.genotypes pct.loci.missing.genotypes
1   4495                          2                         0.4
2   4496                          2                         0.4
3   4498                          0                         0.0
4   5814                          0                         0.0
5   5815                          0                         0.0
6   5816                          0                         0.0
  pct.loci.homozygous
```

```
1              0.667
2              0.333
3              0.600
4              0.200
5              0.600
6              0.000

> #   By-locus summaries for Coastal stratum
> head(checks$by.locus$Coastal)

      locus num.genotyped prop.genotyped num.alleles allelic.richness
D11t   D11t            67          0.985           3           0.0448
EV37   EV37            63          0.926           7           0.1111
EV94   EV94            68          1.000           5           0.0735
Ttr11 Ttr11            68          1.000           4           0.0588
Ttr34 Ttr34            68          1.000           5           0.0735
      prop.unique.alleles exptd.heterozygosity obsvd.heterozygosity
D11t                0.000                0.491                0.522
EV37                0.429                0.608                0.619
EV94                0.000                0.770                0.735
Ttr11               0.000                0.662                0.632
Ttr34               0.000                0.688                0.647

> #   Duplicate checks
> head(checks$dup.df)

  ids.1 ids.2 strata.1 strata.2 num.loci.genotyped num.loci.shared
1 41579 45237  Coastal  Coastal                 4              4
2 23945 78065  Coastal  Coastal                 5              4
3 25503 78053  Coastal  Coastal                 5              4
4 25509 41822  Coastal  Coastal                 5              4
5 41540 78040  Coastal  Coastal                 5              4
6 41578 45233  Coastal  Coastal                 5              4
  prop.loci.shared mismatch.loci
1              1.0
2              0.8          EV37
```

```
3               0.8         Ttr11
4               0.8         Ttr34
5               0.8          D11t
6               0.8          EV94
```

Population Structure

At the heart of *strataG* are the tests of population structure. Included in the package are functions to calculate F_{ST}, F'_{ST}, G_{ST}, G'_{ST}, G''_{ST}, θ_{ST}, χ^2, and Jost's D. Each function takes a `gtypes` object, and returns the test statistic based on the current stratification of samples as well as the permutation test p-value, and optionally the null distribution of the statistic from the random permutations. Each statistic can be run independently or multiple statistics can be run at once with either the `overallTest` or `pairwiseTest` functions. The former performs a "global" test of population differentiation, while the latter performs tests across all pairs of strata. In both `overallTest` and `pairwiseTest`, individual statistics can be specified, or all statistics appropriate to the data will be run. The `pairwiseTest` function produces a single `data.frame` of all results as well as individual pairwise matrices for each test statistic.

```
> # Fst test of overall structure
> statFst(msats.g, nrep = 1000)

$stat.name
[1] "Fst"

