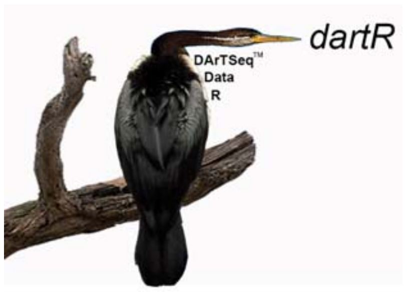
Wildlife Genomics@UC
wildlifegenomics.biomatix.org

UNIVERSITY OF CANBERRA

SNP Analysis using dartR



Data input from sources other than DArT

Version 2

Copies of the latest version of this tutorial are available from:

The Institute for Applied Ecology
University of Canberra ACT 2601
Australia

Email: georges@aerg.canberra.edu.au

dartR is a collaboration between the University of Canberra, CSIRO and Diversity Arrays Technology and is supported with funding from the ACT Priority Investment Program, CSIRO and the University of Canberra.

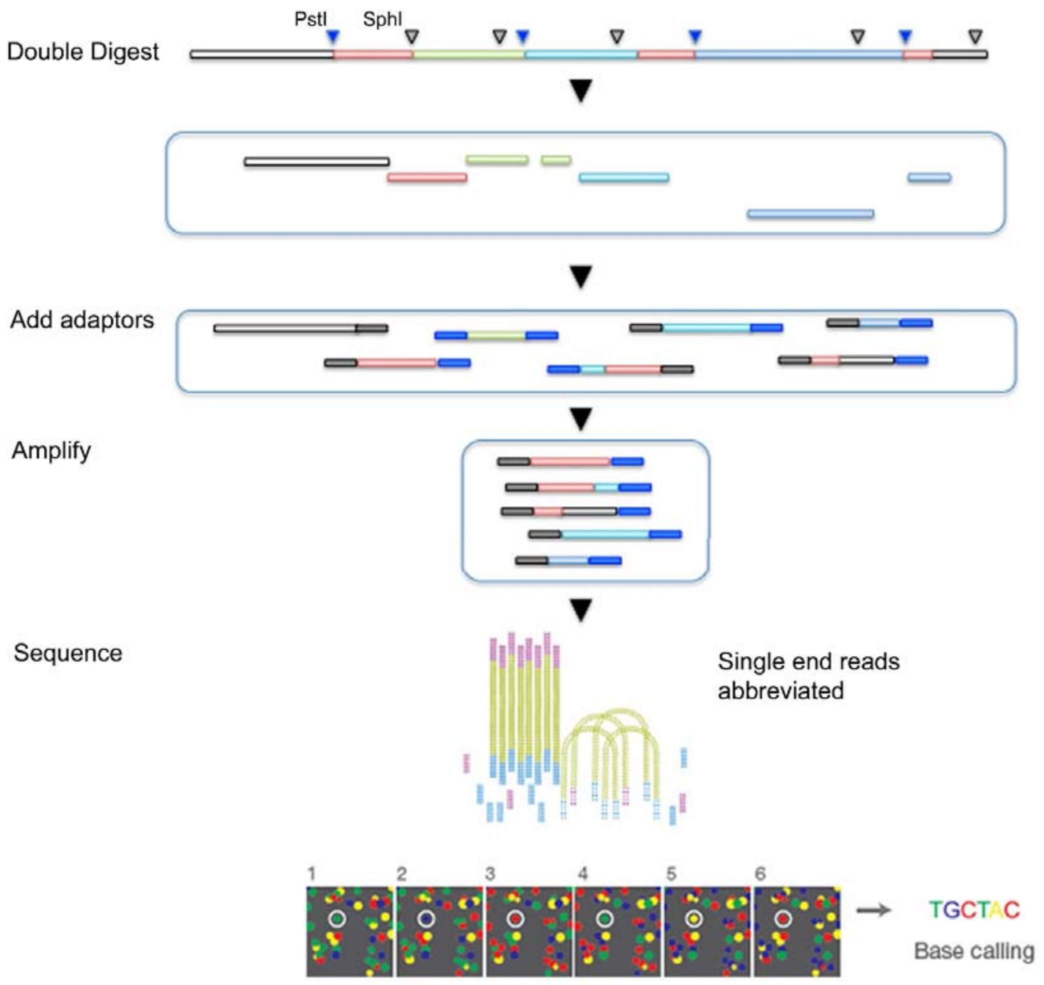
Contents

Session 1: Introduction to ddRAD

Sequencing

Double digest restriction associated DNA (ddRAD) is a method that extracts reproducible genomic variation across the genomes of many individuals at an affordable cost. The technique digests genomic DNA using pairs of restriction enzymes (cutters). When the DNA is cut at two locations within a reasonable distance of each other, the fragment is available for sequencing using the Illumina short-read platforms. Hence, the data are representational in the sense that they are generated for a random selection of small fragments of sequence only, fragments that exhibit variation at the level of single base pairs (SNPs).

