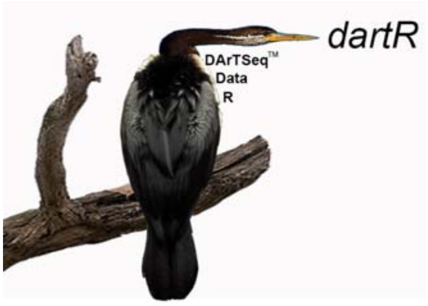
Wildlife Genomics@UC
wildlifegenomics.biomatix.org

UNIVERSITY OF CANBERRA

SNP Analysis using dartR



Guide to Basic Filtering

Version 1.8.3

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**

Contents

Session 1: Basic Filtering

Overview

DArT Pty Ltd has already done much of the filtering of the sequences used to generate your SNPs that would normally be undertaken by researchers who generate their own ddRAD data. Here we present some other filters that you might wish to apply.

It is a good idea to run the gl.report functions in advance of filtering to provide a foundation for selecting thresholds.

Several filters are available to improve the quality of the data represented in your genlight object. The basic ones are:

```
gl <-
  gl.filter.reproducibility()
```

filter out loci for which the reproducibility (strictly repeatability) is less than a specified threshold, say threshold = 0.99

```
gl <- gl.filter.callrate()
```

filter out loci or individuals for which the call rate (rate of non-missing values) is less than a specified threshold, say threshold = 0.95

```
gl <- gl.filter.monomorphs()
```

filter out monomorphic loci and loci that are scored all NA

```
gl <-gl.filter.secondaries()
```

filter out SNPs that share a sequence tag, except one retained at random

```
gl <- gl.filter.hamming()
```

filter out loci that differ from each other by less than a specified number of base pairs

```
gl <- gl.filter.rdepth()
```

filter out loci with exceptionally low or high read depth (coverage)

```
gl <- gl.filter.taglength()
```

filter out loci for which the tag length is less that a threshold

```
gl <- gl.filter.overshoot()
```

filter out loci where the SNP location lies outside the trimmed sequence tag

The order of filtering can be important and requires some thought. Filtering on call rate by individual before filtering on call rate by locus or choosing the alternative order will depend on the weight placed on losing individuals versus losing loci, for example.

