README CONDETRI

Written by Linnéa Smeds May 2011

Contents:

CONDETRI

Summary

Trim FASTQ reads from the 3'-end and extract reads (or read pairs) of good quality. If the reads are paired, the filtering is done pairwise, and if one read in a pair has low quality, the remaining read is saved as single end.

Usage

```
perl condetri.pl -fastq1=file1 [-fastq2=file2 -prefix=s -hq=N
-lq=N -frac=N -minlen=N -mh=N -ml=N -sc=N]
```

Description

The trimming is performed in two steps:

(1) Trimming low quality bases from the 3'-end

(2) Overall quality check of read/pair

Details:

(1) Bases are removed from the 3'-end if the quality score is lower than some threshold (`hq`). When a base with higher quality is reached, it is kept temporarily and the preceding bases are considered. After this point, also bases with low quality can be saved, if they are surrounded by high quality bases. Up to `ml` consecutive low quality bases are saved temporarily, but if the following base also has low quality, all temporarily saved bases are removed, and the trimming starts all over again. The trimming stops either when finding `mh` consecutive high quality bases, or when the read is trimmed down to a certain length (`minlen`).

(2) After the trimming step, the quality scores of the remaining read are controlled for. A read is approved if a certain fraction (`frac`) of the bases have a quality score higher than `hq`, and there is no base in the read that has a quality score below some lower bound (`lq`). If the input is paired-end reads, both reads in a pair must be approved for keeping the pair. If only one of the reads is approved, it is saved in an additional, unpaired file which can be used as single-end data.

Input parameters

(default values in brackets `[]`)

`-fastq1=file`
`-fastq2=file` FASTQ file. If a second file is given, the files are trimmed as a pair. The reads must have the same order in both files.

`-prefix=string` Prefix for the output file(s). The filtered FASTQ file(s) will be named *prefix_trim1.fastq* (and *prefix_trim2.fastq* if present). For pairs, a third file will be given with unpaired reads (reads from pairs where one low quality read has been removed).

`-hq=N` Hiqh quality threshold [25].

`-lq=N` Low quality threshold [10].

`-frac=[0,1]` Fraction of read that must exceed hq [0.8].

`-minlen=N` Min allowed read length [50].

`-mh=N` When this no of consecutive hq bases is reached, the trimming stops [5].

`-ml=N` Max no of lq bases allowed after a stretch of hq bases from 3'-end [1].

-sc=N Illumina scoring table, Score=ASCII-sc, usually 64 for Illumina/Solexa. 33 is Sanger standard. Can be set to any other integer if wanted [64].

-q Print the Illumina scoring table.

-h Print a help message.

Output files

prefix_trim1.fastq
prefix_trim2.fastq
prefix_trim_unpaired.fastq File(s) with the trimmed reads (one file for single-end data, three for paired-end data, where the last file includes reads from the two input files whose read pair had too poor quality.

prefix.stats Includes basic statistics in columns.

The columns for the .stats file are the following:
PREFIX, NUMBER OF READS IN ORIGINAL FILE(S), NUMBER OF BASES IN ORIGINAL FILE(S), NO OF PAIRED READS AFTER TRIMMING,
NO OF BASES IN PAIRS AFTER TRIMMING, NO OF UNPAIRED READS AFTER TRIMMING, NO OF UNPAIRED BASES AFTER TRIMMING.

The .stats files are suitable for concatenation to make a summary table for several FASTQ files, for example one file for each lane in a flowcell or a set of transcriptome samples. Just use:

```
$ cat *.stats
```

filterPCRdupl.pl

Summary

Takes a pair of FASTQ files and removes redundant copies that might have emerged in the PCR step by comparing all read pairs against each other. If there are several copies of a read pair, only one pair is kept (the one with the highest base quality).

Usage

```
perl filterPCRdupl.pl -fastq1=file1 -fastq2=file2 [-prefix=s
-cmp=N]
```

Background

When sequencing with NGS techniques, PCR is often needed for adding adaptors and/or getting a sufficient amount of fragments for sequencing. However, it also infers the risk of amplifying the same fragments over and over again, and in worst case scenario only sequence a small part of the desired material. By inspecting the data and remove these so called PCR-duplicates we can prevent biases and erroneous results in the down stream analysis.

Especially in RNA-sequencing for expression analysis it is crucial no to include these ”false” fragments as they will change the gene expression for the genes with duplicated fragments and skew the analysis. Also for *de novo* assembly the duplicates should be removed not to add extra links in the scaffolding procedure. For mapping and SNP-calling, the removal of duplicates is less of importance, since most SNP-callers can recognize and flag these as unusable. However, since the duplicates can't be used anyway, one can save time and memory by removing them before performing any further analyses.