The first step in the process involves the selection of restriction enzymes that provide the best balance between getting an adequate fraction of the genome represented, an adequate read depth for each fragment, and adequate levels of polymorphism. This is species specific and so requires some initial optimization.

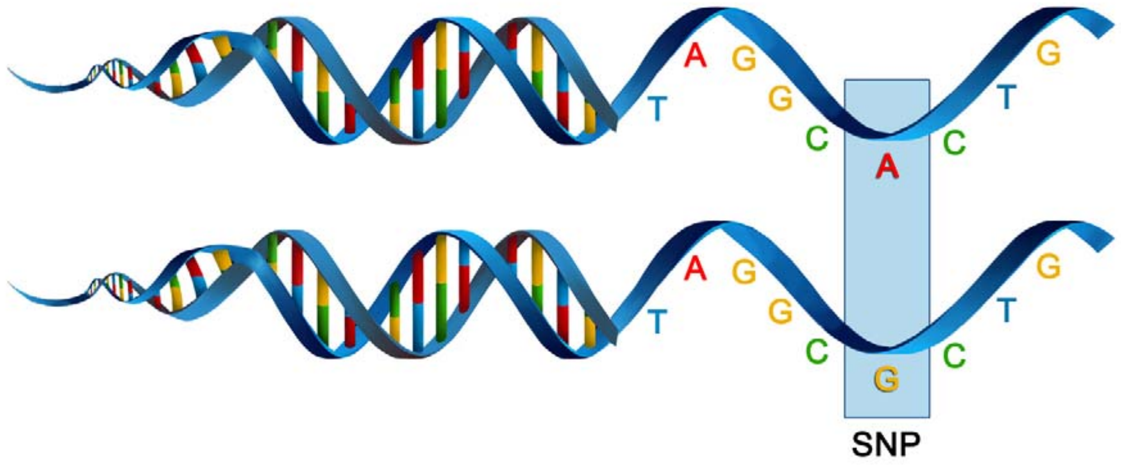


Once the best restriction enzymes are selected, say PstI (recognition sequence 5’-CTGCA|G-3’) and SphI (5’-GCATG|C-3’), then the DNA is digested, and various adaptors added to the sequence fragments to allow Illumina short-read sequencing to proceed. These additional terminal sequences include a barcode to allow disaggregation of the sequences for each sample during later analysis.

The fragments of DNA selected by this process are sequenced in an abbreviated process to yield a set of raw “sequence tags” of length determined by the Illumina platform used for the sequencing (up to 300 bp). The sequence tags are filtered on sequence quality, particularly in the barcode region, truncated based on quality scores and stacked based on sequence similarity. A series of filters are then applied to select those sequence tags that include a reliable SNP marker.

The SNP dataset

SNPs, or single nucleotide polymorphisms, are single base pair mutations at a nuclear locus. That nuclear locus is represented in the dataset by two sequence tags which, at a heterozygous locus, take on two allelic states, one referred to as the reference state, the other as the alternate or SNP state.



Because it is extremely rare for a mutation to occur twice at the same site in the genome (perhaps with the exception of Eucalypts), the SNP data are effectively biallelic.

The data can be represented in a table of SNP bases (A, T, C or G), with two states for each individual at each locus in a diploid organisms.

	Ind 01	Ind 02	Ind 03	Ind 04	Ind 05	Ind 06	Ind 07	Ind 08	Ind 09	Ind 10
Locus 1	A/A	A/A	A/A	A/A	A/G	A/A	A/A	A/A	A/A	-/-
Locus 2	C/C	C/C	C/C	C/C	C/C	C/C	C/T	C/C	C/C	C/C
Locus 3	C/G	G/G	G/G	G/G	G/G	C/C	C/C	C/C	C/C	C/C
Locus 4	A/A	A/T	A/A	A/T	T/T	A/A	A/A	A/A	A/A	A/A
Locus 5	A/A	A/A	A/A	A/A	-/-	A/G	A/A	A/A	A/A	A/A
Locus 6	C/C	C/C	C/C	C/C	C/C	C/C	C/T	C/C	C/C	C/C
Locus 7	C/G	G/G	G/G	G/G	G/G	C/C	C/C	C/C	C/C	C/C
Locus 8	A/A	A/T	A/A	A/T	T/T	A/A	A/A	A/A	A/A	A/A
Locus 9	A/A	A/A	A/A	A/A	A/A	A/A	A/A	A/A	A/A	A/A
Locus 10	C/C	C/C	C/C	C/C	C/C	C/C	C/T	C/C	C/C	C/C
Locus 11	C/G	G/G	G/G	G/G	G/G	C/C	C/C	C/C	C/C	C/C

