Automated scoring of AFLPs using rawgeno v 2.0, a free R CRAN library

Article *in* Methods in molecular biology (Clifton, N.J.) · June 2012

DOI: 10.1007/978-1-61779-870-2_10 · Source: PubMed

Nils Arrigo
University of Lausanne
179 PUBLICATIONS **1,806** CITATIONS

SEE PROFILE

Nadir Alvarez
Natural History Museum of Geneva
384 PUBLICATIONS **3,403** CITATIONS

SEE PROFILE

Automated scoring of AFLPs using RawGeno v 2.0, a free R CRAN library

Authors: Arrigo Nils[1,2], Holderegger Rolf[3], Alvarez Nadir[2]

[1]Laboratory of Evolutionary Botany, Institute of Biology, University of Neuchâtel, 11 rue Emile-Argand, 2009 Neuchâtel, Switzerland

[2]Department of Ecology and Evolution, University of Lausanne – UNIL Sorge, Biophore Building, 1015 Lausanne, Switzerland

[3]WSL Swiss Federal Research Institute, Zürcherstrasse 111, 8903 Birmensdorf, Switzerland

Keywords: Scoring optimization, command lines, graphical user interface, data mining, bin editing

Abstract

Amplified Fragment Length Polymorphisms (AFLPs) are a cheap and efficient protocol for generating large sets of genetic markers. This technique has become increasingly used during the last decade in various fields of biology, including for instance population genomics, phylogeography and genome mapping. Here, we present RawGeno, an R library dedicated to the automated scoring of AFLPs (i.e. the coding of electropherogram signals into ready-to-use datasets). Our program includes a complete suite of tools for binning, editing, vizualizing and exporting results obtained from AFLP experiments. RawGeno can either be used with command lines and program analysis routines or through a user-friendly graphical user interface. We describe

the whole RawGeno pipeline along with recommendations for (i) setting the analysis of electropherograms in combination with PeakScanner, a program freely distributed by Applied Biosystems, (ii) performing quality checks, (iii) defining bins and proceeding to scoring, (iv) filtering non-optimal bins and (v) exporting results in different formats.

1. Introduction

The amplified fragment length polymorphism (AFLP) technique is increasingly used in phylogeographic and population genomics studies, particularly in non-model organisms for which no prior DNA sequence information is available ***(1)***. This relatively cheap technique is based on complete endonuclease restriction digestion of total genomic DNA followed by selective PCR amplification and electrophoresis of a subset of fragments, resulting in a unique, (theoretically) reproducible fingerprint for each individual. Although the AFLP technique is able to generate a large number of informative markers, the success of this method is compromised by different factors ***(2)***. For instance, manual scoring relies on visual inspection and subjective interpretation of the electropherotic profiles during a time-consuming and tedious task. In the last decades, several improvements in automatic scoring have been proposed and implemented in commercial software [see ***(3)*** for a review]. However, until recently, no free open-source software was available to process AFLP data from raw data to ready-to-use presence/absence binary matrices. Two years ago, we developed RawGeno 1.0 ***(4)***, a library performing automated binning, scoring and data mining analyses under the R CRAN environment, based on outputs from the freely available electropherogram-analyzing software PeakScanner (Applied Biosystems, Foster City, USA, http://www.appliedbiosystems.com/peakscanner). Implementing RawGeno 1.0 solutions in a free environment has provided an accessible and accurate solution to many users (with 415 downloads from http://sourceforge.net/projects/rawgeno one year after release). Here, we present RawGeno 2.0, an updated version of this software, built around an optimized, less time-consuming algorithm, implementing new features such as binning editing. We provide examples both in a user-friendly and in a command-line interface. In order to allow users to customize queries (which may vary depending on dataset quality/size), we further provide tips for setting analyses and improving output robustness. All stages of a whole analysis are detailed according to the following five sections: (i) importing features; (ii) quality check; (iii) binning and scoring algorithms; (iv) bin filtering; (v) exporting options.

2. Program Utilization

2.1. Overview of the analysis

Analysing AFLP electropherograms is achieved using two programs: PeakScanner and RawGeno. Whereas PeakScanner detects AFLP peaks along electropherograms and calculates their intensity and size in base pairs (by relying on an internal size standard included in electrophoresis), RawGeno proceeds to the binning and scoring of AFLP electropherograms.

RawGeno includes several filters to assess the quality of electropherograms and checks the consistency of binning. In addition, several preliminary analyses are available to the user for making biological inferences and/or remove outlier samples. Finally, several functions allow exporting resulting data sets into properly formatted files for further analyses.

2.2. Building up a scoring project

In RawGeno 2.0 the AFLP scoring project should be organized according to the following procedure.

a. Create a folder (hereafter “project folder”) from which the project will be managed.
b. In this folder, add a sub-directory including all electropherogram files (*.fsa). Also add an R shortcut (Windows users) to conveniently launch RawGeno scoring sessions. Right-clicking on this R shortcut allows defining the default working directory of R by specifying it into the “Start into” addressing field of the shortcut. Copy-pasting the project folder address into this field will set-up the working directory of R accordingly.
c. Create a text-tabulated table listing individuals included in the project (hereafter referred to as “info table”). The info table is optional as the minimal RawGeno analysis can proceed

without it. However, RawGeno includes several functions relying on this table, for instance to label individuals during preliminary analyses or facilitate the sorting and selection of individuals (for example, according to populations or species) during the production of exports. Therefore, the info table should include any additional relevant information that the user would like to consider. It must contain at least the name of individuals (i.e. in a column named “Tag”) and any supplementary information in extra columns.

*2.3. Obtaining the raw data from *.fsa files using PeakScanner*

The analysis of AFLPs starts by using PeakScanner in order to detect peaks along electropherograms and to calculate their size. The procedure is highly automated, letting the user set peak detection parameters and check the quality of electropherograms. The peak detection parameters are set up using a so-called “Analysis Method”, which is available from the graphical interface in PeakScanner (menu “Resources/Manage Analysis Methods”). Typically, a proper peak detection attempts to detect only peaks that are biologically relevant and exclude peaks only reflecting technical background noise.

We advise to set the “Analysis Method” using the following guidelines.

a. Prior to the detection of peaks *per se*, a light smoothing of electropherograms might be desirable, in order to eliminate small secondary peaks due to technical background noise.

b. The detection of peaks is achieved through a “sliding window” analysis that inspects electropherograms locally. Within the inspected region, PeakScanner first creates a modelled version of the electropherogram by fitting a polynomial curve to the data. Peaks are detected according to this modelled signal, based on their absolute width. Therefore, the detection sensitivity is adjusted by modifying the width of the sliding window (i.e. in terms of data points, the smaller it is, the more sensitive the procedure becomes), the goodness of fit reachable by the polynomial curve (again, increasing the polynomial degree of fitting increases the detection sensitivity) and the minimal width above which a peak is recorded as present. We advise to use the default parameters as a starting point, as they have been shown to provide reliable results ***(4, 5)***: set 15 points for the sliding window width, use a third degree polynomial curve and consider peaks that at least have two measurement points of half-width.

c. Downstream to peak detection, PeakScanner filters peaks according to their absolute fluorescence intensity, i.e. the peak height, measured in relative fluorescent units (rfu). Visually checking electropherograms obtained from blank samples generally helps to adjust the fluorescence threshold to the upper limit of the technical background noise. While some applications might benefit from considering only peaks with a strong fluorescence (e.g. greater than 150 rfu to provide conservative estimates for band presence statistics), most users will prefer using a more permissive threshold at this stage and apply *a posteriori* filtering strategies based on bin quality statistics ***(6, 7)***. We advise to use 50 rfu as a minimal fluorescence for considering individual AFLP peaks.

d. Save the customized "Analysis Method" in order to use it during electropherogram analysis.

e. Once the "Analysis Method" is set up, import the electropherograms (stored as *.fsa files) into PeakScanner using the "Add Files" button.

f. Define the size standard and the "Analysis Method" to be used for all individuals included in the project (set this information for the first individual, then select the columns "Size Standards" and "Analysis Method" and use the "ctrl+D" keyboard shortcut to apply these settings to the remaining individuals).

