Supplementary material

Supplementary methods. Manual of Quantification of Representative Sequences (QRS)

Name of the program and author

Quantification of Representative Sequences (QRS.pl)

Enrique González-Tortuero <email: gonzalez@igb-berlin.de>

Brief description

QRS is a script written in Perl 5.14.2 that allows analysing 454 pyrosequencing platform sequences automatically to study population structure. This program may operate either in batch processing mode where all sequences are automatically analysed in an unsupervised way or it may interact with the user at various checkpoints if no parameters have been specified prior to execution. The program is freely available at https://code.google.com/p/quantification-representative-sequences under GPLv3 licence.

Requirements

Before using this pipeline, the following Perl modules and programs should be installed:

- Perl modules:
  - BioPerl (Bio::AlignIO and Bio::SeqIO) (Stajich *et al.* 2002)
  - File::Find::Rule
  - File::Which
  - JSON
  - String::Approx
  - Sys::CPU
  - Sys::MemInfo
- Programs:
  - PrinSeq (Schmieder & Edwards 2011): it is used to generate summary statistics of sequence and quality data and to filter, reformat and trim 454 data. This program is freely available at http://prinseq.sourceforge.net under the GPLv3 licence.
  - CutAdapt (Martin 2011): a Python script that removes primers and adapters. This program is publicly available at http://code.google.com/p/cutadapt under the MIT licence.
  - HMMER 3.1b (Finn *et al.* 2011): it is used to search sequences using probabilistic models called profile hidden Markov models (profile HMM). This program is freely available at http://hmmer.janelia.org/ under GPLv3 licence.
  - USEARCH (Edgar 2010): it is used to cluster sequences and detect chimaeras according

to the UCHIME algorithm (Edgar *et al.* 2011). This program is available after requiring it according to the authors' page. (http://www.drive5.com/usearch/).

- At least one of the following aligners:
  - Clustal Omega (Sievers *et al.* 2011): a hybrid aligner that mixes progressive multiple sequence alignments with the use of Markov models. This program is freely available at http://www.clustal.org/omega/ under GPLv2 licence.
  - FSA (Bradley *et al.* 2009): a probabilistic aligner that uses the sequence annealing technique (Schwartz & Pachter 2007) for constructing a multiple alignment from pairwise homology estimation. FSA is publicly available at http://fsa.sourceforge.net/ under GPL licence.
  - GramAlign (Russell *et al.* 2008): a time-efficient progressive aligner which estimates distances according to the natural grammar present in nucleotide sequences. This program is publicly available at http://bioinfo.unl.edu/gramalign.php
  - KAlign (Lassmann & Sonnhammer 2005): a progressive aligner based on Wu-Manber string-matching algorithm. KAlign is publicly available at http://msa.sbc.su.se under GPLv2 licence.
  - MAFFT (Katoh & Standley 2013): an iterative/progressive alignment program that is publicly available at http://mafft.cbrc.jp/alignment/software/ under BSD licence.
  - MUSCLE (Edgar 2004): an iterative aligner that it is available at http://www.drive5.com/muscle/.
  - PRANK (Löytynoja & Goldman 2005): a probabilistic multiple alignment program based on maximum likelihood methods used in phylogenetics that considers evolutionary distances between sequences. This program is publicly available at http://code.google.com/p/prank-msa under GPLv3 licence.
- TCS (Clement *et al.* 2000): a Java computer program to estimate phylogenetical networks and intraspecific genealogies according to statistical parsimony (Templeton *et al.* 1992). This program is freely available at http://darwin.uvigo.es/software/tcs.html under GPLv2 licence.

Although the aforementioned aligners have already been tested with QRS, other aligners may be added to use in this pipeline. If you want to work with another alignment program, please feel free to contact the author (gonzalez@igb-berlin.de) to include it in the source code of QRS.

Finally, an optional recommended requirement might be the installation of ReformAlign (Lyras & Metzler 2014). ReformAlign is a recently proposed profile-based meta-alignment approach that aims to fine-tune existing alignments via the employment of standard profiles. This program is

available at http://evol.bio.lmu.de/_statgen/software/reformalign/ under GPLv3 licence.

Installation of QRS

All installation procedures described here were tested in GNU/Linux OS, using the Ubuntu 13.10 distribution. When using a Windows operating system, QRS should work after installing Cygwin. Before executing QRS, install all Perl modules and required programs according to their manual instructions. We recommend installing the latest stable version for all programs.

Then, download the QRS source code and give permission to the Perl script to be auto-executable. Copy the script to the bin folder:

```
chmod +x QRS.pl
sudo cp QRS.pl  /usr/local/bin
```

Usage

If QRS is working in batch mode, then all QRS required parameters have to be defined prior to execution (please see below and table S1 for a detailed description of those parameters).

There are two general options that must be included in all analyses: i) the pipeline step option (defined as **-RM**, **-AM1**, **-AM2** or **AM3**); ii) and the folder containing all required files (**-folder**).

1) Creating a HMM reference file (**-RM** option): in this step, the program aligns your reference dataset/s and calls HMMER 3.1b to generate a Markov model.