Alternatively, because the data are biallelic, it is computationally convenient to code the data as 0 for homozyogotes for one allele, 1 for heterozygotes, and 2 for homozygotes of the other allele.

The reference allele is often arbitrarily taken to be the most common allele, so 0 is the score for homozygous reference, and 2 is the score for homozygous alternate or SNP state. NA indicates that the SNP could not be scored.

	Ind01	Ind02	Ind03	Ind04	Ind05	Ind06	Ind07	Ind08	Ind09	Ind10
Locus 1	0	0	0	0	1	0	0	0	0	NA
Locus 2	0	0	0	0	0	0	1	0	0	0
Locus 3	1	2	2	2	2	0	0	0	0	0
Locus 4	0	1	0	1	2	0	0	0	0	0
Locus 5	0	0	0	0	NA	1	0	0	0	0
Locus 6	0	0	0	0	0	0	1	0	0	0
Locus 7	1	2	2	2	2	0	0	0	0	0
Locus 8	0	1	0	1	2	0	0	0	0	0
Locus 9	0	0	0	0	0	0	0	0	0	0
Locus 10	0	0	0	0	0	0	1	0	0	0
Locus 11	1	2	2	2	2	0	0	0	0	0

This is the form the data are stored in dartR, though note that it departs from the coding arrangement used by DArT PL.

Some sequence tags might contain more than one SNP, in which case they are likely to be closely linked when passed from parent to offspring. These may need consideration when preparing your data for analysis. Note that multiple SNPs occurring in the same sequence tag should each be represented as a different data record in the dataset.

Presence-Absence Data

Just as individuals vary in allelic composition at SNP sites, individuals also vary at the restriction sites used to pull the representation from the genome. A mutation at one or both of the restriction sites will result in allelic drop-out or null alleles. The presence or absence of particular sequence tags across individuals provides a source of information additional to the SNP data.

	Ind01	Ind02	Ind03	Ind04	Ind05	Ind06	Ind07	Ind08	Ind09	Ind10
Locus 01	0	1	0	0	0	0	1	1	0	1
Locus 02	1	0	1	1	1	1	1	0	0	1
Locus 03	1	1	0	1	0	0	0	0	1	0
Locus 04	1	1	0	1	1	0	1	0	0	0
Locus 05	0	0	0	0	1	NA	1	0	1	0
Locus 06	1	1	1	0	1	1	1	0	1	1
Locus 07	1	1	NA	0	0	1	1	0	1	0
Locus 08	1	0	1	0	0	1	1	0	1	1
Locus 09	1	1	0	0	0	1	1	1	0	1
Locus 10	0	0	1	1	0	1	1	1	NA	1
Locus 11	0	1	1	0	1	1	0	0	1	1

Note that, unlike the SNP data, NA represents a truly missing value, in that it could not be determined if a sequence tag was present or absent.

Where have we come?

The above Session was designed to give you a very brief overview to the pipelines for producing SNP and associated data. Having completed this Session, you should now be familiar with the following concepts.

- The concept of a SNP marker and how they are generated.

- The distinction between SNP calls and Presence-Absence calls and associated datasets.
- The coding used for SNP genotypes – 0 for homozygous reference, 2 for homozygous alternate, 1 for heterozygous, and NA for 'missing'.
- The coding used for Presence-Absence genotypes – 0 for absent, 1 for present, and NA for missing.

Further reading

Kess, T., Gross, J., arper, F., Boulding, E.G. (2016). Low-cost ddRAD method of SNP discovery and genotyping applied to the periwinkle *Littorina saxatilis*. Journal of Molluscan Studies 82 104–109.

Peterson B.K., Weber J.N., Kay E.H., Fisher H.S., Hoekstra H.E. (2012). Double Digest RADseq: an inexpensive method for de novo SNP discovery and genotyping in model and non-model species. PLoS One, 7: e37135.

Parchman T.L., Gompert Z., Benkman C.W., Schillkey F.D., Mudge J., Buerkle C.A. (2012). Genome wide association mapping of an adaptive trait in lodgepole pine. Molecular Ecology 21:2991–3005.

