Trimmomatic Manual: V0.32

Introduction

Trimmomatic is a fast, multithreaded command line tool that can be used to trim and crop Illumina (FASTQ) data as well as to remove adapters. These adapters can pose a real problem depending on the library preparation and downstream application.

There are two major modes of the program: Paired end mode and Single end mode. The paired end mode will maintain correspondence of read pairs and also use the additional information contained in paired reads to better find adapter or PCR primer fragments introduced by the library preparation process.

Trimmomatic works with FASTQ files (using phred + 33 or phred + 64 quality scores, depending on the Illumina pipeline used). Files compressed using either 'gzip' or 'bzip2' are supported, and are identified by use of '.gz' or '.bz2' file extensions.

Implemented trimming steps (Quick reference)

Trimmomatic performs a variety of useful trimming tasks for illumina paired-end and single ended data. The selection of trimming steps and their associated parameters are supplied on the command line.

The current trimming steps are:

- ILLUMINACLIP: Cut adapter and other illumina-specific sequences from the read.
- SLIDINGWINDOW: Performs a sliding window trimming approach. It starts scanning at the 5' end and clips the read once the average quality within the window falls below a threshold.
- MAXINFO: An adaptive quality trimmer which balances read length and error rate to maximise the value of each read
- LEADING: Cut bases off the start of a read, if below a threshold quality
- TRAILING: Cut bases off the end of a read, if below a threshold quality
- CROP: Cut the read to a specified length by removing bases from the end
- HEADCROP: Cut the specified number of bases from the start of the read
- MINLEN: Drop the read if it is below a specified length
- AVGQUAL: Drop the read if the average quality is below the specified level
- TOPHRED33: Convert quality scores to Phred-33
- TOPHRED64: Convert quality scores to Phred-64

Running Trimmomatic

Processing Order

The different processing steps occur in the order in which the steps are specified on the command line. It is recommended in most cases that adapter clipping, if required, is done as early as possible, since correctly identifying adapters using partial matches is more difficult.

Single End Mode

For single-ended data, one input and one output file are specified. The required processing steps (trimming, cropping, adapter clipping etc.) are specified as additional arguments after the input/output files.

java -jar <path to trimmomatic jar> SE [-threads <threads>] [-phred33 | -phred64] [-trimlog <logFile>] <input> <output> <step 1> ...

or

java -classpath <path to trimmomatic jar> org.usadellab.trimmomatic.TrimmomaticSE [-threads <threads>] [-phred33 | -phred64] [-trimlog <logFile>] <input> <output> <step 1> ...

-phred33 or -phred64 specifies the base quality encoding. If no quality encoding is specified, it will be determined automatically (since version 0.32). The prior default was -phred64.

Specifying a trimlog file creates a log of all read trimmings, indicating the following details:

- the read name
- the surviving sequence length
- the location of the first surviving base, aka. the amount trimmed from the start
- the location of the last surviving base in the original read
- the amount trimmed from the end

Multiple steps can be specified as required, by using additional arguments at the end as described in the section processing steps.

For input and output files adding .gz/.bz2 to an extension tells Trimmomatic that the file is provided in gzip/bzip2 format or that Trimmomatic should gzip/bzip2 the file, respectively.

Paired End Mode

For paired-end data, two input files, and 4 output files are specified, 2 for the 'paired' output where both reads survived the processing, and 2 for corresponding 'unpaired' output where a read survived, but the partner read did not.

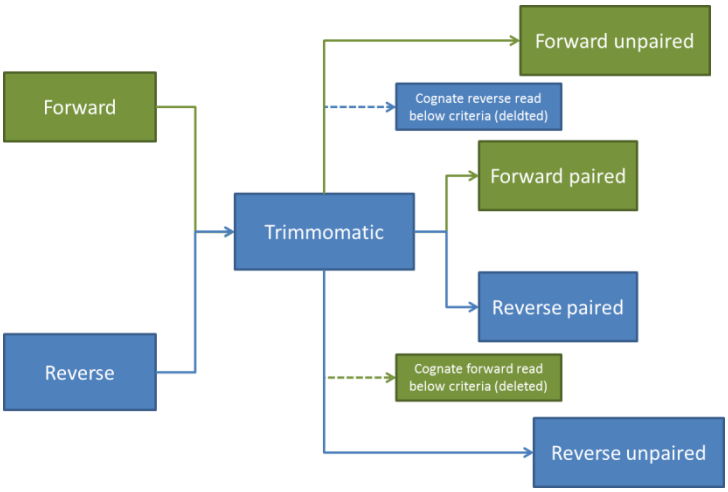


Figure 1: Flow of reads in Trimmomatic Paired End mode

java -jar <path to trimmomatic.jar> PE [-threads <threads] [-phred33 | -phred64] [-trimlog <logFile>] >] [-basein <inputBase> | <input 1> <input 2>] [-baseout <outputBase> | <unpaired output 1> <paired output 2> <unpaired output 2> <step 1> ...

or

java -classpath <path to trimmomatic jar> org.usadellab.trimmomatic.TrimmomaticPE [-threads <threads>] [-phred33 | -phred64] [-trimlog <logFile>] [-basein <inputBase> | <input 1> <input 2>] [-baseout <outputBase> | <paired output 1> <unpaired output 1> <paired output 2> <unpaired output 2> <step 1> ...