To run this step in batch mode, the required parameters are as follows:

-reffiles=<Name of the FASTA files>: This parameter is used to indicate what the reference FASTA files are, i.e. FASTA files with sequences of specific marker/s. If you want to analyse more than one marker, all file names have to be separated by a single dash '-' (e.g. file1-file2-file3).

-aligner=<Name of the alignment program>: this option specifies which program you want to use to align your sequences. The program can use one of the following aligners:

- ClustalOmega (**-aligner=clustalo**)
- FSA (**-aligner=fsa**)
- GramAlign (**-aligner=gramalign**)
- KAlign (**-aligner=kalign**)
- MAFFT:
    - **-aligner=mafft-accurate** if you want a precise alignment based on G-INS-i (iterative global alignment method; see Katoh *et al.* (2005)).

- **-aligner=mafft-fast** if you want to perform an alignment based on FFT-NS-2 (progressive alignment algorithm; see Katoh et al. (2002)). This is recommended for more than 200 sequences.

- MUSCLE (**-aligner=muscle**)
- PRANK (**-aligner=prank**)

By default, QRS uses PRANK.

-reformal=[yes|no]: this argument is used to specify if you want to use ReformAlign to fine-tuning your alignment. By default, QRS has this option turned on to do it but since ReformAlign cannot currently handle ambiguous characters, this switch should be off if your data contains Ns.

2) Describing the 454 data set (**-AM1** option): 454 data files are processed in PrinSeq to calculate basic statistics about length distribution, GC content distribution, base quality distribution, occurrence of bad nucleotides (Ns), presence of poly-A/T tails, tag sequence check, sequence duplication, sequence complexity and dinucleotide odd-ratios. All these measures are important to set the best parameters in future analyses (described in the "processing a 454 data set (-AM2 option)" step).

The result of this step is a HTML file and a folder with all pertinent graphs to study the quality control. The quality control can be examined as described in the PrinSeq manual (for more information see http://prinseq.sourceforge.net/manual.html#QC).

To run this step in batch mode, the accepted parameters are as follows:

- Required parameters:

  -informat=[fasta|fastq]: this argument indicates the type of input sequence.

  -fasta=<Name of the FASTA file>: this option indicates the name of the FASTA file. The option is mandatory if the input is a FASTA file (**-informat=fasta**).

  -fastq=<Name of the FASTQ file>: this parameter indicates the name of the FASTQ file. This option is mandatory if the input is a FASTQ file (**-informat=fastq**).

  -fastq2=<Name of the FASTQ file>: this option indicates the name of the paired FASTQ file. This option is only mandatory if paired-end data are used (**-paired=yes**).

- Optional parameters:

  -quality=<Name of the quality file>: this argument indicates the name of the quality file. This file can be used with a FASTA file as input (**-informat=fasta**).

  -paired=[yes|no]: this option is only defined if a FASTQ file is used as a input file

(**-informat=fastq**) and indicates if a paired-end FASTQ is used or not. By default it is disabled (**-paired=no**).

-phred64=[yes|no]: this parameter indicates if the quality data of the FASTQ files (**-informat=fastq**) have Phred+64 format (used for Illumina 1.3 – 1.7). By default it is disabled (**-phred64=no**).

3) Processing a 454 data set (**-AM2** option): in this step, a 454 FASTA or FASTQ file is processed to filter all sequences according to length, mean quality of each sequence and complexity of the sequences and to trim poly-A tails and bad quality nucleotides (Ns). After that, all these sequences are filtered according to a specific marker using a Markov model (see "Creating a reference HMM file (**-RM** option)" for more details). Later, accepted sequences are split into different FASTA files according to the information of the *design* and *oligos* files (see **-design** and **-oligos** options to know more) and the primers for all sequences using CutAdapt are removed. Then, the 454 data set is clustered at the level of 100% identity and de-noised using USEARCH according to the CD-HIT-OTU de-noising method unless a specific cutoff is defined using the parameter -**cutoff**. Later, USEARCH is called again to remove all chimaeras using the UCHIME algorithm. Finally, these sequences are aligned using a multiple sequence alignment program (see **-aligner** option for more details).

To run this step in batch mode, the accepted parameters are as following:

- Required parameters:

  -informat=[fasta|fastq]: this argument indicates the type of input sequence: a FASTA (**-informat=fasta**) or a FASTQ file (**-informat=fastq**).

  -fasta=<Name of the FASTA file>: this option indicates the name of the FASTA file (**-informat=fasta**).

  -fastq=<Name of the FASTQ file>: this parameter indicates the name of the FASTQ file (**-informat=fastq**).

  -fastq2=<Name of the FASTQ file>: this argument indicates the name of the paired FASTQ file. This option is only mandatory if it paired-end data are used (**-paired=yes**).

  -**hmmfile**=<Name of the HMM file>: this option indicates the name of the reference HMM file (or Markov model). This HMM can be created in "creating a reference HMM file (**-RM** option)" step.