Description

All pairs are compared against each other by examination of the first N (default 50) bases of both reads in the pairs. If there are several pairs starting with exactly the same sequence in both ends, it is assumed that it's a duplication since it is unlikely that the two DNA-molecules have been fragmented at exactly the same positions at both ends just by random (and if N is large enough, it's also unlikely that one would find exactly the same pattern somewhere else in the genome). When finding several copies of a fragment, the quality scores for the pairs are compared, and the pair with the highest total score is kept while the others are discarded.

Input parameters

(default values in brackets [ ])

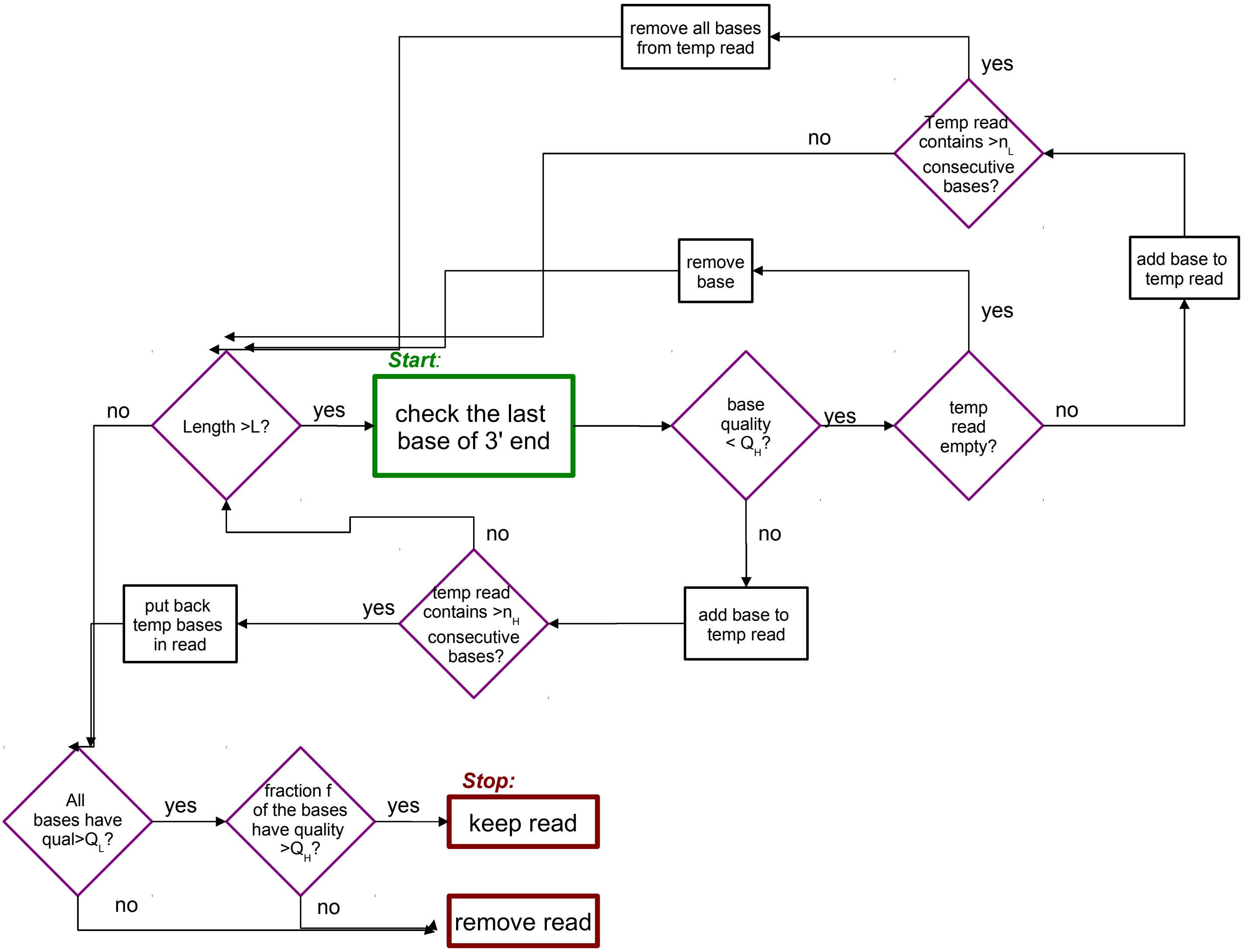
-fastq1=file1	First file in a FASTQ pair.
-fastq2=file2	Second file in a FASTQ pair. The reads must have the same order in both files.
-prefix=string	Prefix for the output files. The filtered FASTQ file(s) will be named *prefix_uniq1.fastq* and *prefix_uniq2.fastq* if present [same as file1].
-cmp=N	Number of bases in the beginning of both reads in a pair that are used for comparison [50].