Session 2: Getting data into dartR

If you are coming back to us, create or load a project, set a working directory, and do not forget to set the default directory for files, the outpath, to getwd(). Refer to the previous Session.

A sensible workflow

Let us begin by jumping the gun and defining a sensible pipeline for entering your data, as a context for the material in this and subsequent Sessions.

1. Examine the data to determine the means by which it will be imported to the dartR package.
2. Prepare the metadata associated with each individual. This dataset, stored in csv format, contains at a minimum the individual/specimen labels in a column headed `id`, and a population column that assigns individuals to groups or populations in a column headed `pop`. Other columns are optional, but might include latitude, longitude of capture, sex, or other possible groupings of the individuals.
3. Prepare the metadata associated with each locus. This dataset, also stored in csv format, contains at a minimum the locus labels in a column headed `CloneID`, and a variable that indicates the position within the sequence tag of the SNP variant. Other columns are optional, but might include ReadDepth, TrimmedSequence and other variables that cannot be imported with the data (say from a vcf file) or be calculated from the data.
4. Read the data into dartR and associate it with the metadata.

We elaborate on this workflow in the sections that follow.

How dartR stores SNP data

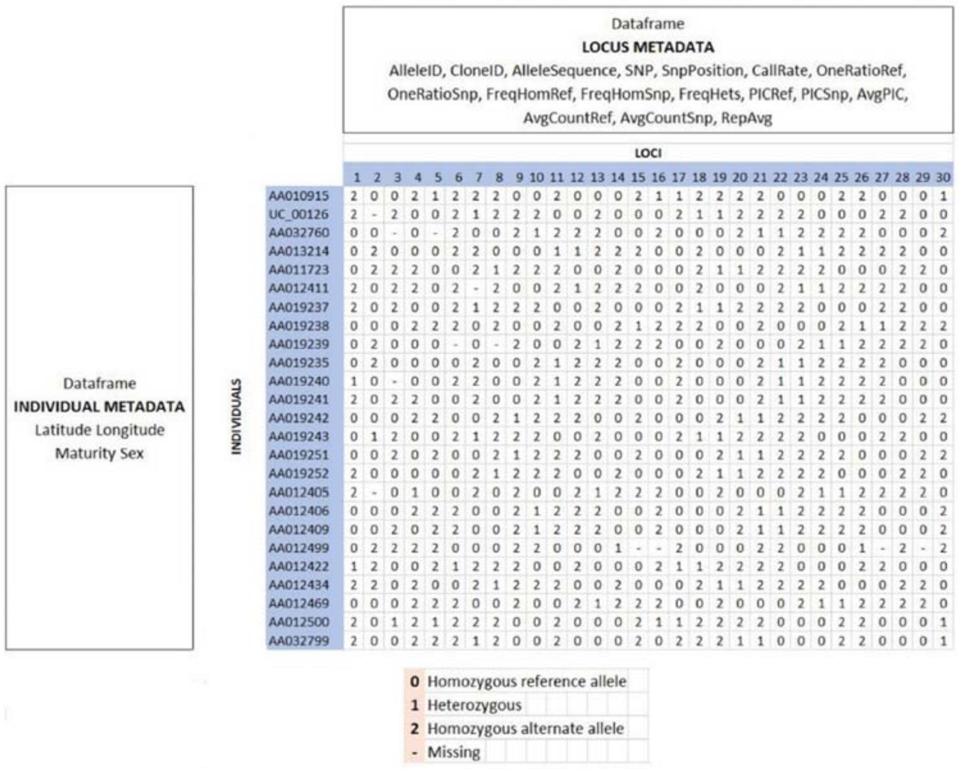
The package dartR relies on the SNP data being stored in a compact form using a bit-level coding scheme. SNP data coded in this way are held in a genlight object that is defined in the R package adegenet (Jombart, 2008; Jombart and Ahmed, 2011). Refer to the tutorial prepared by Jombart and Collinson (2015), called *Analysing genome-wide SNP data using adegenet 2.0.0*, if you require further information.

The complex storage arrangement of genlight objects is hidden from the user because it is accompanied by a number of "accessors". These allow the data to be accessed in a way similar to the manipulation of standard objects in R, such as lists, vectors and matrices.

A genlight object can be considered to be a matrix containing the SNP data encoded in a particular way. The matrix entities (rows) are the individuals, and the attributes (columns) are the SNP loci. In the body of this individual x locus matrix are the SNP data, coded as 0 for homozygous reference state, 1 for heterozygous, and 2 for homozygous alternate (or SNP) state.