The detection and sizing of peaks is processed using the "Analysis" menu, from the graphical interface. Once achieved, electropherograms can be visualized and compared among individuals. This might help identifying AFLP reactions that were not successful (e.g. individuals with a systematically low fluorescence or showing abnormal peaks). Removing such individuals prior to the RawGeno analysis will help to enhance the final quality of scoring.

The PeakScanner analysis ends with a simple export process in which the list of peaks detected throughout the complete set of analyzed individuals is stored in a table. This is achieved using the "Export/Export Combined Table" menu, producing a text-tabulated file containing the size, height, area and width of all detected peaks (this can be checked using the “Edit Table Settings” menu).

2.4. From raw data to ready-to-use matrices using RawGeno

2.4.1. Installing RawGeno

RawGeno is freely available from http://sourceforge.net/projects/rawgeno as a zip file. In windows, the installation is achieved either using the graphical user interface of R (menu “Packages/Install package(s) from local zip files”) or the following command line in the R console:

```
utils:::menuInstallLocal()
```

Installing RawGeno with Linux requires decompressing RawGeno.zip into the library folder of R. Using the shell command line, this is done as follows:

```
sudo mv RawGeno.zip /usr/lib64/R/library/RawGeno.zip
cd /usr/lib64/R/library/RawGeno.zip
unzip RawGeno.zip
rm RawGeno.zip
```

Finally, RawGeno requires the installation of two companion packages: vegan and tkrplot, that both are available from usual R CRAN repositories (Linux users should see Note 1). Their installation is achieved either using the graphical user interface of R (menu “Packages/Install package(s) from CRAN”) or with the following command lines (prompted into the R console):

```
install.packages("vegan")
install.packages("tkrplot")
```

2.4.2. Importing PeakScanner results

All following steps are performed in the R CRAN environment, using the R shortcut described above (or ensuring that the correct working directory has been selected). Once the RawGeno package has been installed, it can be called applying the following command line:

a. Call RawGeno, vegan and tkrplot as libraries into R and launch the graphical user interface

```
require(RawGeno)
require(vegan)
require(tkrplot)
RawGeno()
```

b. Importing the PeakScanner text-tabulated file in RawGeno can then be done using the graphical user interface (menu “RawGeno/1. Files/Electroph./PeakScanner (*.txt)”) or using the following command lines:

Choose interactively the PeakScanner file

```
myfile=tk_choose.files(caption='Choose PeakScanner File')
OPENAFLP(myfile, pksc=T, dye="B")
```

Or explicitly specify the path of file of interest

```
mypath="C:/MyDocuments/MyPeakScannerFile.txt"
OPENAFLP(mypath, pksc=T, dye="B")
```

During importation, RawGeno handles a single dye color at a time, which is user-specified and considers the “dye” parameter with the following values: "B" (blue; FAM), "G" (green; HEX), "Y" (yellow; NED), “R” (red; ROX) or “O” (orange; LIZ). If electrophoresis was achieved using several dyes simultaneously (e.g. multiplexing of PCR products), each dye must be analysed separately in RawGeno. Datasets obtained from several dyes can be merged *a posteriori* in a final binary table (see below).

2.4.3. Quality check

Because the detection of AFLP peaks is based on a defined threshold, it is not easy to handle reactions showing electropherograms with varying intensities (see Note 2). When improperly handled, such a situation leads to the inclusion of samples characterised by many false-absences in the final dataset. Although the only way to correctly address this issue is a robust wet-lab protocol, RawGeno still attempts limiting the influence of low quality AFLPs on binning and scoring, by filtering individuals that were unsuccessful. Here, the variability in the number of peaks detected per individual is used as a proxy of AFLP reactions quality. Empirical evidence shows that this statistics is dependent of the specific dataset used and the biological organism studied. The lower bound of this distribution most generally includes individuals with low AFLP intensities, characterized by many AFLP peaks that remain undetected in the electropherograms. Because such individuals usually represent a small fraction of the complete project, we advise removing them from the dataset. The upper bound of the distribution can either reflect a biologically relevant signal (e.g. hybridization) or a technical bias (e.g. contamination, odd PCR reaction). Such individuals should be either discarded or identified as outliers for proper interpretation in further analyses.

RawGeno includes the two following options for checking quality of individuals:

a. Filter individuals according to the number of detected peaks, by manually selecting individuals that should be kept for further analysis or by using a dedicated device (Fig 1 a). The command line version of this operation relies on percentiles and conserves individuals within 5%-95% bounds of the detected peaks distribution.

Retrieve imported electropherograms

```
data.electroph=AFLP$all.dat
```

Compute number of AFLP peaks per individual

```
pk.per.smp=table(data.electroph$sample.id)
```

Compute 5% and 95% quantiles

```
qtles=quantile(pk.per.smp, probs=c(0.05, 0.95))
```

Determine what samples can be kept

```
to.keep=which(pk.per.smp>=qtles[1] & pk.per.smp<=qtles[2])
```

Filter electropherograms, keep only retained individuals and update individuals indexing accordingly

```
smp.ok=match(data.electroph$sample.id, to.keep)
data.clean=data.electroph[is.na(smp.ok)==F, ]
data.clean$sample.id=as.factor(data.clean$sample.id)
levels(data.clean$sample.id)=1:length(to.keep)
smp.names=AFLP$samples.names
smp.clean=smp.names[to.keep]
AFLP$all.dat=data.clean
AFLP$samples.names=smp.clean
```

b. Check the quality of individuals, by taking into account their position in PCR plates (only available from the graphical interface, Fig 1 b). This display helps to highlight systematic biases having a technical origin (e.g. pipetting errors or thermocycler bias) and to identify batches of individuals that were not successful.

2.4.4. Binning

Building a presence/absence matrix requires recognizing which AFLP peaks are homologous across individuals. This procedure relies on the size of peaks along electropherograms and assumes homology for peaks sharing identical sizes. Because peak sizes are determined empirically using electrophoresis, measurements generally include technical variations preventing the observation of strictly identical sizes across homologous AFLP peaks. Indeed, Holland *et al.* ***(8)*** reported measurement variations ranging between 0 and 0.66 bp (with 0.08 bp in average) for replicated AFLP peaks. Therefore, properly recording the signals of AFLP peaks asks for taking into account size variations by defining size categories (i.e. “bins”) into which the presence / absence of AFLP peaks are recorded. Bins are characterised by their position along the electropherogram (i.e. the average size of peaks they include) and their width (the size difference between the longest and shortest peaks included in the bin). RawGeno uses a binning algorithm relying on the size of AFLP peaks over all individuals included in the project and defines bins in a way that respects the two following conditions (Fig 2 a).

a. The first constrain is a maximal bin width. This prevents the definition of too large bins that could lead to homoplasy (i.e. erroneously assigning non-homologous AFLP peaks within the same bin). This limit is set using the MaxBin parameter. We advise to use MaxBin values ranging between 1.5 and 2 bp (see Note 3). Using small values (i.e. MaxBin < 0.5 bp) should be avoided as this generally causes oversplitting, a situation where the presence / absence of homologous AFLP bands are coded using an exaggerated number of bins.

b. The second constrain prevents the assignment of more than one peak from the same individual within the same bin [i.e. “technical homoplasy”, as defined in ***(4)***]. If such a situation occurs, RawGeno defines two separate bins in which the two peaks are assigned. This constrain can be relaxed by increasing the “MinBin” parameter in order to include two peaks of the same individual in the same bin (when not exceeding the “MinBin” value in size difference). Such a relaxing might be desirable, for instance when artefactual peaks (i.e. shoulder, stutter or secondary peaks bordering the authentic peak in a same individual) lead to the definition of numerous extra-bins. In such a situation, artefactual peaks can cause the local definition of extra bins into which homologous peaks can be inconsistently assigned. We advise to use MinBin values ranging between 1 and 1.5 bp. The binning is launched through the graphical interface (menu “RawGeno/2. Scoring”) or using the following command lines:

Proceed to binning

```
EXTRACTAFLP(all.dat=AFLP$all.dat,
samples.names=AFLP$samples.names, MAXBIN=2, MINBIN=1)
```

View results (that are assigned into a “data.binary” object)

```
attributes(data.binary)
data.binary$data.binary
```

Binning is an automated and straightforward analysis step that users might want to review interactively. RawGeno includes a visualization device for manually editing the binning by adding, removing or modifying the width and position of bins (this tool is only available from the graphical interface, Fig 2 b). This device includes several help-to-decision statistics such as the average size and the number of presences associated with each bin.