  -oligos=<Namefile>: The name of the *oligos* file. This file is a plain text file that has two columns separated by tab character (except if the label is barcode, where there

are a third column) and the last value should be followed by a newline character. The first column contains always a label that indicates forward or reverse adapter, forward or reverse primers and barcode (`seqadapfor`, `seqadaprev`, `forward`, `reverse` and `barcode` respectively). The second column contains the DNA sequence for each element and, in the case of barcode label, the third one indicates the barcode ID (MID1, MID2, MID3...). An example of this file is as follows:

```
seqadapfor      SEQUENCE
seqadaprev      SEQUENCE
forward   SEQUENCE
reverse   SEQUENCE
barcode   SEQUENCE  BARID
```

This option is mandatory if the **-nosplit** option is not indicated.

-design=<Namefile>: The name of the *design* file. This file is a plain text file that has two or three columns (depending of the use of reverse barcoded primers) separated by tab characters and the last value should be followed by a newline character. An example of these file is as follows:

```
BARID1    (BARID2)  SAMPLEID
```

Here, BARID1 is the Barcode ID for the forward primer, BARID2 is the Barcode ID for the reverse primer (if exists) and SAMPLEID is the name of the sample. This option is mandatory if the **-nosplit** option is not indicated.

-brn or **-bry**: this parameter indicates presence (**-bry**) or absence (**-brn**) of barcoded reverse primers. It is mandatory if the **-nosplit** option is not specified.

- Optional parameters:

  -quality=<Name of the quality file>: this parameter indicates the name of the quality file and can be used along with a FASTA file as input (**-informat=fasta**).

  -paired=[yes|no]: this option can only be defined if a FASTQ file is used as a input file (**-informat=fastq**). The parameter indicates if a paired-end FASTQ is used or not. By default it is disabled (**-paired=no**).

  -phred64=[yes|no]: this parameter indicates if the quality data in the FASTQ files (**-informat=fastq**) have Phred+64 format (as for Illumina 1.3 – 1.7). By default it is

disabled (**-phred64=no**).

-**minlen**=<Number>: this option indicates the minimum sequence length which is allowed to pass the filters. It should be defined according to the basic statistics of the analysed data set (see "Describing the 454 data set (**-AM1** option)").

-**maxlen**=<Number>: this argument indicates the maximum sequence length which is allowed to pass the filters. It should be defined according to the basic statistics of the analysed data set (see "Describing the 454 data set (**-AM1** option)").

-**minqual**=<Number>: this parameter indicates the minimum mean quality which is allowed to pass the filters. It should be defined according to the basic statistics of the analysed data set (see "Describing the 454 data set (**-AM1** option)") and when a quality or a FASTQ file is used.

-**trimqual**=<Number>: this option is used to allow trimming of the sequences flanks by "bad quality nucleotides". The number indicates the minimum quality that is accepted for the beginning and the end of the sequences. It should be defined according to the basic statistics of the analysed data set (see "Describing the 454 data set (**-AM1** option)") and when a quality or a FASTQ file is used.

-**trimtails**=<Number>: this argument is used to trim poly-nucleotide tails at the beginning or the end of the sequences. It should be defined according to the basic statistics of the analysed data set (see "Describing the 454 data set (**-AM1** option)").

-**allowns**=<Number>: this parameter is used to define the maximum undefined nucleotides (Ns) allowed. We do not recommend to allow Ns (**-allowns=0**). However, the parameter can be defined according to the basic statistics of the analysed data set (see "Describing the 454 data set (**-AM1** option)").

-**filmet**=<Method>: this option is used to specify the filtering method to remove sequences with low complexity, i.e. sequences stretches that consist of a repeated tandem of nucleotides (e.g. $(ATCG)_{20}$ or $(GGTC)_{10}$). These sequences are considered as sequencing artefacts. To detect them, QRS can use entropy (**-filmet=entropy**) or DUST (**-filmet=dust**) methods (Morgulis *et al.* 2006). The parameter should be defined according to the basic statistics of the analysed data set (see "Describing the 454 data set (**-AM1** option)").

-**filthr**=<Number>: this option defines the threshold for the filtering method. It is mandatory to put this option after defining the filtering method (**-filmet=[dust|entropy]**). The parameter should be defined according to the basic statistics of the analysed data set (see "Describing the 454 data set (**-AM1** option)").

-**hmmthr**=<Number>: This option defines the maximum allowed *e*-value to consider

a sequence as a specific marker using a Markov model. By default, it is 10^{-10} (**-hmmthr=1E-10**) and we do not recommend to modify this value.

-nosplit: this parameter can be used when all primers of your data set has been trimmed (e.g., when using FASTQ files from SRA NCBI). By default, this option is disabled.

-**allowmis**=<Exact number or percent of mismatches>: this argument indicates the maximum number of mismatches allowed to detect the barcodes using CutAdapt. By default, the value is set to 1% (**-allowmis=1%**).

-**cutoff**=<Number>: this option indicates the clustering identity threshold to de-noise the data set. By default, QRS calculates this value according to the CDHIT-OTU algorithm, i.e. this program considers that one or two mismatches (including gaps) in a pairwise comparison indicate that are the same sequence. If you put a value here, it must be a positive real number ranging between 0.00 and 1.00.