$result
estimate    p.val
0.111807 0.000999

$null.dist
NULL

> # Pairwise test of four measures
> pws <- pairwiseTest(msats.g, stats = c("Fst", "Chi2", "Gst", "D"), nrep =
```

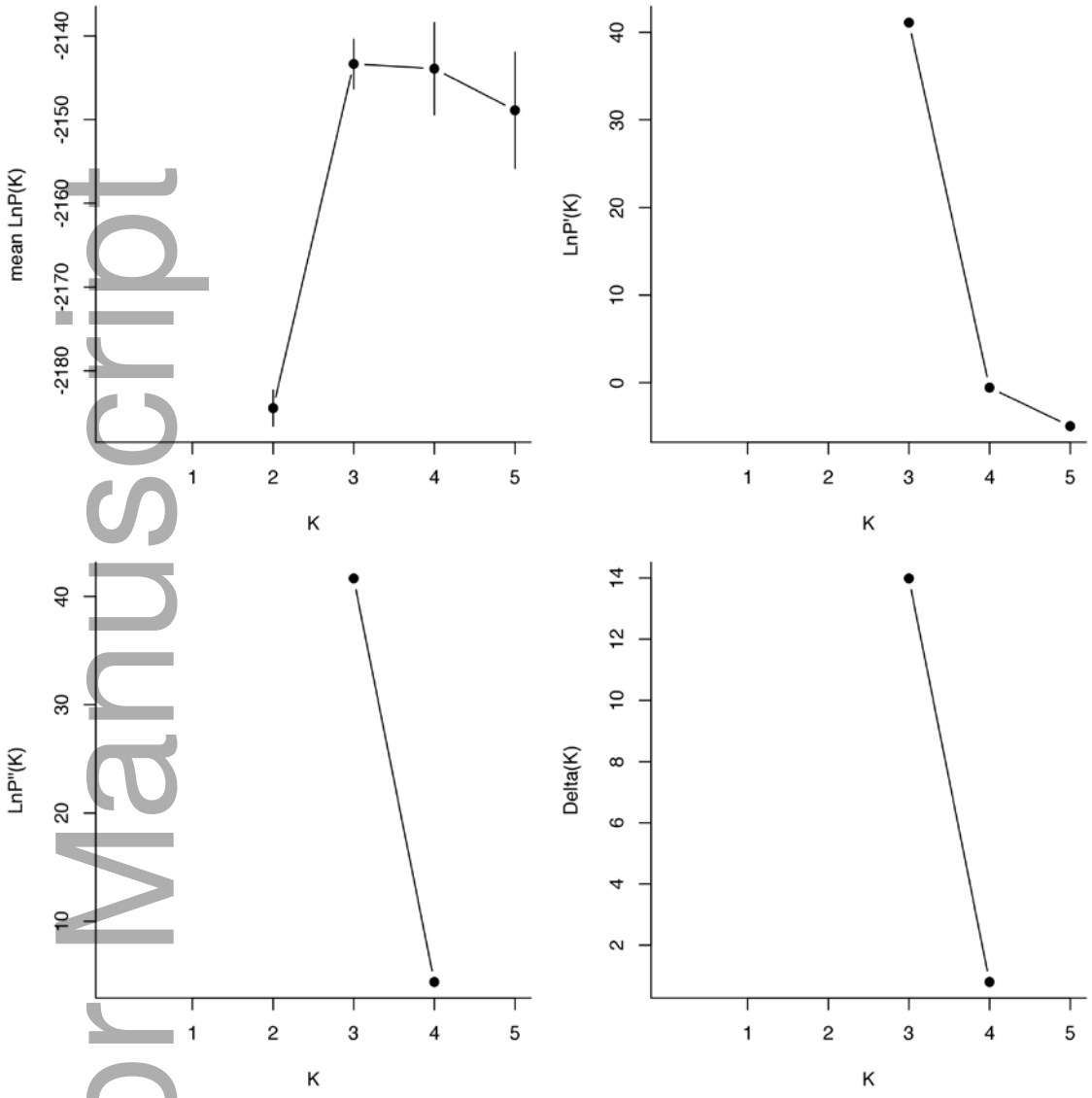
```
1000, quietly = TRUE)
> print(pws$result)