2.4.5. Filtering

Once defined, bins can be filtered according to their properties and/or quality. Note that such filtering strategies require analyzing AFLP reactions with a consistent quality across individuals. Filtering options are accessible from the scoring menu when using the graphical user interface (menu “RawGeno/2. Scoring”). Command line users will set accordingly the EXTRACTAFLP function.

Proceed to binning and filtering simultaneously

```
EXTRACTAFLP(all.dat=AFLP$all.dat,
samples.names=AFLP$samples.names, MAXBIN=2, MINBIN=1, RMIN=100,
RMAX=500, cutRFU=50, who='B', thresh=95)
```

In its current version, RawGeno includes three kinds of filters.

a. The size filter restricts binning to a given portion of the electropherogram (RMIN and RMAX parameters). We advise to limit the binning to peaks included in the range of the size ladder, because their size is accurately interpolated by PeakScanner (in contrast to larger peaks where the size is extrapolated). We recommend discarding peaks with small sizes (i.e. smaller than 100 bp, RMIN=100) as they are more likely to be homoplasic ***(9, 10)***. In addition, large size peaks should as well be considered cautiously because their fluorescence intensity might not always be consistent across individuals. Because this upper limit might vary according to datasets, we advise to run preliminary analyses and check electropherograms to confidently determine it.

b. The second filter eliminates bins according to their average fluorescence (i.e. the cutRFU parameter). This filter assumes that bins with a high average fluorescence retrieve a more consistent signal than bins with a low fluorescence. The rationale for this strategy is the following. The fluorescence of an AFLP fragment largely determines its detection probability during the PeakScanner analysis of electropherograms. Therefore, fragments that systematically produce low fluorescences are more likely to be erroneously recorded as absent from electropherograms as they might pass the threshold in some reactions but not in others just by chance. The computation starts by normalizing fluorescence intensities across samples using the sum normalization method ***(6)***, before computing the average fluorescence of each bin. Hence, keep in mind that the cutRFU parameter applies on normalized values and does not scale with fluorescence measures provided in PeakScanner. Setting this filter is dataset-dependent and we recommend running several trials before producing a definitive dataset (see Note 4). Refer to the works of Whitlock *et al.****(6)*** and Herrmann *et al.* ***(7)*** for more sophisticated filtering R scripts based on fluorescence intensities. Bridging RawGeno with these algorithms is achieved by exporting fluorescence results instead of a binary matrix. Once binning is achieved, use the following command lines.

Retrieve raw fluorescence results, stored in a matrix corresponding to the usual binary matrix

```
mat.rfu=t(data.binary$data.height.raw)
```

For normalised fluorescences (sum normalization), use instead

```
mat.rfu=t(data.binary$data.height)
```

Prepare for export and save as a text-tabulated file (refer to programs' documentation for properly formatting files).

```
mat.rfu[is.na(mat.rfu)==T]=0
write.table(mat.rfu, "MyFluorescenceFile.txt", quote=F,
sep="\t")
```

c. The reproducibility filter evaluates bin quality according to their robustness across AFLP reactions, by relying on replicated samples. This filter assumes that replicated individuals were selected randomly from the original dataset, so as to scan the genetic diversity at best (see Notes 2 and 4). Keep in mind that RawGeno identifies replicated individuals using their names. Replicates must be named using the original individual name plus a suffix letter. The suffix is matched using the “who” parameter of the filtering algorithm. As an example, “mysample.fsa” and “mysampleB.fsa” are a pair of original-replicated samples, being identified with a “B” suffix (therefore, set who = “B” when filtering). For each bin, RawGeno compares original to replicated individuals and calculates the percentage of original-replicated pairs for which the AFLP signal is successfully reproduced. Bins where reproducibility cannot reach a satisfactory rate (i.e. the “thresh” parameter, a user-defined reproducibility percentage) are eliminated from the final dataset.

2.4.6. Review of results

RawGeno offers two displays for exploring scoring results (menu “RawGeno/3. Quality Check/Samples Checking”). The binary matrix can be directly visualized using a heatmap, showing individuals sorted according to their genetic relatedness. Alternatively, individuals can be examined with a principal coordinates analysis. Both displays allow plotting quality statistics or external information (i.e. picked from the “info table” cited above) against the AFLP results. These displays are only available from the graphical user interface (Fig 3 a, b).

2.4.7. Exporting files

RawGeno includes functions for producing binary tables and standard exports for Arlequin, Hickory, Popgen, AFLPsurv, STRUCTURE 2.2, Mltr, Spagedi, Dfdist, Treecon, Baps, PAUP, Structurama, MrBayes and NewHybrids ***(11)***. Furthermore, these exports can be sliced according to information provided in the info table and produce ad-hoc subsets. These functions are accessible from the graphical interface (“RawGeno/4. Save”) or using the following command lines.

Retrieve the binary matrix from RawGeno and import the info table

```
mat01=t(data.binary$data.binary)
matinfo=read.delim("MyInfoTable.txt", header=T)
```

Cross-reference the AFLP results to the info table

```
popsA = row.names(mat01)
popsB = matinfo$Tag
mat01= mat01[match(intersect(popsA, popsB), popsA), ]
matinfo = matinfo[match(intersect(popsA, popsB), popsB), ]
```

Remove monomorphic bins

```
mat01=mat01[ , colSums(mat01)>0 & colSums(mat01)<nrow(mat01)]
```

Produce the required outputs, e.g. for STRUCTURE 2.2. (refer to the library documentation for further details regarding exporting functions).

```
Structure.popsD(mat01, pops=matinfo$MyPopsColumn, path=getwd(),
name="MyStructure2.2File.txt")
```

Users willing to analyse AFLP signals as codominant markers ***(12)*** should use command lines described above to export fluorescence data (a proxy of allele copy number in genomes) associated with binary matrices.

2.4.8. Handling data from previously scored projects

Users willing to merge, visualize and/or produce exports from datasets that were already scored can import binary tables within RawGeno using the “RawGeno/1. Files/Import” menu. From the command line, such an operation is done as follows.

Select files to merge

```
list.merge=tk_choose.files(caption='Choose Files to Merge')
```

Or specify a directory in which the binary matrices are stored.

```
mypath="C:/MyDocuments/MyBinaryMatricesDirectory"
list.merge=dir(mypath, pattern='.txt')
```

Proceed to merging

```
MERGING(transpose = "indRows", exclude = T, replacewith = NA)
```

The transpose parameter states whether the binary matrices store individuals as lines (“indRows”) or as columns (“indColumns”), the exclude parameter defines whether individuals that are not shared by all matrices will be removed from the final merged dataset (exclude = “T”). If kept (exclude = “F”), individuals with missing AFLP genotypes will be completed using NA values (replacewith = NA) when no data is available. Note that the merged matrix is stored into a “mergedTable” object. Visualization and exports can be performed using the graphical user interface, as described above. Command lines for exporting merged matrices are given below.

```
mat01=mergedTable
matinfo=read.table("MyInfoTable.txt", header=T)
popsA = row.names(mat01)
popsB = matinfo$Tag
mat01= mat01[match(intersect(popsA, popsB), popsA), ]
matinfo = matinfo[match(intersect(popsA, popsB), popsB), ]
mat01=mat01[ , colSums(mat01)&colSums(mat01)<nrow(mat01)]
```

```
Structure.popsD(mat01, pops=matinfo$MyPopsColumn, path=getwd(),
name="MyStructure2.2File.txt")
```

3. Conclusions and perspectives

We present here a complete suite of tools to automate the scoring of AFLP datasets using free software applications Our program proposes an integrated solution to manage all the components of the analysis pipeline. Accordingly, samples are checked by removing non-satisfactory electropherograms at the very beginning of the analyses and AFLP genotypes are associated with user-specified information while producing ad-hoc exports. Bins are managed using an automated algorithm and can be edited manually using a dedicated graphical user interface.

As a next milestone, we plan to develop RawGeno into two complementary directions: incorporating the handling of electropherograms (which is now is part of PeakScanner) and the enhancement of bin filtering possibilities. Indeed, the RawGeno version currently under development already integrates functions for detecting and calculating the size of AFLP peaks along electropherograms. Finally, achieving connections with the R scripts of Herrmann *et al.* ***(6)*** and Whitlock *et al.* **(7)** is another way to improve RawGeno.