-phred33 or -phred64 specifies the base quality encoding. If no quality encoding is specified, it will be determined automatically (since version 0.32). The prior default was -phred64.

-threads indicates the number of threads to use, which improves performance on multi-core computers. If not specified, it will be chosen automatically.

Specifying a trimlog file creates a log of all read trimmings, indicating the following details:

- the read name
- the surviving sequence length
- the location of the first surviving base, aka. the amount trimmed from the start
- the location of the last surviving base in the original read
- the amount trimmed from the end

Multiple steps can be specified as required, by using additional arguments at the end as described in the section processing steps.

Input/Output Files

Paired-end mode requires 2 input files (for forward and reverse reads) and 4 output files (for forward paired, forward unpaired, reverse paired and reverse unpaired reads).

Since these files often have similar names, the user has the option to provide either the individual file names, or just one name from which the file names can be derived.

For input files, either of the following can be used:

- Explicitly naming the 2 input files
- Naming the forward file using the -basein flag, where the reverse file can be determined automatically. The second file is determined by looking for common patterns of file naming, and changing the appropriate character to reference the reverse file. Examples which should be correctly handled include:
    - Sample_Name_R1_001.fq.gh -> Sample_Name_R2_001.fq.gz
    - Sample_Name.f.fastq -> Sample_Name.r.fastq
    - Sample_Name.1.sequence.txt -> Sample_Name.2.sequence.txt

For output files, either of the following can be used:

- Explicity naming the 4 output files
- Providing a base file name using the –baseout flag, from which the 4 output files can be derived. If the name "mySampleFiltered.fq.gz" is provided, the following 4 file names will be used:
    - mySampleFiltered_1P.fq.gz - for paired forward reads
    - mySampleFiltered_1U.fq.gz - for unpaired forward reads
    - mySampleFiltered_2P.fq.gz - for paired reverse reads
    - mySampleFiltered_2U.fq.gz - for unpaired reverse reads

For input and output files adding .gz to an extension tells Trimmomatic that the file is provided in gzipped format or that Trimmomatic should gzip the file, respectively. This extension can be used with both explicitly named and template-based file naming.

Processing Steps in Detail

Most processing steps take one or more settings, delimited by ':' (a colon)

ILUMINACLIP

This step is used to find and remove Illumina adapters.

Identifying adapter or other contaminant sequences within a dataset is inherently a trade off between sensitivity (ensuring all contaminant sequences are removed) and specificity (leaving all non-contaminant sequence data intact). This problem is even more acute when only a small part of the contaminant sequence is included within the read. The possibility of sequencing errors within the reads complicates the process still further.

Although adapter and other technical sequences can potentially occur in any location within reads, by far the most common cause of adapter contamination is sequencing of a DNA fragment which is shorter than the read length. In this scenario, the beginning of the read contains valid data, but when the end of the fragment is reached, the sequencer continues to 'read-through' into the adapter. This results in a partial or full adapter sequence towards the 3' end of the read. While a full adapter sequence can be identified relatively easily, reliably identifying a short partial adapter sequence is inherently difficult.