                                  pair.label       strata.1       strata.2
1        Coastal (68) v. Offshore.North (40)        Coastal Offshore.North
2        Coastal (68) v. Offshore.South (18)        Coastal Offshore.South
3 Offshore.North (40) v. Offshore.South (18) Offshore.North Offshore.South
  n.1 n.2      Fst Fst.p.val  Chi2 Chi2.p.val     Gst Gst.p.val       D
1  68  40  0.13064  0.000999 476.8   0.000999  0.0626  0.000999 0.37171
2  68  18  0.14641  0.000999 438.9   0.000999  0.0643  0.000999 0.41159
3  40  18 -0.00417  0.802198  63.9   0.576424 -0.0126  0.758242 0.00597
  D.p.val
1 0.00105
2 0.00108
3 0.66152
```

External Software

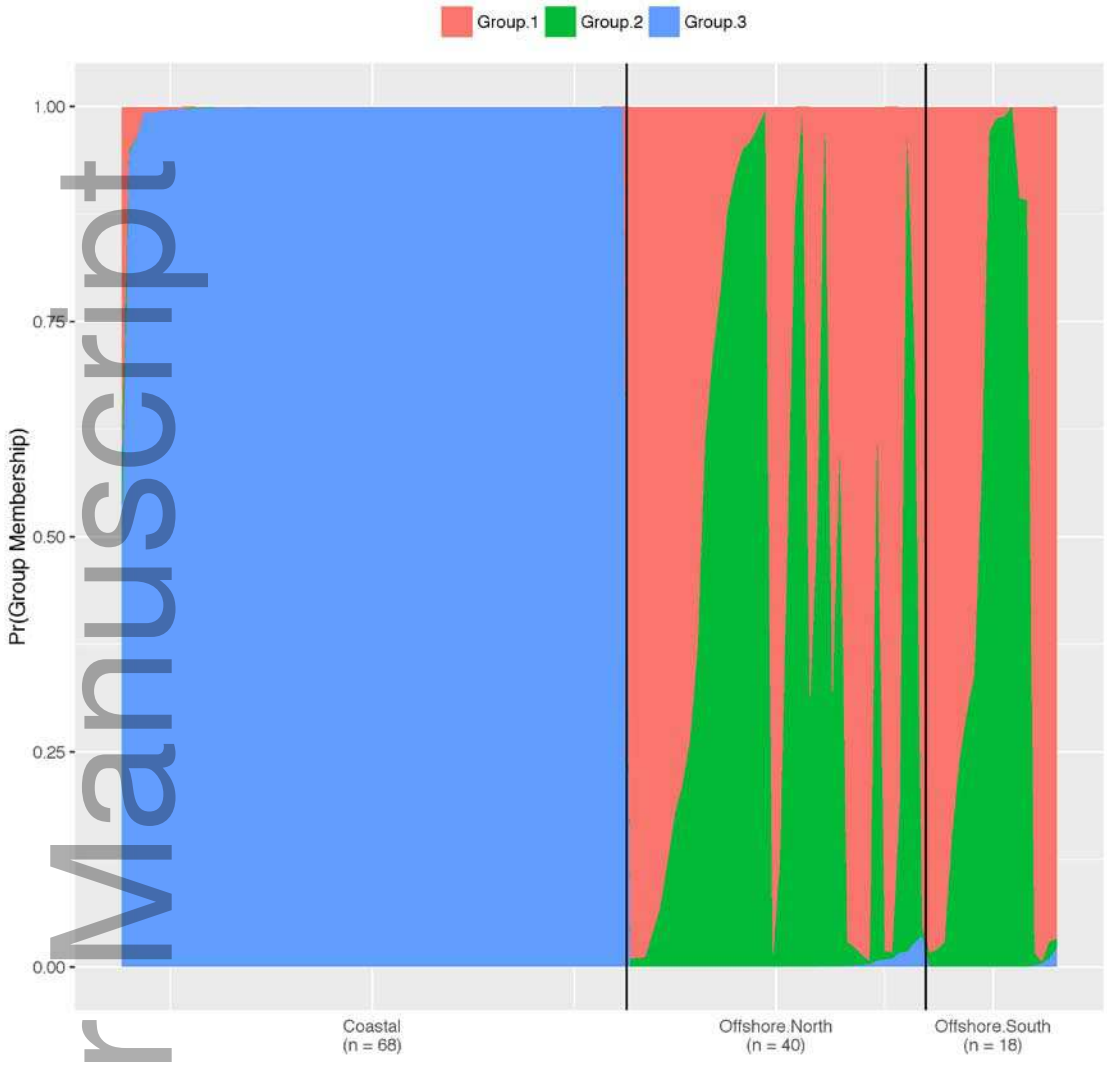
strataG provides wrapper functions for several popular population genetics programs. As of this writing, functions are available for Genepop (Raymond & Rousset 1995), the Bayesian clustering program STRUCTURE (Pritchard *et al.* 2000), the DNA sequence alignment program MAFFT (Katoh & Standley 2013), the coalescent simulator fastsimcoal (Excoffier & Foll 2011), and PHASE (Stephens & Donnelly 2003) for identifying haplotypes of linked loci. The wrappers are designed to facilitate use of these programs within the R environment, especially in cases where R-coded packages with the same functionality are not available. The wrappers are composed of functions to format and write input data and parameter files, execute the programs on the command line with options specified as arguments to the functions, and then in most cases, parse the output from the programs and return results to the user as R objects. These programs are not distributed with the *strataG* package and must be downloaded and installed separately. *strataG* assumes that the programs are installed such that they are executable on the command line from the working directory. This usually means installing them into a folder within the system path environmental variable. More detailed instructions are available in a vignette in the package.

Below we demonstrate an example run of STRUCTURE on the bottlenose dolphin microsattelite data showing differentiation between Coastal and Offshore populations, but none between Offshore.North and Offshore.South (Lowther-Thieleking *et al.* 2015). The `structureRun` function formats the microsatellite data, writes input files, runs the external executable for STRUCTURE, then reads and parses and reads the output files into an R list structure. The `evanno` function displays diagnostic plots of number of groups (K) and first and second-order changes in LnP(K) as described in Evanno et al (2005) using base R graphics. The `clumpp` function aggregates STRUCTURE runs for a single value of K and is visualized with the `structurePlot` function which uses the `ggplot2` graphics package (Wickham 2009).