4. Notes

Note 1

Linux users might need to run R as “sudo” users to properly install companion packages (i.e. vegan and tkrplot). In addition, troubles might arise because R libraries are downloaded as source code and compiled locally before being installed. This requires that all compilers needed by R (such as gc, gcc, gcc-fortran and others) have been installed locally, before attempting the installation of external R packages. In OpenSUSE, the necessary compilers can be obtained using YaST2 (into the rpm groups dedicated to development tools). Ubuntu users are more fortunate because Synaptic Manager can install ready-to-use R libraries in addition to usual compilers (refer to http://cran.r-project.org/bin/linux/ubuntu/README for further details regarding repository addresses).

Note 2

Whereas manual scoring allows permanent but subjective adjusting of the criteria defining whether an AFLP peak should be recorded as present or absent, automated peak detection algorithms apply uniform fluorescence thresholds. This requires datasets showing low AFLP quality variation, because fluorescence differences between samples will be reflected in the final binary matrix. For instance, we encountered problematic situations with AFLP reactions showing different fluorescence offsets among PCR plates. These “plate effects” can be highlighted with principal coordinate analyses, as samples are clustered according to PCR plates (i.e. due to plate-specific losses of AFLP bands). These situations are especially difficult to handle without repeating wet-lab experiments. Solving this problem computationally remains difficult and asks for setting sample (or plate)-specific detection sensitivities, which is beyond PeakScanner possibilities. The present version of RawGeno conservatively proposes the removal of samples with unusual AFLP profiles, by relying on the distribution of the number of AFLP peaks. Future developments of RawGeno will attempt analysing electropherograms directly. The problem highlights the crucial importance of using robust and standardized lab protocols. In the following, we list several tips helping to retrieve consistent results from AFLP reactions, when using automated scoring [also consult Bonin *et al.*

(13) and Gugerli *et al.* ***(14)***].

a. Randomize reactions on PCR plates, in order to properly discriminate technical bias from biological signals.

b. Standardize all reaction steps: adjust DNA concentrations after spectrometer quantification, limit impacts of pipetting errors by preparing reaction mixes in batches, run critical reactions in uniform conditions (for instance perform restriction steps in an incubator rather than in a thermocycler) and, importantly, run PCRs on the same thermocycler.

c. Optimize signal detection during genotyping analysis, for instance by increasing the injection time of automated sequencers at the beginning of electrophoresis, and prefer primer pairs showing strong and consistent amplifications.

Note 3

Based on empirical case-study datasets (available from the authors upon request), we provide help-to-decision statistics aimed at guiding users in setting their RawGeno analysis. We analysed 17 AFLP primer datasets: *Aegilops geniculata*, two primer pairs ***(5)***; *Arum* spp., two primer pairs (Espindola *et al.* unpubl. data); *Baldellia* spp., two primer pairs (Arrigo *et al.*, unpubl. data); *Bupleurum ranunculoides*, three primer pairs (Labhardt *et al.*, unpubl. data); *Deschampsia litoralis* and *Deschampsia caespitosa*, each with three primer pairs ***(15)***; *Peucedanum ostruthium*, two primer pairs (Borer *et al.*, unpubl. data). These studies included between 87 and 509 individuals (mean: 255) of either intra- and interspecific sampling; 4% to 51% (mean: 23%) of the individuals were replicated. We varied binning parameters (i.e. MinBin and MaxBin) and measured their effects on the width of bins, the datasets' polymorphism (i.e. the proportion of bins with presence frequencies ranging between 5% and 95%) and the bin reproducibility (i.e. the proportion of replicated sample pairs over which the focal bin is successfully reproduced; this measure is independent of the total number of bins in the dataset and is therefore suitable in the context of binning optimization).

MinBin and MaxBin acted as boundaries on the width of bins. They determined how accurately an AFLP signal was reflected into the final presence/absence matrix. Both parameters were explored for values ranging between 0.1 bp and 5 bp, with 0.1 bp increments, therefore totalizing 1176 “binning trials” per dataset (Fig 4. a, b).

Using exaggeratedly small MaxBin values forces the binning algorithm to define narrow bins. This situation leads to “oversplitting”, a bias where AFLP signals are coded into more bins than needed. In this case, an AFLP locus is coded using several adjacent bins appearing as inconsistent when considered independently. Our results showed oversplitting evidence for bin widths below 0.5 bp, with decreased bins' polymorphism and reproducibility (Fig 4. c, d). This situation should be avoided, and we recommend using values larger than 0.5 bp for MaxBin (in contrast, the MinBin parameter had little effects on oversplitting).

Using exaggeratedly large MinBin and MaxBin values introduces “technical homoplasy” [as defined in ***(4)***], a bias where AFLP signals are coded using less bins than required. Although merging artefactual secondary peaks with authentic peaks is desirable, a process that increases the consistency of binning (see above), exaggerated merging tends to artificially increase similarity between unrelated samples and has immediate effects on AFLP polymorphism and reproducibility. Our results showed that technical homoplasy was reflected by a decrease in bin polymorphism, when MinBin and MaxBin both exceeded 2 bp (i.e. corresponding to bin widths larger than 0.8 bp). On the other hand, reproducibility increased along with technical homoplasy due to the addition of some level of artefactual similarity among samples.

From these results, we suggest to screen binning parameters by considering bin polymorphism as a main optimization criterion (Fig 4. a, c). Reproducibility statistics should not be considered for binning optimization because of their inability to detect technical homoplasy (Fig 4. b, d). See Holland *et al.* ***(7)*** for further considerations about binning optimization.

Note 4

In its current version, RawGeno includes three filters that can be applied after binning has been achieved. These are a size (i.e. the region of electropherograms over which the analysis must be carried out), a fluorescence and a reproducibility filters. We applied filters to the 17 datasets explored during binning optimizations (see Note 3). The size filter was not tested and all datasets were analysed for bins ranging between 100 and 400 bp (i.e. corresponding to the electrophoresis region where size interpolation is accurate). The two remaining filters were explored starting from binned datasets that maximized polymorphism.

The first filter considers bins' quality to vary according to average fluorescence. Indeed, bins with an average fluorescence close to the detection threshold used in PeakScanner are more likely to reflect inconsistent signals (e.g. technical false absences) than bins with strong average fluorescence (see above). The filter removes bins according to a user-defined lower fluorescence limit. Keep in mind that this limit applies on normalized values. We tested limits ranging between 0 (i.e. no filtering) and 400 rfu (with 10 rfu increments) and measured polymorphism and reproducibility changes caused by this filtering (Fig 5. a, b). All but one dataset gained in polymorphism and reproducibility when filtering was optimized. However, while filtering increased dataset reproducibility, it drastically decreased polymorphism when set up improperly. Its use hence requires dataset-depending optimization to limit information reduction in datasets. We recommend testing fluorescence thresholds below 200 rfu.

The reproducibility filter assesses the robustness of AFLP signals and removes bins that are not satisfactorily reproducible. We tested filtering using bin reproducibility limits ranging between 0 and 100% (with 5% increments) and measured polymorphism and global error rate of datasets (Fig 5. c, d). We present error rates ***(13)*** instead of bin reproducibility here because error rates are more commonly reported in AFLP studies. Filtering effectively reduced the global error rate along with the removal of non-reproducible bins from datasets. As expected, filtering also decreased the polymorphism of datasets. Nevertheless, this filter generally removed numerous bins, and we clearly propose avoiding the use of reproducibility values larger than 95%. Our experience shows that using 85% as a threshold provides satisfactory results by balancing the error rates of datasets (i.e. 5% in average, which we consider reasonable) with polymorphism. A more stringent filtering can be applied (e.g. in studies where peak reproducibility is especially relevant such as in genome scans), but it might affect the information content of datasets and thus requires the inclusion of additional primer pairs to achieve a reasonable level of polymorphism.

5. Acknowledgments

We are grateful to Anahi Espindola, Matthias Borer and Aurélien Labhardt for kindly providing the AFLP datasets presented in Notes 2-4. We thank Nicolas Salamin for having provided a working environment to N. Arrigo during the redaction of this manuscript. Felix Gugerli, Sabine Brodbeck, Piya Kittipadakul, Julie B. Hébert, Julie Lee Yaw, Christian Parisod, Roland Dubillard and Robin Arnoux provided helpful comments during the development of RawGeno. This project was supported by the National Centre of Competence in Research (NCCR) Plant Survival and the PNR-59, two programs of the Swiss National Science Foundation.