Output files

prefix_uniq1.fastq Files with the filtered (non redundant) reads.
prefix_uniq1.fastq

prefix_copy.hist Histogram of the copy distribution.

CONDETRI FLOWCHART:



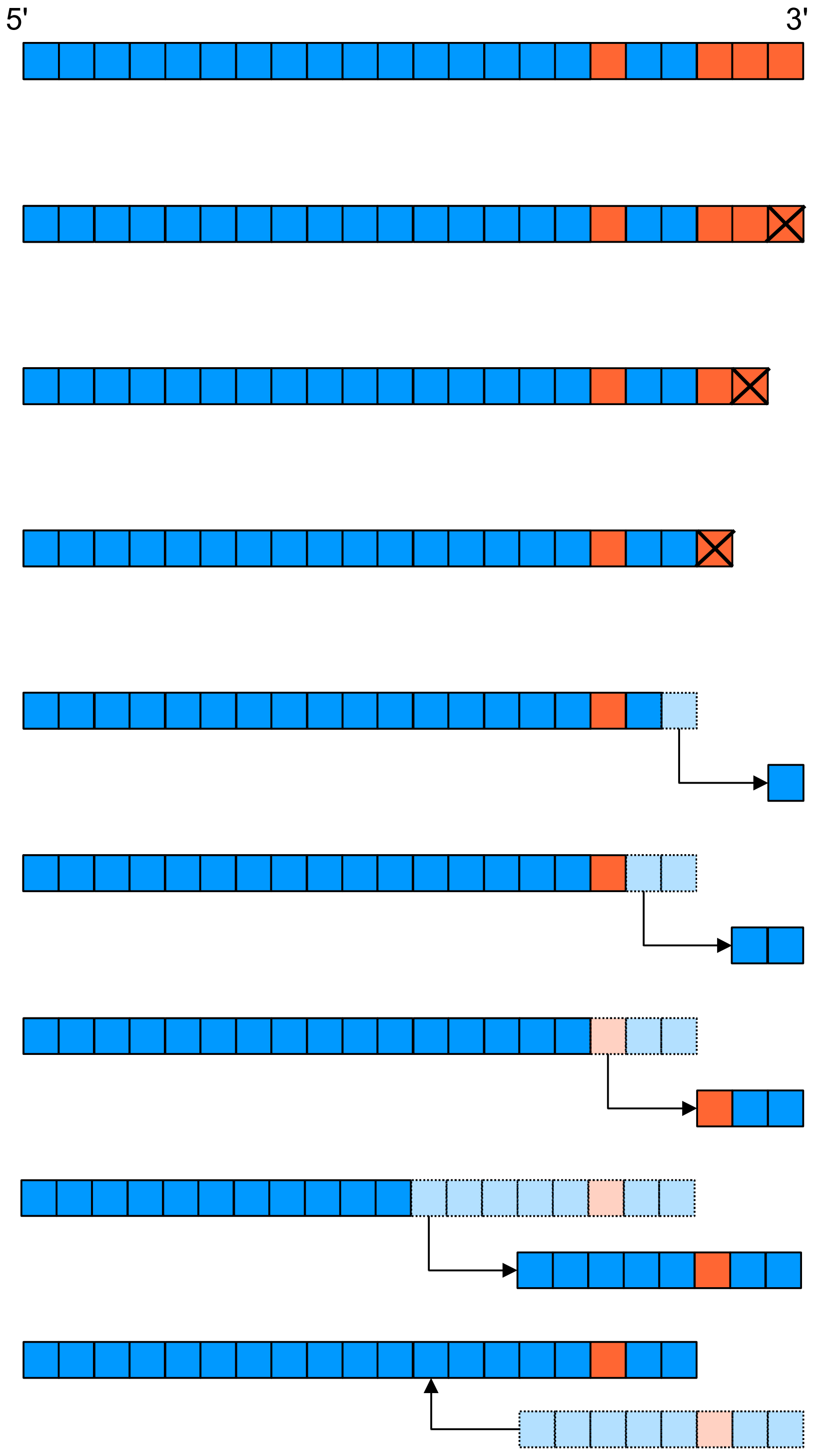
remove all bases from temp read
yes
no
Temp read contains >n_L consecutive bases?
remove base
add base to temp read
yes
Start:
no
Length >L?
yes
check the last base of 3' end
base quality < Q_H?
yes
temp read empty?
no
no
no
put back temp bases in read
yes
temp read contains >n_H consecutive bases?
add base to temp read
All bases have qual>Q_L?
yes
fraction f of the bases have quality >Q_H?
yes
Stop:
keep read
no
no
remove read