```
> # Run STRUCTURE for k = 2 to 5
> msats.struct <- structureRun(msats.g, k = 2:5, num.k.rep = 100)
>
> # Display diagnostic plots and table from Evanno et al (2005).
> evanno(msats.struct)
```




```
  k reps mean.ln.k sd.ln.k  ln.pk ln.ppk delta.k
1 2  100     -2184    2.20     NA     NA      NA
2 3  100     -2143    3.57 40.545  39.65   11.10
3 4  100     -2142    5.55  0.898   8.95    1.61
4 5  100     -2151    6.66 -8.053     NA      NA

> # Run CLUMPP to aggregate the results of k = 3
> msats.clumpp <- clumpp(msats.struct, k = 3)

> # Show distribution of group membership probabilities by strata
> sp <- structurePlot(msats.clumpp, horiz = FALSE)
```



Performance

Where possible, *strataG* has been designed for optimal performance in terms of computational speed and memory management, while still retaining ease of code maintenance. The core algorithms of the computationally-intensive population structure tests have been written in C and integrated using the `Rcpp` package (Eddelbuettel 2013). Additionally, where useful, code has been written to take advantage of multiple CPUs using functions in the `parallel` package that is distributed as part of base R. The functions will automatically detect the user's

operating system and set up the clusters in the appropriate format. Users only need to specify the number of cores to use as a function argument.

Although it is difficult to conduct performance benchmark tests that would cover every dataset and analysis, as a simple example, we generated a simulated microsatellite dataset of 300 samples, 100 from each of three populations to demonstrate computational times in *strataG* (Supplemental Materials) Using the `fastsimcoal` interface in *strataG*, we simulated 1000 loci, then randomly subsampled for 50, 100, and 500 loci. Mutation rates were randomly chosen for each loci from a uniform distribution from 10^{-7} to 10^{-4} mutations per generation, which produced from one to 15 alleles per locus, with a mean of seven. On a MacBook Pro with a 2.8GHz Intel Core i7 CPU and 16GB of 1600MHz RAM, the average execution time for calculating overall F_{ST} and conducting 1000 permutation replicates to estimate a p-value on each subset as well as all 1000 loci was 0.2s, 0.4s, 2s, and 4s respectively. This indicates a simple linear increase in time with the number of loci. For most population genetics datasets, processing should be relatively rapid on standard personal systems or servers.

Obtaining *strataG*

The current stable release of *strataG* (version 1.0.5 as of this writing) is available for download from the Comprehensive R Archive Network (CRAN) at [https://cran.r-project.org]. Pre-release versions with recent bug fixes and additions are available through GitHub at [https://github.com/EricArcher/strataG], which also has instructions on how to install the package using the `install_github` function available in the `devtools` package.

strataG is actively maintained, and contains many other additional functions. Users are encouraged to explore the package starting with the list provided by `help(package = "strataG")`. The package includes vignettes covering how to create `gtypes`, summarize data, conduct population differentiation tests, and install and run external programs. With the rapid spread of next generation sequencing and the growth of population genomics, we expect an increasing demand to store