6. References

1. Vos P, Hogers R, Bleeker M, Reijans M, van de Lee T, Hornes M, Frijters A, Pot J, Peleman J, Kuiper M, Zabeau M (1995) AFLP: a new technique for DNA fingerprinting. Nucleic Acids Res 23:4407–4414.
2. Pompanon F, Bonin A, Bellemain E, Taberlet P (2005) Genotyping errors: causes, consequences and solutions. Nat Rev Genet 6:847–859.
3. Meudt HM, Clarke AC (2007) Almost forgotten or latest practice? AFLP applications, analyses and advances. Trends Plant Sci 12(3):106-117.
4. Arrigo N, Tuszynski JW, Ehrich D, Gerdes T, Alvarez N (2009) Evaluating the impact of scoring parameters on the structure of intra-specific genetic variation using RawGeno, an R package for automating AFLP scoring. BMC Bioinformatics doi:10.1186/1471-2105-10-33.
5. Arrigo N, Felber F, Parisod C, Buerki S, Alvarez N, David J, Guadagnuolo R (in press) Origin and expansion of the allotetraploid *Aegilops geniculata*, a wild relative of wheat. New Phytol.
6. Whitlock R, Hipperson H, Mannarelli M, Butlin RK, Burke T (2008) An objective, rapid and reproducible method for scoring AFLP peak-height data that minimizes genotyping error. Mol Ecol Res 8:725-735.
7. Herrmann D, Poncet BN, Manel S, Rioux D, Gielly L, Taberlet P, Gugerli F (2010) Selection criteria for scoring amplified fragment length polymorphisms (AFLPs) positively affect the reliability of population genetic parameter estimates. Genome 53(4):302-310.
8. Holland BR, Clarke AC, Meudt HM (2008) Optimizing automated AFLP scoring parameters to improve phylogenetic resolution. Syst Biol 57(3):347-366.
9. Vekemans X, Beauwens T, Lemaire M, Roldan-Ruiz I (2002) Data from amplified fragment length polymorphism (AFLP) markers show indication of size homoplasy and of a relationship between degree of homoplasy and fragment size. Mol Ecol 11(1):139-151.
10. Paris M, Bonnes B, Ficetola GF, Poncet BN, Després L (2010) Amplified fragment length

homoplasy: in silico analysis for model and non-model species. BMC Genomics doi:10.1186/1471-2164-11-287.

11. Ehrich D (2006) AFLPdat: a collection of R functions for convenient handling of AFLP data. Mol Ecol Notes 6: 603-604.
12. Gort G, van Eeuwijk F (2010) Codominant scoring of AFLP in association panels. Theor Appl Genet doi:10.1007/s00122-010-1313-x.
13. Bonin A, Bellemain E, Eidesen PB, Pompanon F, Brochmann C, Taberlet P (2004) How to track and assess genotyping errors in population genetics studies. Mol Ecol 13:3261–3273.
14. Gugerli F, Englisch T, Niklfeld H, Tribsch A, Mirek Z, Ronikier M, Zimmermann NE, Holderegger R, Taberlet P, IntraBioDiv Consortium (2008) Relationships among levels of biodiversity and the relevance of intraspecific diversity in conservation - a project synopsis. Perspect Plant Ecol Evol Syst 10: 259-281.
15. Peintiger M, Arrigo N, Brodbeck S, Koller A, Imsand M Holderegger R (2010) Genetische und morphologische Differenzierung der endemischen Grasart *Deschampsia littoralis* (Gaudin) Reut. – Wie verschieden sind die Population am Bodensee und am Lac de Joux im Vergleich zu *D. cespitosa* (L.) P. Beauv.? Vogelwarte Radolfszell, University of Neuchâtel, WSL.

7. Figure captions

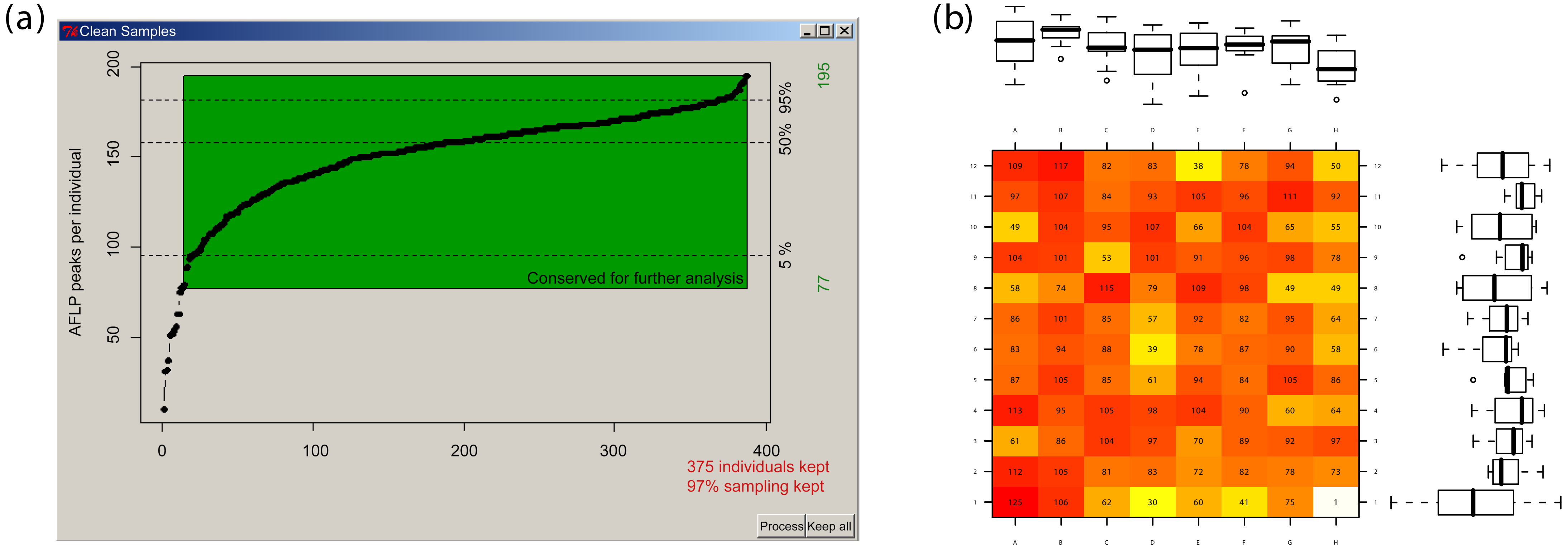
Fig 1. Checking quality of AFLP reactions. (a) Interactive device for filtering samples, before performing binning. Samples are sorted according to the number of AFLP peaks successfully amplified. Summary statistics (i.e. 5%, 50% and 95% quantiles) are provided to help users selecting samples to be included in further analyses. (b) Visualization of 96-well PCR plates. Four quality statistics are available in RawGeno (number of AFLP peaks per sample, mean and variance of fluorescence intensity and outlier detection index; see RawGeno manual). Boxplots provide summary statistics for rows and columns of PCR plates, respectively.

Fig 2. Binning algorithm implemented in RawGeno (from ***(4)***). (a) Left panel: main steps followed by the algorithm to define bins; right panel: illustration of binning with two samples (S1 and S2); the bin widths (i.e. the difference in size between the longest and the shortest amplicons included in the considered bin) and the technical homoplasy rates (i.e. HR, the mean number of peaks belonging to the same sample that are included in a same bin) are indicated. (b) Binning edition device, allowing users to translate, resize, add or remove bins interactively, starting from bins that were initially defined by the automated algorithm. Summary statistics (i.e. the average size of bins and the number of AFLP peaks present per bin) are provided as editing guidelines.

Fig 3. Reviewing results. RawGeno includes basic visualization devices for performing preliminary data mining. Specifically, results can be reviewed using (a) heatmaps of the binary matrix, where samples are sorted according to their genetic similarity and (b) principal coordinates analysis of the corresponding matrix. Both devices can compare AFLP results with either quality statistics (i.e. number of AFLP peaks per sample, mean and variance in fluorescence intensity and outlier detection index) or external information provided by users (e.g. the population from where samples were collected). In addition, both devices are handled through a graphical user interface for sorting and selecting samples to be visualized.