Note that a genlight object used by dartR differs in some important respects from the default genlight object of adegenet (a dartR genlight object is a superset of an adegenet genlight object). By this, we mean that all functions in the adegenet package work on dartR genlight objects, but dartR genlight objects have other essential components. So creating a genlight object to hold your data manually from a vcf or csv format requires a few steps in addition to importing the data to an adegenet genlight object, as outlined later in this tutorial.

Genlight objects not only have the SNP data, but also allow for attachment of locus metadata to the loci, and attachment of individual metadata to the individuals/samples. This is represented diagrammatically below.


Dataframe
LOCUS METADATA
AlleleID, CloneID, AlleleSequence, SNP, SnpPosition, CallRate, OneRatioRef, OneRatioSnp, FreqHomRef, FreqHomSnp, FreqHets, PICRef, PICSnp, AvgPIC, AvgCountRef, AvgCountSnp, RepAvg
Dataframe
INDIVIDUAL METADATA
Latitude Longitude
Maturity Sex
INDIVIDUALS
LOCI

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
AA010915	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0	0	0	2	2	0	0	0	1
UC_00126	2	-	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0	0	0	2	2	0	0
AA032760	0	0	-	0	-	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0	0	0	2
AA013214	0	2	0	0	0	2	2	0	0	0	1	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0	0
AA011723	0	2	2	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0	0	0	2	2	0
AA012411	2	0	2	2	0	2	-	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0	0
AA019237	2	0	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0	0	0	2	2	0	0
AA019238	0	0	0	2	2	2	0	2	0	0	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2
AA019239	0	2	0	0	0	-	0	-	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0
AA019235	0	2	0	0	0	0	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0	0	0
AA019240	1	0	-	0	0	2	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0	0	0
AA019241	2	0	2	2	0	0	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0	0	0
AA019242	0	0	0	2	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0	0	0	2	2
AA019243	0	1	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0	0	0	2	2	0	0
AA019251	0	0	2	0	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0	0	0	2	2
AA019252	2	0	0	0	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0	0	0	2	2	0
AA012405	2	-	0	1	0	0	2	0	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0
AA012406	0	0	0	2	2	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0	0	0	2
AA012409	0	0	2	0	2	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0	0	0	2
AA012499	0	2	2	2	2	0	0	0	2	2	0	0	0	1	-	-	2	0	0	0	2	2	0	0	0	1	-	2	-	2
AA012422	1	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0	0	0	2	2	0	0	0
AA012434	2	2	0	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0	0	0	2	2	0
AA012469	0	0	0	2	2	2	0	0	2	0	0	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0
AA012500	2	0	1	2	1	2	2	2	0	0	2	0	0	0	2	1	1	2	2	2	2	0	0	0	2	2	0	0	0	1
AA032799	2	0	0	2	2	2	1	2	0	0	2	0	0	0	2	0	2	2	2	1	1	0	0	0	2	2	0	0	0	1

0	Homozygous reference allele
1	Heterozygous
2	Homozygous alternate allele
-	Missing

Locus metadata

The locus metadata are held in an R data.frame that is associated with the SNP data as part of the genlight object.

The locus metadata would typically include:

SNP	Mutational change and its position in the sequence tag, referenced from zero*
SnpPosition	Position (zero is position 1) in the sequence tag of the defined SNP variant base*
TrimmedSequence	(Optional) The sequence containing the SNP or SNPs (the sequence tag), trimmed of adaptors*
CallRate	Proportion of samples for which the genotype call is non-missing (that is, not "-")
OneRatioRef	Proportion of samples for which the genotype score is 0
OneRatioSnp	Proportion of samples for which the genotype score is 2

]

FreqHomRef	Proportion of samples homozygous for the Reference allele
FreqHomSnp	Proportion of samples homozygous for the Alternate (SNP) allele
FreqHets	Proportion of samples which score as heterozygous, that is, scored as 1
PICRef	Polymorphism information content (PIC) for the Reference allele
PICSnp	Polymorphism information content (PIC) for the SNP
AvgPIC	Average of the polymorphism information content (PIC) of the Reference and SNP alleles
AvgCountRef	Sum of the tag read counts for all samples, divided by the number of samples with non-zero tag read counts, for the Reference allele row
AvgCountSnp	Sum of the tag read counts for all samples, divided by the number of samples with non-zero tag read counts, for the Alternate (SNP) allele row
RepAvg	Proportion of technical replicate assay pairs for which the marker score is consistent*

Those marked with an * cannot be calculated from the data, and must be provided by the user, however they are optional.