-**minclustersize**=<Number>: this parameter indicates the minimum cluster size that is accepted to assume that the analysed data set contains no artefacts. By default, QRS considers all clusters that have at least three sequences as valid clusters (**-minclustersize=3**).

-aligner=<Name of the alignment program> and **-reformal**=[Yes|No]: see details above in the "1) Creating a HMM reference file (-RM option)" section.

4) Classifying all 454 sequences into TCS-types (**-AM3** option): in this step, your aligned data set is classified into different TCS-types. By default, sequences with at most three differences are joined.

To run this step in batch mode, the accepted parameters are as following:

- Required parameters:

  -**fasta**=<Name of the FASTA file>: this parameter indicates the name of the alignment file in FASTA format.

  -**outfile**=<Name of the files>: this argument indicates the output file name. The extension (i.e. csv, txt, etc.) should not be added as QRS automatically adds the correct file extension.

  -**sample**=<Name of the file>: this option indicates the *sample* file which is a plain text file that contains the following information always separated by tab characters and the last value should be followed by a newline character:

```
SAMPLEID  DATA1     (…)  DATAN
```

Here, SAMPLEID is the name of the sample and DATA are data you want to put here (place, year, species...).

- Optional parameters:

  -**nrsteps**=<Number>: this parameter indicates the maximum number of steps you want to use to determine the TCS-types. By default, the number of maximum steps is three (**-nrsteps=3**).

Examples for batch mode runs

The program QRS can be executed as batch command or as an interactive program if no parameters are defined. To understand how QRS works in batch mode, we provide several use-case scenarios.

RM. Creating a reference HMM file (-RM)

RM.1. Basic parameters

When this option is used, the batch command can only be executed with basic parameters:

```
QRS.pl -RM -folder=test -reffiles=test.fasta
```

In this case, we call QRS to create a reference HMM file of `test.fasta` in the folder `test` using default parameters (the aligner is PRANK and QRS will call ReformAlign to fine-tune the alignment).

However, QRS can create more than a single HMM file if you specify different reference files joined with a single dash ('-') in the `-reffiles` parameter, like in the following example, where the program will create two HMM files using default parameters:

```
QRS.pl -RM -folder=test -reffiles=test1.fasta-test2.fasta
```

RM.2. Alignment options

Another option that you can modify is the multiple sequence alignment program. You can choose the program with the `-aligner` parameter. PRANK is the default aligner program, like in this example where QRS will create a single reference HMM file:

```
QRS.pl -RM -folder=test -aligner=prank -reffiles=test.fasta
```

To employ a different aligner for the sequence alignment task simply modify the `-aligner` parameter. In the following example, QRS will call MUSCLE to align a reference file to create a single reference HMM file:

```
QRS.pl -RM -folder=test -aligner=muscle -reffiles=test.fasta
```

Here, we use MAFFT with an iterative global alignment algorithm called G-INS-i to perform an accurate alignment:

```
QRS.pl -RM -folder=test -aligner=mafft-accurate -reffiles=test.fasta
```

The last option you may modify is whether to use ReformAlign to fine-tune the alignment or not. By default, this option is enabled (`-reformal=yes`), like in the following example, where QRS will create two reference HMM files after calling GramAlign to align your reference sequences:

```
QRS.pl -RM -folder=test -aligner=gramalign -reformal=yes -reffiles=test1.fasta-test2.fasta
```

If you want to disable this option, you have to type `-reformal=no`. In this example, QRS will align your three reference files using the default aligner (i.e., PRANK) and will not curate the alignments before creating three reference HMM files:

```
QRS.pl -RM -folder=test -reformal=no -reffiles=test1.fasta-test2.fasta-test3.fasta
```

In the following example, we call QRS to align four reference files using MAFFT in the accurate mode and we do not want to fine-tune the alignments:

```
QRS.pl -RM -folder=test -aligner=mafft-accurate -reformal=no -reffiles=test1.fasta-test2.fasta-test3.fasta-test4.fasta
```

AM1. Describing the 454 data set (-AM1)

AM1.1. Basic parameters

When this option is used, the batch command cannot be executed using only basic parameters. Instead, you have to specify if you have a 454 FASTA file or a 454 FASTQ file. In this example the input file is a FASTA file:

```
QRS.pl -AM1 -folder=test -informat=fasta -fasta=file.fna
```

Here, the input file is a FASTQ file:

```
QRS.pl -AM1 -folder=test -informat=fastq -fastq=file.fastq
```

AM1.2 Another input files

If you have a FASTA file as input, you can add a quality file to do all basic statistics on the base quality using the parameter `-quality`:

```
QRS.pl -AM1 -folder=test -informat=fasta -fasta=file.fna
-quality=file.qual
```

If you have a paired FASTQ file, you have to add the parameter `-paired=yes` and the name of the paired FASTQ:

```
QRS.pl -AM1 -folder=test -informat=fastq -fastq=file.fastq
-paired=yes -fastq2=file2.fastq
```

AM2. Processing a 454 data set (-AM2)

AM2.1. Basic parameters

Before executing this part, you have to evaluate the characteristics of your data set according to the previous step (see AM1 section). As all 454 pyrosequencing data sets are different depending on the case study, there are no default parameters to filter and trim sequences according to length, quality and complexity.