Fig 4. Binning parameters. We analysed 17 different datasets (see Note 3) in RawGeno, by varying binning parameters that have an effect on the width of bins (i.e. MinBin and MaxBin) and measured the associated effects on bin width, polymorphism and reproducibility. All results were corrected in order to normalize statistics among datasets (i.e. using non-linear mixed effect models, with the dataset origin considered as a covariable). Upper row: triangle plots displaying (a) bin polymorphism and (b) reproducibility averaged for the 17 datasets as a function of MinBin and MaxBin parameters. Lower row: scatterplots of corresponding (c) bin polymorphism and (d) bin reproducibility statistics, displayed according to the median bin width of binning trials. 5% and 95% confidence intervals (dashed lines) and the average (continuous line) are displayed. (a) to (d) Trials optimizing bin polymorphism are represented as dots for each of the 17 datasets (boxplots indicate bin widths associated to optimized trials).

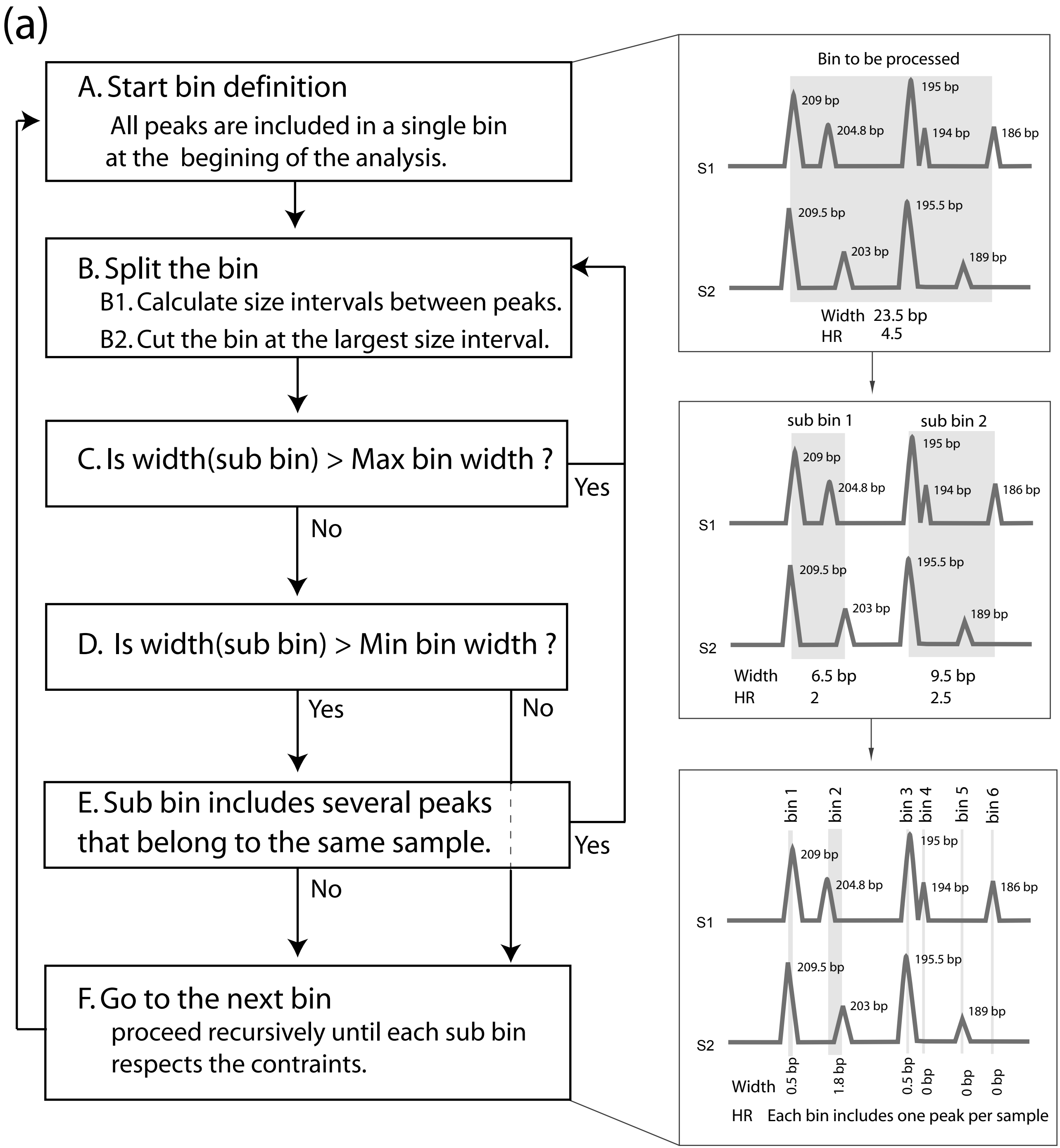
Fig 5. Filtering parameters. We filtered datasets (using optimal binning parameters as defined in Note 3) by increasing bin fluorescence and reproducibility thresholds. Median bin polymorphism and error rates were measured for the final datasets (dataset-specific offsets were not corrected). Upper row: bins are filtered according to their average fluorescence. Results are displayed as a comparison with non-filtered trials, with “D bin polymorphism” and “D bin reproducibility” being the difference between filtered and non-filtered datasets. Therefore, values greater than zero reflect cases where filtering increased the median (a) polymorphism or (b) reproducibility of bins in the final dataset. Conversely, values lower than zero indicate decreases in these statistics associated with filtering. Lower row: filtered bins according to their reproducibility and consequences on median bin (c) polymorphism and (d) error rate of the final datasets. Results of each dataset (grey lines) are displayed along with summary statistics (dashed lines: 5% and 95% confidence intervals; continuous line: average). (a) to (d) Trials optimizing bin polymorphism after filtering are presented as dots for each of the 17 datasets (boxplots indicate fluorescence threshold associated to optimized filtering).