In addition, dartR calculates the minor allele frequency and an estimate of read depth, and stores it in the locus metadata.

These metadata variables are held in the genlight object as part of a data.frame called loc.metrics, which can be accessed in the following form:

```
# Make a genlight object to work with
  gl <- testset.gl

# Only entries for the first 10 individuals are shown
  gl@other$loc.metrics$RepAvg[1:10]
```

```
## [1] 1.000000 1.000000 1.000000 1.000000 0.989950 1.000000 0.993274
## [8] 1.000000 1.000000 0.980000
```

You can check the names of all available loc.metrics via:

```
names(gl@other$loc.metrics)
```

```
## [1] "AlleleID" "CloneID" "AlleleSequence" "SNP"
## [5] "SnpPosition" "CallRate" "OneRatioRef" "OneRatioSnp"
## [9] "FreqHomRef" "FreqHomSnp" "FreqHets" "PICRef"
## [13] "PICSnp" "AvgPIC" "AvgCountRef" "AvgCountSnp"
## [17] "RepAvg" "clone" "uid" "rdepth" "maf"
```

Examine the first 10 values of CallRate and some other listed locus metadata in `testset.gl` and your own dataset once you have it imported.

These metadata are used by the dartR package for various purposes, so if any are missing from your dataset, then there will be some analyses that will not be possible. For example, `TrimmedSequence` is used to generate output for subsequent phylogenetic analyses that require estimates of base frequencies and transition and transversion ratios.

`CloneID` is essential (with its very special format), and dartR scripts for loading your data sets will terminate with an error message if this is not present.

Individual metadata

Individual (=specimen/sample) metadata are user specified. Individual metadata are held in a second dataframe associated with the SNP data in the genlight object. See the figure above.

Two special individual metrics are:

`id`	Unique identifier for the individual or specimen that links back to the physical sample
`pop`	A label for the biological population from which the individual was drawn

Individual metrics are supplied by the user by way of an individual metafile, provided at the time of inputting the SNP data to the genlight object. A metafile is a comma-delimited file, usually named ind_metrics.csv or similar, that contains labelled columns. The file must have a column headed `id`, which contains the individual (=specimen or sample labels), and a column headed `pop`, which contains the populations to which individuals are assigned.

These special metrics can be accessed using:

```
pop(gl)
```

```
popNames(gl)
```

```
indNames(gl)
```

Try these for yourself to see the output they produce, for the testset.gl and for your own data once it is imported.

A number of other user-defined metrics can be included in the metadata file. Examples of user-defined metadata for individuals include:

sex	Sex of the individual (Male, Female)
maturity	Maturity of the individual (Adult, Subadult, juvenile)
lat	Latitude of the location of collection
long	Longitude of the location of collection

These optional data are provided by the user in the same metafile used to assign id labels and assign individuals to populations at the time of reading in the data.

The individual metadata are held in the genlight object as a dataframe named `ind.metrics`. You can check the names of all available `ind.metrics` via:

```
names(gl@other$ind.metrics)
```

```
[1] "id"      "pop"     "lat"     "lon"     "sex"     "maturity"  "collector" "location"  "basin"     "drainage"
```

and can be accessed using the following form:

```
# Only first 10 entries shown
  gl@other$ind.metrics$sex[1:10]
```

```
[1] Male    Male    Male    Male    Unknown Male    Female  Female  Male    Female
Levels: Female Male Unknown
```

Try these for yourself to see the output they produce, for the testset.gl and for your own data once it is imported.

How dartR stores SilicoDArT data

dartR also stores SilicoDArT presence/absence data in a genlight object, but distinguishes the data from SNP data by setting ploidy=1.

The locus metadata would typically include:

AlleleSequence	Sequence of the tag which is present in samples with genotype score "1"*
TrimmedSequence	Same as the full sequence, but with removed adapters in short marker tags*
AvgReadDepth	Sum of the tag read counts for all samples, divided by the number of samples with non-zero tag read counts*
StDevReadDepth	Standard deviation of the number of tag reads for all samples with non-zero tag read counts*
CallRate	Proportion of samples for which the genotype call is either "1" or "0", rather than "-"
CloneID	Unique identifier of the sequence tag*
OneRatio	Proportion of samples for which the genotype score is "1"
PIC	Polymorphism Information Content
Qpmr	Average of the normalized non-zero tag read counts divided by the standard deviation of the normalized non-zero tag read counts (If standard deviation is zero or near zero, the Qpmr value will be 100).
Reproducibility	Proportion of technical replicate assay pairs for which the marker score is consistent*

Those marked with an * cannot be calculated from the data, and must be provided by the user.