However, in a hypothetical case that you don´t need to filter your data set, the input parameters are the input files added as describe before (see AM1 section). Moreover, you have to add the reference HMM file (created in -RM step, see RM section for more details), *oligos* and *design* files (see "Processing a 454 data set (-AM2)" in Usage to know more about these files) and specify if you have re-

verse barcoded primers or not. In the following example, we use a 454 FASTA file with quality file and our data set has reverse barcoded primers (`-bry`):

```
QRS.pl -AM2 -folder=test -informat=fasta -fasta=file.fna
-quality=file.qual -hmmfile=test.hmm -oligos=oligos.csv
-design=design.csv -bry
```

AM2.2. Filtering by length

As it is said in AM2.1 section, there are no default parameters to filter and trim sequences according to length, quality and complexity because it depends of your data set. In the following example calls QRS to accept sequences that are greater than 300 bp in a 454 FASTQ file and have no reverse barcoded primers:

```
QRS.pl -AM2 -folder=test -informat=fastq -fastq=file.fastq -hm-
mfile=test.hmm -minlen=300 -oligos=oligos.csv -design=design.csv
-brn
```

In this example, QRS is executed to accept sequences that have between 355 and 500 bp (the data is given as 454 FASTA file without quality file):

```
QRS.pl -AM2 -folder=test -informat=fasta -fasta=file.fna
-hmmfile=test.hmm -minlen=355 -maxlen=500 -oligos=oligos.csv
-design=design.csv -bry
```

AM2.2. Filtering and trimming according to quality

We offer another filter based on base quality. In this example, QRS accepts only sequences that have at least a mean sequence quality value of 25:

```
QRS.pl -AM2 -folder=test -informat=fastq -fastq=file.fastq -hm-
mfile=test.hmm -minqual=25 -oligos=oligos.csv -design=design.csv
-bry
```

If you have a FASTA file, it is convenient to have a quality file to filter by base quality. In the following example, QRS will filter an input 454 FASTA file by length (accepting only sequences that are between 375 and 480 bp long) and quality (rejecting sequences that have a mean sequence quality score smaller than 28). This data set has no reverse barcoded primers:

```
QRS.pl -AM2 -folder=test -informat=fasta -fasta=file.fna -quality=file.qual -hmmfile=test.hmm -oligos=oligos.csv -design=design.csv -minlen=375 –maxlen=480 -minqual=28 -brn
```

The following option is used to trim bases that have an insufficient quality score. In the following example, QRS will trim all nucleotides with a quality value less than 25 in the beginning and at the end of the sequences in a non-reverse barcoded 454 data set:

```
QRS.pl -AM2 -folder=test -informat=fasta -fasta=file.fna -quality=file.qual -hmmfile=test.hmm -oligos=oligos.csv -design=design.csv -trimqual=25 -brn
```

AM2.3. Trimming poly-A/Ts tails

It is feasible to trim poly-A/Ts tails. In this example, QRS trims the regions that have more than three followed A/T at the beginning or the end of the sequences:

```
QRS.pl -AM2 -folder=test -informat=fastq -FASTQ=file.fastq -hmmfile=test.hmm –trimtails=3 -oligos=oligos.csv -design=design.csv -bry
```

AM2.4. Filtering by ambiguous characters

The number of allowed ambiguous characters in your data set can be modified with the parameter `-allowns`. In the following example, QRS accepts all sequences that have at most one ambiguous character in the sequence:

```
QRS.pl -AM2 -folder=folder -informat=fastq -fastq=file.fastq -hmmfile=ref.hmm -oligos=oligos.csv -design=design.csv -allowns=1 -bry
```

However, it is not recommended to allow ambiguous characters as they might indicate bad quality nucleotides (Huse *et al.* 2007). In the following example, QRS removes all sequences that have more than one ambiguous character:

```
QRS.pl -AM2 -folder=folder -informat=fasta -fasta=file.fna -hmmfile=ref.hmm -oligos=oligos.csv -design=design.csv -allowns=0
```

`-bry`

AM2.5. Filtering by complexity

Finally, if you want to remove all low complexity sequences like homopolymers, you can do this with `-filmet` and `-filthr` parameters. The first argument defines the method to remove this kind of sequences (DUST (Morgulis *et al.* 2006) or Entropy-based filter) and the second argument defines the threshold for the filtering method. For more details on these filtering methods, see the PrinSeq manual (http://prinseq.sourceforge.net/manual.html#QCCOMPLEXITY). In the following example, we use DUST to remove all sequences with a complexity greater than 7:

```
QRS.pl -AM2 -folder=folder -informat=fasta -fasta=file.fna -hmmfile=ref.hmm -oligos=oligos.csv -design=design.csv -filmet=dust -filthr=7 -bry
```