CONDETRI TRIMMING STEP:

Example 1: Good quality read

🟦 Base with quality >Q_H

🟧 Base with quality <Q_H



Trim bases with quality less than Q_H

Save high quality bases temporarily

Up to n_L consecutive low quality bases can also be saved temporarily

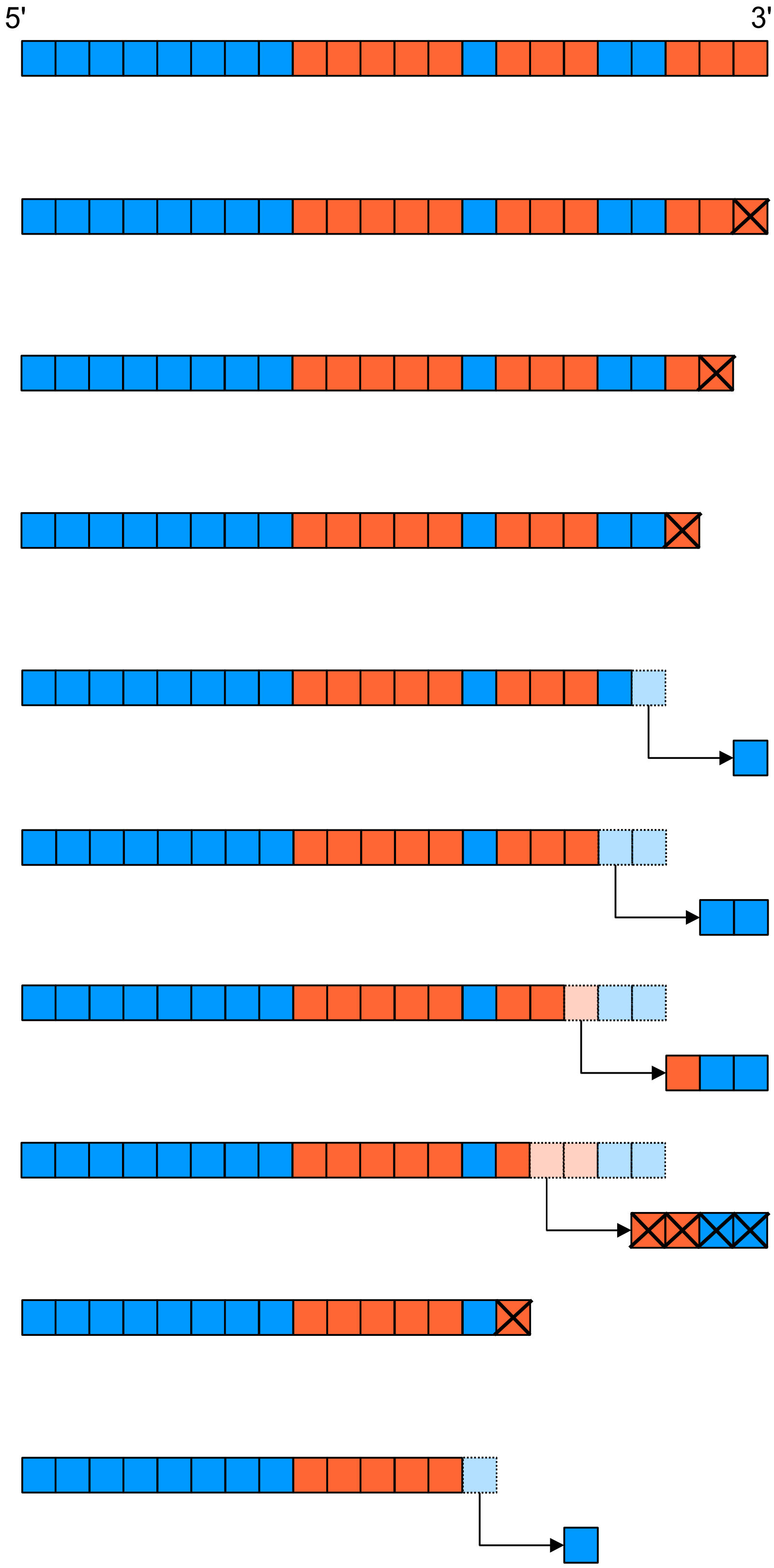
When n_H consecutive bases are found the trimming is terminated

Temporarily saved bases are added back to the read.

Example 2: Poor quality read

🟦 Base with quality >Q_H

🟧 Base with quality <Q_H



Trim bases with quality less than Q_H

Save high quality bases temporarily

More than n_L consecutive low quality bases – remove temporarily saved bases

Start the process over again

Continue either until finding n_H consecutive bases, or the length of the read reaches L