and process ever larger datasets and the development of novel analytical methods. Because the `gtypes` object is a strongly-typed S4 object with accessor functions for the data, the underlying structures can easily be modified to take advantage of more efficient storage methods without necessitating changes to existing code. Given the open source nature of the R programming environment and the growing popularity of collaborative development platforms such as GitHub, we envision *strataG* to be a dynamic toolkit that can grow with the addition of new methods and community needs. We invite the population genetics community to actively participate in its development with suggestions for improvements and contributed code.

Acknowledgements

The authors wish to thank the members of the Marine Mammal Genetics Group at the Southwest Fisheries Science Center and the many other users who have helped in the development of *strataG* through their feedback on early versions of the package. We would also like to thank the National Evolutionary Synthesis Center (NESCent) for organizing 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. FIA was a participant in the hackathon, and is indebted to NESCent (NSF #EF-0905606) for hosting and supporting the event.

Literature Cited

Eddelbuettel D (2013) *Seamless R and C++ Integration with Rcpp.* Springer.

Excoffier L, Foll M (2011) Fastsimcoal: a continuous-time coalescent simulator of genomic diversity under arbitrarily complex evolutionary scenarios. *Bioinformatics* **27**, 1332-1334.

Fu Y-X (1997) Statistical tests of neutrality of mutations against population growth, hitchiking and background selection. *Genetics* **147**,915-925.

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

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

Jombart T, Schliep KP, Archer FI*, et al.* (In Prep) apex: phylogenetics with multiple genets. *Molecular Ecology Resources*.

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.

Katoh K, Standley DM (2013) MAFFT Multiple Sequence Alignment Software Version 7: Improvements in Performance and Usability. *Molecular Biology and Evolution* **30**, 772-780.

Keenan K, McGinnity P, Cross TF, Crozier WW, Prodöhl P (2013) diveRsity: An R package for the estimation and exploration of population genetics parameters and their associated errors. *Methods in Ecology and Evoluton* **4**, 782-788.

Lowther-Thieleking JL, Archer FI, Lang AR, Weller DW (2015) Genetic differentiation among coastal and offshore common bottlenose dolphins, *Tursiops truncatus*, in the eastern North Pacific Ocean. *Marine Mammal Science* **31**, 1-20.

Morin PA, Archer FI, Foote AD*, et al.* (2010) Complete mitochondrial genome phylogeographic analysis of killer whales (*Orcinus orca*) indicates multiple species. *Genome Research* **20**, 908-916.

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

Paradis E, Claude J, Strimmer K (2004) APE: Analyses of Phylogenetics and Evolution in R language. *Bioinformatics* **20**, 289-290.

Pritchard JK, Stephens M, Donnelly P (2000) Inference of population structure using multilocus genotype data. *Genetics* **155**, 945-959.

R Core Team (2015) R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing.

Raymond M, Rousset F (1995) GENEPOP (Version 1.2): Population genetics software for exact tests and ecumenicism. *Heredity* **86**, 248-249.

Schliep KP (2011) phangorn: phylogenetic analysis in R. *Bioinformatics* **27**, 592-593.

Stephens M, Donnelly P (2003) A comparison of Bayesian methods for haplotype reconstruction from population genotype data. *American Journal of Human Genetics* **73**, 1162-1169.

Tajima F (1989) Statistical method for testing the neutral mutation hypothesis by DNA polymorphism. *Genetics* **123**,585-595.

Wickham H (2009) *ggplot2: Elegant Graphics for Data Analysis*. Springer.