In this example, we use entropy to remove all sequences with a complexity smaller than 70:

```
QRS.pl -AM2 -folder=folder -informat=FASTA -FASTA=file.fna -hmmfile=ref.hmm -oligos=oligos.csv -design=design.csv -filmet=entropy -filthr=70 -bry
```

In this example, a 454 data set (that consists in a FASTQ file) is filtered by length (accepting only sequences that are between 390 and 500 bp long), base quality (removing all sequences with mean sequence quality score less than 28), and low-complexity based on the DUST filter (considering all sequences with values greater than 5 as low complexity sequences):

```
QRS.pl -AM2 -folder=folder -informat=fasta -fasta=file.fna -quality=file.qual -hmmfile=ref.hmm -oligos=oligos.csv -design=design.csv -minlen=390 -maxlen=500 -minqual=28 -filmet=dust -filthr=5 -bry
```

AM2.6. Classifying sequences as a specific marker

You can also modify the maximum allowed *e*-value to classify a sequence as a specific marker using a HMM profile (`-hmmthr`). By default, this argument is set to 10^{-10} as then similar results as with the usage of BLAST are achieved (Altschul *et al.* 1990). **We do not recommend changing this value**. However, you can change the value like in the following example where QRS classifies

a sequence as a specific marker if the *e*-value is smaller than 10^{-25}:

```
QRS.pl -AM2 -folder=folder -informat=fasta -fasta=file.fna -hmmfile=ref.hmm -oligos=oligos.csv -design=design.csv -hmmthr=1E-25 -bry
```

AM2.7. Assigning sequences to samples

If you want to modify the maximum number of allowed mismatches to detect the barcodes in your data set, you can use the parameter `-allowmis`. In this example, QRS classifies a non reverse barcoded data set in different samples with an exact match of the barcodes:

```
QRS.pl -AM2 -folder=folder -informat=fasta -fasta=file.fna -hmmfile=ref.hmm -oligos=oligos.csv -design=design.csv -allowmis=0 -brn
```

You can define a percentage to define the maximum value for mismatches. By default, we consider a maximum percentage of mismatches of 1% to detect the barcodes. Although we do not recommend modifying this value, it is possible to change it like in this example, where QRS considers a maximum percentage of 2% to detect the barcodes in a reverse barcoded data set:

```
QRS.pl -AM2 -folder=folder -informat=FASTQ -FASTQ=file.FASTQ -hmmfile=ref.hmm -oligos=oligos.csv -design=design.csv -allowmis=2% -bry
```

AM2.8. Clustering step

Another parameter you can modify is `-cutoff`. This parameter is used to cluster sequences in order to de-noise your data set, i. e., to remove all spurious nucleotides across all sequences. By default, QRS calculates this value according to the CD-HIT-OTU algorithm (Li *et al.* 2012) but you can define the value by a number between 0.00 and 1.00. For example, if you want to cluster all sequences that have 99.5% of similarity, you can type:

```
QRS.pl -AM2 -folder=folder -informat=fasta -fasta=file.fna -hmmfile=ref.hmm -oligos=oligos.csv -design=design.csv -cutoff=0.995 -brn
```

The parameter -minclustersize specifies the minimum size of cluster to consider that analysed sequences are not sequencing artifacts. By default, this filter is enabled and all clusters that have at least three sequences are considered as good clusters. If you want to disable this argument, you have to type -minclustersize=0, like in this example:

```
QRS.pl -AM2 -folder=folder -informat=fasta -fasta=file.fna -hmmfile=ref.hmm -oligos=oligos.csv -design=design.csv -minclustersize=0 -bry
```

AM2.9. Alignment step

The last options you may modify concern the use of the alignment program and ReformAlign to fine-tune the alignment. For more information, see the RM examples.

AM2.10. A complex example

In the following example, QRS retrieves all sequences that have more than 350 bp, a mean sequence quality score of 30, no ambiguous characters, and a sequence complexity greater than 79 according to the entropy-based filter. This data set has reverse paired barcodes and the script discards all clusters that have only one sequence. Finally, QRS uses KAlign to align all accepted sequences and this alignment is not fine tuned:

```
QRS.pl -AM2 -folder=folder -informat=fasta -fasta=file.fna -quality=file.qual -hmmfile=ref.hmm -oligos=oligos.csv -design=design.csv -minlen=350 -minqual=30 -allowns=0 -filmet=entropy -filthr=79 -bry -minclusterzise=2 -aligner=kalign -reformal=no
```

AM3. Classifying all 454 sequences into TCS-types (-AM3)

AM3.1. Basic parameters

When this option is used, the batch command can be executed with only basic parameters, like in the following example:

```
QRS.pl -AM3 -folder=test -fasta=myaligneddata.fasta -sample=sample.csv -outfile=allsamples
```

In the previous example, QRS will use myaligneddata alignment to run TCS for the data and

then retrieve a TCS-types FASTA file called `allsamples.fasta` and a frequencies matrix file called `allsamples.abs.csv`. The program makes use of some information from the `samples.csv` file (see "Classifying all 454 sequences into TCS-types (-AM3)" in Usage to read more about `samples.csv`).