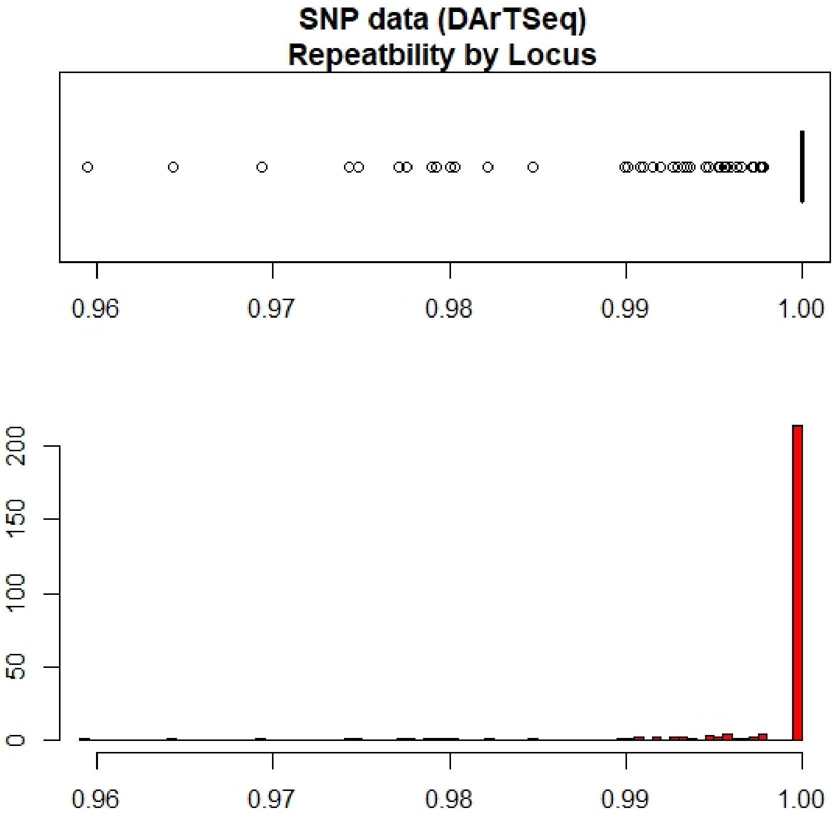
Example: Filtering on reproducibility

First, we should examine the distribution of reproducibility measures (RepAvg) in our dataset.

```
gl.report.reproducibility(testset.gl,plot=TRUE)
```

```
Starting gl.report.reproducibility
  Processing a SNP dataset
  No. of loci = 255
  No. of individuals = 250
  Miniumum repeatability:  0.96
  Maximum repeatability:  1
  Mean repeatability:  0.998
Boxplot adjusted to account for skewness

Completed: gl.report.reproducibility
   Threshold Retained Percent Filtered Percent
1  1.0000000      214    83.9       41    16.1
2  0.9979729      214    83.9       41    16.1
3  0.9959459      222    87.1       33    12.9
4  0.9939189      231    90.6       24     9.4
5  0.9918918      237    92.9       18     7.1
6  0.9898647      242    94.9       13     5.1
7  0.9878377      242    94.9       13     5.1
8  0.9858107      242    94.9       13     5.1
9  0.9837836      243    95.3       12     4.7
10 0.9817565      244    95.7       11     4.3
11 0.9797295      246    96.5        9     3.5
12 0.9777025      248    97.3        7     2.7
13 0.9756754      250    98.0        5     2.0
14 0.9736483      252    98.8        3     1.2
15 0.9716213      252    98.8        3     1.2
16 0.9695942      252    98.8        3     1.2
17 0.9675672      253    99.2        2     0.8
18 0.9655401      253    99.2        2     0.8
19 0.9635131      254    99.6        1     0.4
20 0.9614860      254    99.6        1     0.4
21 0.9594590      255   100.0        0     0.0
```



Clearly, with a minimum repeatability of 0.96 and a maximum of 1 across loci, we can be fairly stringent in our choice of a threshold. A value of 0.99 will not result in the loss of much data.

We now filter on that basis.

```
gl <- gl.filter.reproducibility( testset.gl,
           threshold=0.99, verbose = 3)
```

```
Starting gl.filter.reproducibility
  Processing a SNP dataset
  Identifying loci with repeatability below : 0.99
  Removing loci with repeatability less than 0.99

  Summary of filtered dataset
  Retaining loci with repeatability >= 0.99
  Original no. of loci: 255
  No. of loci discarded: 14
  No. of loci retained: 241
  No. of individuals: 250
  No. of populations:  30

Completed: gl.filter.reproducibility
```

Only 14 loci out of 255 were deleted.

It is wise not to filter too heavily on reproducibility. A threshold of 1 is often tempting but can result is some bias being introduced.

Exercises

1. Just in case you have accidentally modified the genlight object `gl`, recreate it by copying it from `testset.gl`.
2. Filter `gl` using the filters listed above. Request a report first to inform your choice of threshold. Be sure to set plot=TRUE to examine the distribution of each parameter and optionally smearplot=TRUE to examine the smear plot.

Recalculating locus metadata after filtering

Remember, the locus metrics are no longer valid if individuals or populations are deleted from the dataset. For example, if you filter out a population for which the individuals have particularly bad call rates, then the call rate parameter held in the locus metrics will no longer be accurate. It will need to be recalculated. This is true of many of the locus metrics.

So, after filtering your data, it is wise to recalculate the locus metrics with

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

Try this for yourself on genlight object `gl` after filtering or on your own data

Similarly, when filtering has resulted in removal of some individuals or populations, variation at several loci may be lost. Some loci may even be scored as missing across all individuals. You may wish to remove these monomorphic loci from your dataset with

```
gl <- gl.filter.monomorphs(gl)
```

Try this for yourself on genlight object `gl` after filtering or on your own data

Where have we come?

The above Session was designed to give you an overview of the scripts in dartR for filtering your data. Having completed this Session, you should now able to:

- Filter on call rate, repeatability, secondaries, hamming distance, and minor allele frequency.
- Recalculate locus metrics after deleting individuals or populations as part of the filtering process.
- Filter out resultant monomorphic loci.

Further reading

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

Gruber, B., Unmack, P.J., Berry, O. and Georges, A. 2018. dartR: an R package to facilitate analysis of SNP data generated from reduced representation genome sequencing. Molecular Ecology Resources, 18:691–699

Ende