The SilicoDarT data and associated metadata can be accessed in the same way as for SNP data, as described above.

Reading Data into a Genlight Object

There are six paths for reading non-DArT data into dartR.

1) Using the function `import2genind()` from package {adegenet} to convert data from other software into a genind object. This function recognises the files from the following software:

- **GENETIX** (Belkhir, 2004) files with extension ".gtx".
- **GENEPOP** (Raymond, 1995) files with extension ".gen".
- **FSTAT** (Goudet, 1995) files with extension ".dat".
- **STRUCTURE** (Pritchard *et al*., 2000) files with extension ".str" or ".stru".

Once the data have been converted to an {adegenet} genind object, it is necessary to convert it into a genlight object using the function `gi2gl()`, for example:

```
obj <- import2genind(system.file("files/nancycats.gen",
       package="adegenet"))

gl <- gi2gl(obj)
```

2) Using the function `read.PLINK()` from package {adegenet} converts data from the software PLINK (Purcell *et al*., 2007) directly into a genlight object. The input file for this function must be exported from PLINK using the option "–recode A", as described in this function's documentation. An example of PLINK's export command, using the Command Prompt on a PC or Terminal on a Mac, is:

```
> path_to_plink/plink –recode A –file ~/in_file –out
~/out_file
```

3) Using the function `read.snp()` from package {adegenet} to read adegenet's own format file with extension ".snp", which is described in the following document: https://raw.githubusercontent.com/thibautjombart/adegenet/master/tutorials/tutorial-genomics.pdf

4) Using the function `fasta2genlight()` from package {adegenet} to read aligned sequences in FASTA format with extensions ".fasta", ".fas" or "fa".

5) Using the {dartR} function `gl.read.vcf()` to read data from a vcf file and convert it to a genlight object.

6) Using the {dartR} function `gl.read.csv()` to read a csv file, together with locus and individual metadata files, and converted it into a genlight object. The SNP data need to be in one of two forms. SNPs can be coded 0 for homozygous reference, 2 for homozygous alternate, and 1 for heterozygous with NA for missing values; or the SNP data can be coded A/A, A/C, C/T, G/A etc, with -/- for missing values.

Checking that your Data are dartR Compliant

To ensure your manually generated genlight object (say converted from a vcf file) is compliant, be sure to use the function `gl.compliance.check()`. This function will check to see that the genlight object conforms to expectation in regard to dartR requirements. This function will add also the slots loc.metrics and ind.metrics to the genlight object.

```
gl <- gl.read.vcf(system.file('extdata/test.vcf',
     package='dartR'))

gl <- gl.compliance.check(gl)
```

To recalculate the locus metrics (where this is possible), you can use

```
gl <- utils.recalc.metrics(gl)
```

Adding metadata to individuals and loci

Metadata for individuals should be added manually to the genlight object if you use the reading paths described above. The individual metadata should be stored in a csv file with at least two columns with the headings: `id`, containing individual names, and `pop`, containing the population name of each individual. For reading the individual metadata and attaching it to a genlight object you can use:

```
gl$other$ind.metrics <-
     read.table(“ind_metrics.csv”,header=T,sep =",")
```

You can then add the population of each individual to the accessor `pop` using:

```
pop(gl) <- gl$other$ind.metrics$pop
```

and adding the individual names to the accessor `indNames` using:

```
indNames(gl) <- gl$other$ind.metrics$id
```

The process of adding other locus metrics to the genlight object is slightly different to that of ind.metrics because the slot loc.metrics has been already generated by the function `gl.compliance.check()`. The first step of this process is to prepare a csv file with the locus metrics you want to add, such as locus position or chromosome information. In this file, loci should be in the rows and information for each locus in the columns, and each column should have a heading with a sensible name, for example `pos` for locus position and `chr` for the locus chromosome. It is important to double-check that the order of the loci in the csv file agrees with the order in the genlight object. You can check this for example by using the function `locNames()`.