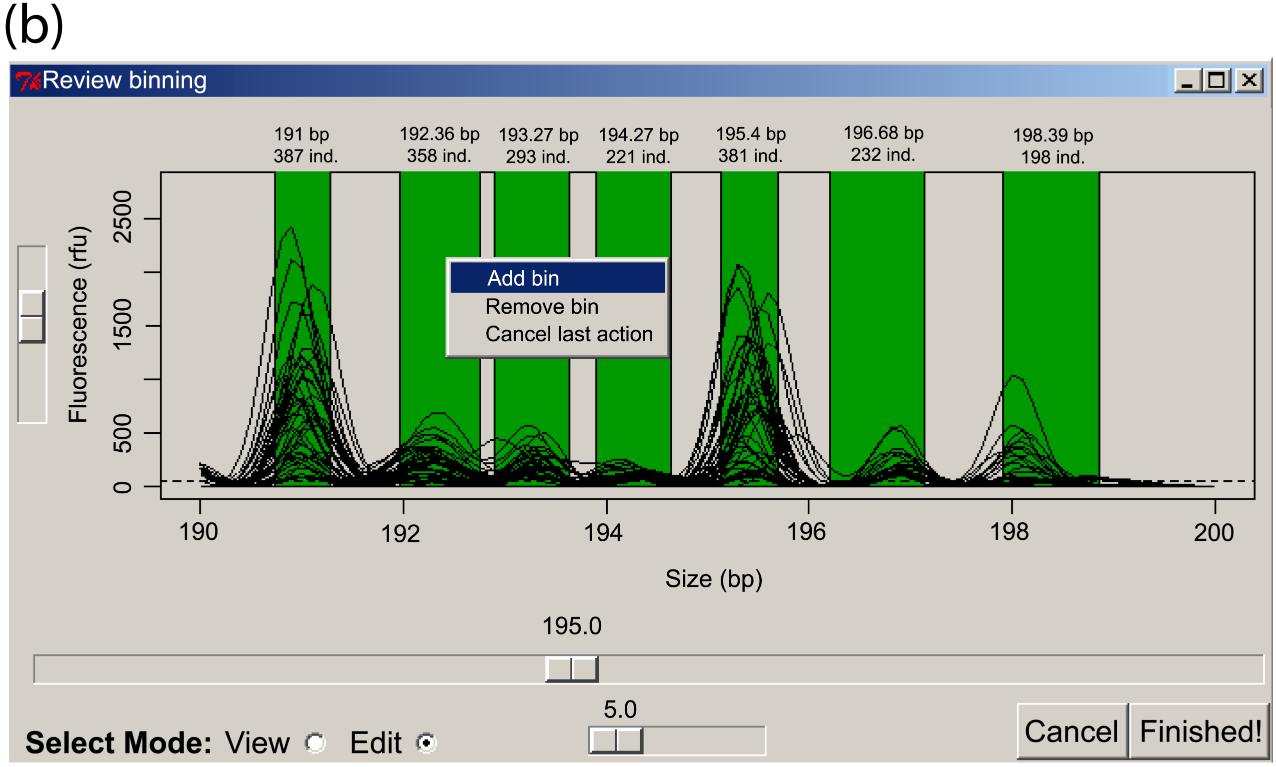


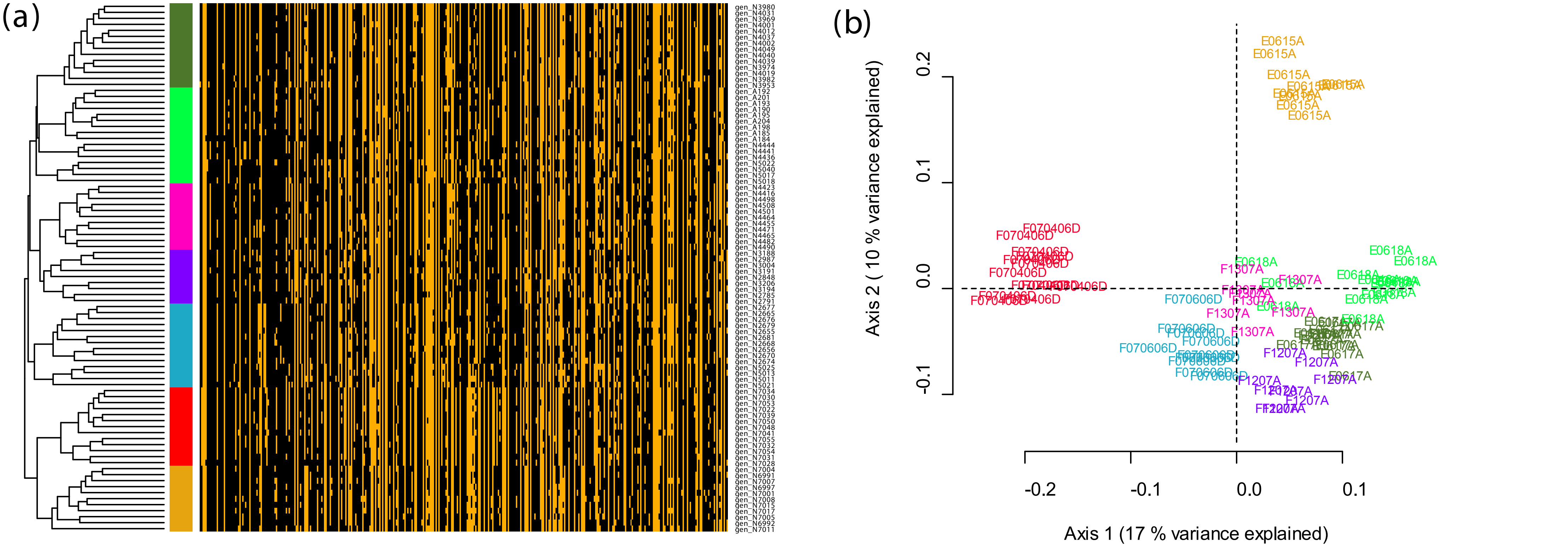
Conserved for further analysis

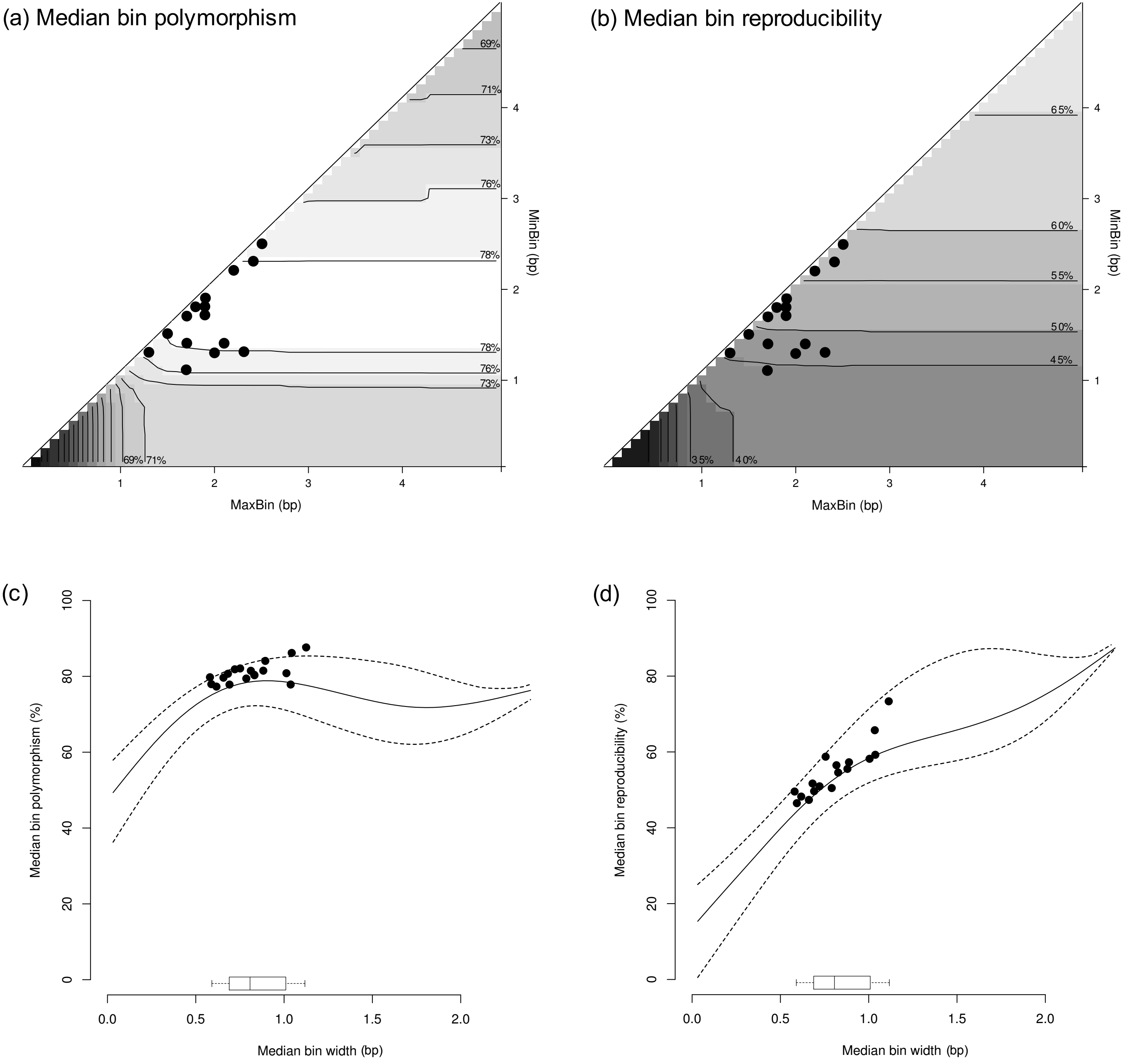
375 individuals kept
97% sampling kept

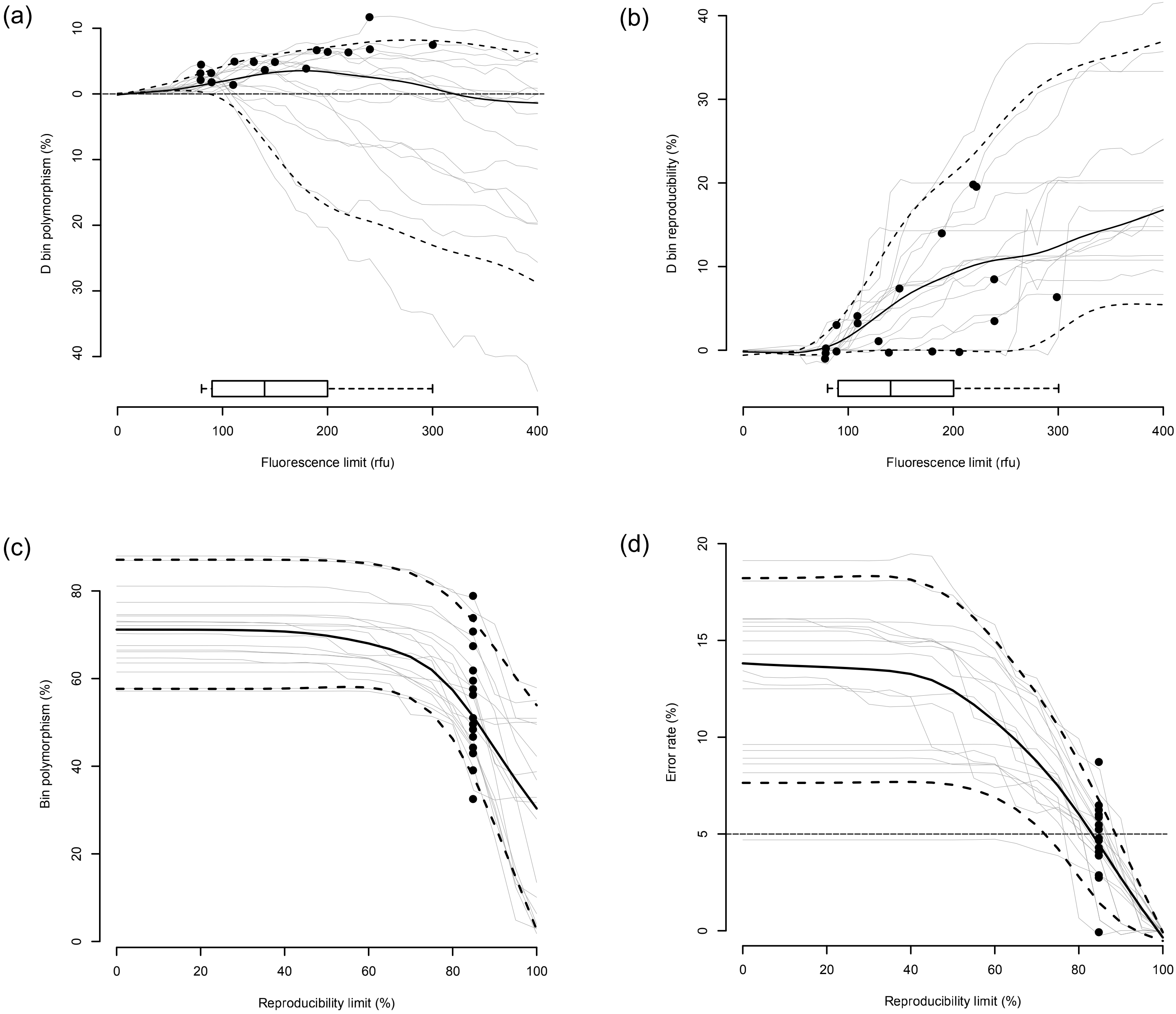
	A	B	C	D	E	F	G	H
12	109	117	82	83	38	78	94	50
11	97	107	84	93	105	96	111	92
10	49	104	95	107	66	104	65	55
9	104	101	53	101	91	96	98	78
8	58	74	115	79	109	98	49	49
7	86	101	85	57	92	82	95	64
6	83	94	88	39	78	87	90	58
5	87	105	85	61	94	84	105	86
4	113	95	105	98	104	90	60	64
3	61	86	104	97	70	89	92	97
2	112	105	81	83	72	82	78	73
1	125	106	62	30	60	41	75	1


(a)
A. Start bin definition
All peaks are included in a single bin at the begining of the analysis.
B. Split the bin
B1. Calculate size intervals between peaks.
B2. Cut the bin at the largest size interval.
C. Is width(sub bin) > Max bin width ?
Yes
No
D. Is width(sub bin) > Min bin width ?
Yes
No
E. Sub bin includes several peaks that belong to the same sample.
Yes
No
F. Go to the next bin
proceed recursively until each sub bin respects the contraints.
Bin to be processed
S1
209 bp
204.8 bp
195 bp
194 bp
186 bp
S2
209.5 bp
203 bp
195.5 bp
189 bp
Width 23.5 bp
HR 4.5
sub bin 1
sub bin 2
S1
209 bp
204.8 bp
195 bp
194 bp
186 bp
S2
209.5 bp
203 bp
195.5 bp
189 bp
Width 6.5 bp 9.5 bp
HR 2 2.5
bin 1
bin 2
bin 3
bin 4
bin 5
bin 6
S1
209 bp
204.8 bp
195 bp
194 bp
186 bp
S2
209.5 bp
203 bp
195.5 bp
189 bp
Width 0.5 bp 1.8 bp 0.5 bp 0 bp 0 bp 0 bp
HR Each bin includes one peak per sample


(b)
Review binning
191 bp 387 ind.
192.36 bp 358 ind.
193.27 bp 293 ind.
194.27 bp 221 ind.
195.4 bp 381 ind.
196.68 bp 232 ind.
198.39 bp 198 ind.
Add bin
Remove bin
Cancel last action
Fluorescence (rfu)
0
500
1500
2500
190
192
194
196
198
200
Size (bp)
195.0