AM3.2. Statistical parsimony parameters

You can modify the maximum number of steps you want to use to determine TCS-types with the `-nrsteps` parameter. By default, QRS considers that two sequences belong to the same TCS-type if it has three differences or less (`-nrsteps=3`), like in the following example:

```
QRS.pl -AM3 -folder=test -fasta=myaligneddata.fasta
-sample=sample.csv -nrsteps=3 -outfile=allsamples
```

In this example, the maximum number of steps to determine if two sequences belong from the same TCS-type is five differences:

```
QRS.pl -AM3 -folder=test -fasta=myaligneddata.fasta
-sample=sample.csv -nrsteps=5 -outfile=allsamples
```

Output files

All output files are always saved in different files but their number and kind of files depend of the pipeline step.

In -RM option ("creating a reference file" step), QRS creates a Hidden Markov Model file (*.hmm) that will be used later in the same pipeline. When QRS is used to describe the 454 data set (-AM1 option), this script generates a HTML file and a folder called PRINSEQ_GRAPHS with all PNG files to study the quality control. If the program is used to process the 454 data set (-AM2 option), QRS generates as output a fasta file with contains all accepted and aligned sequences from your 454 data set. Finally, after classifying all 454 sequences into TCS-types (-AM3 option), the program generates a matrix plain file in format *.csv with the frequencies of all TCS-types and the TCS-types sequences in fasta format.

References

Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ (1990) Basic local alignment search tool. *Journal of Molecular Biology*, **215**, 403–410.

Bradley RK, Roberts A, Smoot M *et al.* (2009) Fast statistical alignment. *PLoS Computational Biology*, **5**, e1000392.

Clement M, Posada D, Crandall KA (2000) TCS: a computer program to estimate gene genealogies. *Molecular Ecology*, **9**, 1657–1659.

Edgar RC (2004) MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Research*, **32**, 1792–1797.

Edgar RC (2010) Search and clustering orders of magnitude faster than BLAST. *Bioinformatics*, **26**, 2460–2461.

Edgar RC, Haas BJ, Clemente JC, Quince C, Knight R (2011) UCHIME improves sensitivity and speed of chimera detection. *Bioinformatics*, **27**, 2194–2200.

Finn RD, Clements J, Eddy SR (2011) HMMER web server: interactive sequence similarity searching. *Nucleic Acids Research*, **39**, W29–W37.

Huse S, Huber J, Morrison H, Sogin M, Welch D (2007) Accuracy and quality of massively parallel DNA pyrosequencing. *Genome Biology*, **8**, R143.

Katoh K, Kuma K, Toh H, Miyata T (2005) MAFFT version 5: improvement in accuracy of multiple sequence alignment. *Nucleic Acids Research*, **33**, 511–518.

Katoh K, Misawa K, Kuma K, Miyata T (2002) MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform. *Nucleic Acids Research*, **30**, 3059–3066.

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

Lassmann T, Sonnhammer E (2005) Kalign – an accurate and fast multiple sequence alignment algorithm. *BMC Bioinformatics*, **6**, 1–9.

Li W, Fu L, Niu B, Wu S, Wooley J (2012) Ultrafast clustering algorithms for metagenomic sequence analysis. *Briefings in Bioinformatics*.

Löytynoja A, Goldman N (2005) An algorithm for progressive multiple alignment of sequences with insertions. *Proceedings of the National Academy of Sciences of the United States of America*, **102**, 10557–10562.

Lyras DP, Metzler D (2014) ReformAlign: Improved multiple sequence alignments using a profile-based meta-alignment approach. *Submitted*.

Martin M (2011) Cutadapt removes adapter sequences from high-throughput sequencing reads. *EMBnet.journal,* **17.**

Morgulis A, Gertz EM, Schaffer AA, Agarwala R (2006) A fast and symmetric DUST implementation to mask low-complexity DNA sequences. *Journal of Computational Biology*, **13**, 1028–1040.

Russell D, Otu H, Sayood K (2008) Grammar-based distance in progressive multiple sequence

alignment. *BMC Bioinformatics*, **9**, 306.

Schmieder R, Edwards R (2011) Quality control and preprocessing of metagenomic datasets. *Bioinformatics*, **27**, 864-864.

Schwartz AS, Pachter L (2007) Multiple alignment by sequence annealing. *Bioinformatics*, **23**, e24–e29.

Sievers F, Wilm A, Dineen D *et al.* (2011) Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega. *Molecular Systems Biology*, **7**.

Stajich JE, Block D, Boulez K *et al.* (2002) The Bioperl toolkit: Perl modules for the life sciences. *Genome Research*, **12**, 1611–1618.

Templeton AR, Crandall KA, Sing CF (1992) A cladistic analysis of phenotypic associations with haplotypes inferred from restriction endonuclease mapping and DNA sequence data. III. Cladogram estimation. *Genetics*, **132**, 619–633.