The next step is to read the csv file by using for example:

```
  loc_metrics <-
read.table(“loc_metrics.csv”,header=T,sep =",")
```

After reading the csv file, you can add the additional locus metrics to the slot loc.metrics by merging these two data.frames and inserting them back to the slot loc.metrics:

```
  gl$other$loc.metrics <-
cbind(gl$other$loc.metrics,loc_metrics)
```

Finally, you can add locus metrics to some genlight accessors, for example:

```
gl$position <- gl$other$loc.metrics$pos

gl$chromosome <- gl$other$loc.metrics$chr
```

Saving a genlight object

Reading the data in from an Excel spreadsheet and converting it to a genlight object takes a lot of computation, and so time. You will also have done some tidying up of the data. It is sensible to save your genlight object in binary form using

```
gl.save(gl,file="tmp.Rdata")
```

and then read it in again with

```
gl.new <- gl.load("tmp.Rdata")
```

Try saving `gl` or your own genlight object to your workspace, and verify that it has been saved to the appropriate directory. Then restore it to a new genlight object.

History

A history of manipulations is also stored in the genlight object. This is convenient should you wish to interrogate (or indeed repeat) the process that created the current version of the genlight object.

Display the history of a genlight object using

```
gl.report.history(gl)
```

Repeat the history of a genlight object using

```
gl.play.history(gl)
```

Exercises

Exercise 1

Open the csv files platy_test.csv and platy_ind.csv in Excel. You can find the location of these files by using:

```
system.file('extdata','platy_test.csv',package='dartR')

system.file('extdata','platy_ind.csv',package='dartR')
```

The first file contains the genotypes of 13 individuals for six loci. Note the genotypes are formatted using the letters to represent the four DNA nucleotides: A, T, G and C. Now examine the second file that contains the individual metadata and note that this file contains the two mandatory columns: id and pop.

Read the SNP data and the individual metadata into dartR as a genlight object called gl using the function `gl.read.csv()`. Now check its contents by using genlight accesors such as `popNames(gl)` and `indNames(gl)`.

Determine which one of the two alleles in each SNP is used as the reference allele. For this you can compare the data from the file platy_test.csv and accesing the genotypes of the genlight object by using `as.matrix(gl)`. Remember that that the homozygous for the reference allele in the genlight object is coded as 0, heterozygote as 1 and homozygote for the alternative alleles as 2.

Tidy up the workspace

We have created files that we will not use again, so they should be removed from the workspace.

```
rm(gl.new, gl.1row, gl.2row)
```

Where have we come?

In this Session, we have covered a range of topics on data entry, the storage of data and some preliminary approaches to examining those data. Having completed the Session, you should understand:

- What is a sensible pipeline for preliminary handling of your SNP data.
- How a genlight object is organised in terms of the SNP and how locus and sample metadata are associated with the genotypes.
- The different types of locus metadata.
- How to read your data into a dartR genlight object and check its compliance.
- How to interrogate the locus and individual (specimen/sample) metadata.

References

Belkhir, K. (2004). GENETIX 4.05, logiciel sous Windows TM pour la génétique des populations. http://www. genetix. univ-montp2. fr/genetix/genetix. htm.

Goudet, J. (1995). FSTAT (Version 1.2): A computer program to calculate F-statistics. Journal of Heredity, 86(6), 485-486.

Jombart T. and Caitlin Collins, C. (2015). Analysing genome-wide SNP data using adegenet 2.0.0. http://adegenet.r-forge.r-project.org/files/tutorial-genomics.pdf

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

Jombart, T., Kamvar, Z.N., Collins, C., Lustrik, R., Beugin, M.P., Knaus, B.J., Solymos, P., Mikryukov, V., Schliep, K., Maié, T., Morkovsky, L., Ahmed, I., Cori, A., Calboli, F. and Ewing, R.J. (2018). Package 'adegenet'. Version 2.1.1. Exploratory Analysis of Genetic and Genomic Data. https://github.com/thibautjombart/adegenet

Peterson BK, Weber JN, Kay EH, Fisher HS, Hoekstra HE (2012) Double Digest RADseq: An Inexpensive Method for *De Novo* SNP Discovery and Genotyping in Model and Non-Model Species. PLoS ONE 7(5): e37135. **https://doi.org/10.1371/journal.pone.0037135**

Pritchard, J. K., Stephens, M., & Donnelly, P. (2000). Inference of population structure using multilocus genotype data. Genetics, 155(2), 945-959.

Purcell, S., Neale, B., Todd-Brown, K., Thomas, L., Ferreira, M. A. R., Bender, D., . . . Daly, M. J. (2007). PLINK: a tool set for whole-genome association and population-based linkage analyses. The American Journal of Human Genetics, 81(3), 559-575.

Raymond, M. (1995). GENEPOP (version 1.2): population genetics software for exact tests and ecumenicism. Journal of Heredity, 86, 248-249.

Ende