5.0
Select Mode: View ○ Edit ●
Cancel
Finished!


(a)
gen_N3980
gen_N4031
gen_N3969
gen_N4001
gen_N4012
gen_N4037
gen_N4002
gen_N4049
gen_N4040
gen_N4039
gen_N3974
gen_N4019
gen_N3982
gen_N3953
gen_A192
gen_A201
gen_A193
gen_A190
gen_A195
gen_A204
gen_A198
gen_A185
gen_A184
gen_N4444
gen_N4441
gen_N4436
gen_N5022
gen_N5040
gen_N5017
gen_N5018
gen_N4423
gen_N4416
gen_N4498
gen_N4508
gen_N4501
gen_N4464
gen_N4455
gen_N4471
gen_N4465
gen_N4482
gen_N4490
gen_N3188
gen_N2987
gen_N3004
gen_N3191
gen_N2848
gen_N3206
gen_N3194
gen_N2785
gen_N2791
gen_N2677
gen_N2665
gen_N2676
gen_N2679
gen_N2655
gen_N2681
gen_N2668
gen_N2656
gen_N2670
gen_N2674
gen_N5025
gen_N5013
gen_N5011
gen_N5021
gen_N7034
gen_N7030
gen_N7053
gen_N7022
gen_N7039
gen_N7050
gen_N7048
gen_N7041
gen_N7055
gen_N7032
gen_N7054
gen_N7031
gen_N7028
gen_N7004
gen_N6991
gen_N7007
gen_N6997
gen_N7001
gen_N7008
gen_N7015
gen_N7017
gen_N7005
gen_N6992
gen_N7011
(b)
Axis 2 (10 % variance explained)
0.2
0.1
0.0
-0.1
Axis 1 (17 % variance explained)
-0.2
-0.1
0.0
0.1
E0615A
F070406D
E0618A
F1307A
F070606D
E0617A
F1207A


(a) Median bin polymorphism
69%
71%
73%
76%
78%
78%
76%
73%
69%
71%
MaxBin (bp)
MinBin (bp)
1
2
3
4
(b) Median bin reproducibility
65%
60%
55%
50%
45%
35%
40%
MaxBin (bp)
MinBin (bp)
(c)
Median bin polymorphism (%)
0
20
40
60
80
100
Median bin width (bp)
0.0
0.5
1.0
1.5
2.0
(d)
Median bin reproducibility (%)
Median bin width (bp)


(a)
D bin polymorphism (%)
10
0
10
20
30
40
0
100
200
300
400
Fluorescence limit (rfu)
(b)
D bin reproducibility (%)
40
30
20
10
0
0
100
200
300
400
Fluorescence limit (rfu)
(c)
Bin polymorphism (%)
80
60
40
20
0
0
20
40
60
80
100
Reproducibility limit (%)
(d)
Error rate (%)
20
15
10
5
0
0
20
40
60
80
100
Reproducibility limit (%)