Table S1. Brief summary of all parameters

Parameter	Used in the mentioned steps				Label type	Short description	Observations
	Creation of reference data set	Description of 454 data set	Processing data set and alignment	Classification of sequences into TCS-types			
-folder=	X	X	X	X	General	The folder containing all required files	Mandatory parameter.
-AM1		X			Pipeline step	Describes the 454 data set	It is incompatible with -RM, -AM2, -AM3.
-AM2			X		Pipeline step	Obtains a alignment from all accepted sequences of a 454 data set	It is incompatible with -RM, -AM1, -AM3.
-AM3				X	Pipeline step	Classifies all 454 sequences into TCS-types	It is incompatible with -RM, -AM1, -AM2.
-RM	X				Pipeline step	Creates a reference HMM file from a FASTA file	It is incompatible with -AM1, -AM2, -AM3.
-design=			X		Input files	Input the *design* text file. For a brief description of these file, see main text.	Mandatory parameter if -nosplit is disabled.
-fasta=		X	X	X	Input files	Input 454	Mandatory parameter if

						FASTA file or input an alignment FASTA file.	-informat=fasta (if you are analysing your 454 data set) and mandatory if you classify your sequences into TCS-types (-AM3 option).
-fastq=		X	X		Input files	Input 454 FASTQ file.	Mandatory parameter if -informat=fastq.
-fastq2=		X	X		Input files	Input paired FASTQ file	Mandatory parameter if you have a paired FASTQ data-set (-paired=yes).
-hmmfile=			X		Input files	Input a reference HMM file	Mandatory parameter.
-informat=		X	X		Input files	Kind of 454 input sequences (FASTA or FASTQ files)	Mandatory parameter.
-oligos=			X		Input files	Input the *oligos* text file. For a brief description of these file, see main text.	Mandatory parameter if -nosplit is disabled.
-paired=		X	X		Input files	Specify if your FASTQ files are paired or not.	By default, the option is disabled (-paired=no)
-phred64=		X	X		Input files	Specify if your quality data in FASTQ files are in Phred+64	By default, the option is disabled (-phred64=no)

						format	
-quality=		X	X		Input files	Input quality file	
-reffiles=	X				Input files	Input reference FASTA files, i.e., FASTA files with sequences of specific marker/s.	Mandatory parameter. If you want to analyse more than one marker, you have to specify these files concatenated with a single dash.
-sample=				X	Input files	Input the *sample* text file. For a brief description of these file, see main text.	Mandatory parameter.
-allowns=			X		Filtering	Define the allowed maximum ambiguous bases (Ns).	We recommend not allowing Ns (-allowns=0).
-filmet=			X		Filtering	Filtering method to remove sequences with low complexity.	Accepted options are dust or entropy.
-filthr=			X		Filtering	Specify the threshold for the filtering method (see –filmet option).	Mandatory parameter if you define a filtering method to remove low complexity sequences (-filmet=[dust|entropy])
-hmmthr=			X		Filtering	Specify the	By default, the parameter is

						maximum allowed *e*-value to classify a sequence as a specific marker using a HMM profile.	set to 10^{-10} (-hmmthr=1E-10).
-maxlen=			X		Filtering	Specify the maximum sequence length allowed to pass the filters.	
-minlen=			X		Filtering	Specify the minimum sequence length allowed to pass the filters.	
-minqual=			X		Filtering	Specify the minimum mean sequence quality allowed to pass the filters.	
-nosplit			X		Filtering	Specify this parameter if your data set has trimmed all your primers (e.g.	By default, it is disabled.

						public 454 data sets).	
-trimqual=			X		Filtering	Specify the minimum base quality to trim sequences flanks.	
-trimtails=			X		Filtering	Specify the number of allowed A/Ts to trim poly-A/Ts in sequences flanks.	
-allowmis=			X		Assigning samples	Specify the maximum mismatches number allowed to detect the barcodes. It can be an exact number or a percent.	By default, this value is set to 1% mismatches (-allowmis=1%).
-brn/-bry			X		Assigning samples	Absence or presence of barcoded reverse primers, respectively.	Mandatory parameter if -nosplit is disabled.
-cutoff=			X		De-noising	Specify clustering	By default, QRS calculates this value according to the

						identity threshold to de-noise the data set. It should be a number between 0.00 and 1.00	CDHIT-OUT algorithm.
-minclustersize=			X		De-noising	Specify the minimum size of cluster is accepted to consider that the analysed data set has no artefacts.	By default, all clusters that have at least three sequences are considered as good clusters (-minclustersize=3).
-aligner=	X		X		Alignment	Program used to align your sequences.	Accepted programs are: clustalo, fsa, gramalign, kalign, mafft-accurate, mafft-fast, muscle or prank. By default, PRANK is used (-aligner=prank)
-reformal=	X		X		Alignment	Use ReformAlign to fine-tune the alignment.	By default, this option is enabled (-reformal=yes).
-nrsteps=				X	Defining TCS-types	Specify the maximum number of steps you want to use	By default, the value is set to 3 (-nrsteps=3).

						to determine TCS-types.	
-outfile=				X	Output files	Name of the output files.	Mandatory parameter. Do not add the extension for the file (e.g., txt).