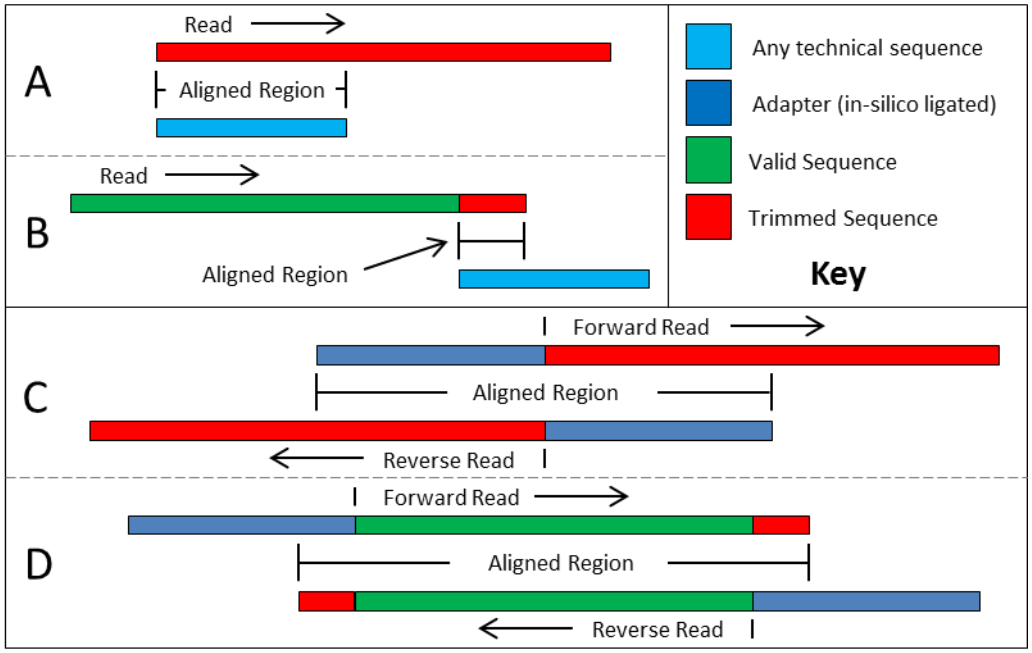
Interestingly, in a paired-end dataset, 'read-through' will occur on both the forward and reverse reads of a particular fragment in the same position, and also, since the fragment was entirely sequenced from both ends, the non-adapter portion of the forward and reverse reads will be reverse-complements. Since adapter read-through is a relatively common occurrence, and since Illumina datasets are often paired-end, Trimmomatic includes a second adapter identification strategy, specifically for adapter read-though and which takes advantage of the added evidence available in paired-end data. This strategy is known as 'palindrome' mode.

The diagram below illustrates both strategies.

In A, the read contains the entire technical sequence within the read, and thus a standard alignment approach is sufficient to determine this fact. In B, only part of the technical sequence is contained at the 3' end of the read, and thus only a short alignment can be used. Below some length threshold, which depends on the relative costs of false positives and false negatives, it is no longer possible to identify an adapter sequence, thus many short adapter fragments will remain.

In D, the 'palindrome' approach is used to check a similar situation with a short contaminant at the 3' end of the read. However, due to exploiting the additional data available in paired-end mode, the region tested as part of the alignment is much longer. Not only are both adapter sequences tested at once, but the fragment sequence from each reads are also checked. This alignment is thus much more reliable than the short alignment in B, and allows adapter 'read-though' to be detected even when only one base of the adapter has been sequenced.

'C' shows how palindrome mode can also detect long 'read-though'. In this example, there is no useful fragment at all, and both reads begin with sequence from the adapters. None the less, there is still a sizeable alignment, and thus this scenario can be reliably identified.



Trimmomatic uses a two-step approach to find matches between the adapters and reads. First, short sections of each adapter (maximum 16 bp) are tested in each possible position within the reads. If this short alignment, known as the 'seed' is a perfect or sufficiently close match, determined by the seedMismatch parameter (see below), the entire alignment between the read and adapter is scored. This two-step strategy results in considerable efficiency gains, since the seed alignment can be calculated very quickly, while the full alignment score is calculated relatively rarely.

The full alignment score is calculated as follows. Each matching base increases the alignment score by 0.6, while each mismatch reduces the alignment score by Q/10. By considering the quality of the base calls, mismatches caused by read errors have less impact. A perfect match of a 12 base sequence will score just over 7, while 25 bases are needed to score 15. As such we recommend values of between 7 - 15 as the threshold value for simple alignment mode. .

For palindromic matches, a longer alignment is possible, as described above. Therefore this threshold can be higher, in the range of 30. Even though this threshold is very high (requiring a match of almost 50 bases) Trimmomatic is still able to identify very, very short adapter fragments. (See Figure 2 panels C and D, where the alignment regions are shown).

ILLUMINACLIP:<fastaWithAdaptersEtc>:<seed mismatches>:<palindrome clip threshold>:<simple clip threshold>

fastaWithAdaptersEtc: specifies the path to a fasta file containing all the adapters, PCR sequences etc. The naming of the various sequences within this file determines how they are used. See the section below or use one of the provided adapter files

seedMismatches: specifies the maximum mismatch count which will still allow a full match to be performed

palindromeClipThreshold: specifies how accurate the match between the two 'adapter ligated' reads must be for PE palindrome read alignment.

simpleClipThreshold: specifies how accurate the match between any adapter etc. sequence must be against a read.

ILLUMINACLIP also supports two additional optional parameters, which affect palindrome mode only.

ILLUMINACLIP:<fastaWithAdaptersEtc>:<seed mismatches>:<palindrome clip threshold>:<simple clip threshold>:<minAdapterLength>

ILLUMINACLIP:<fastaWithAdaptersEtc>:<seed mismatches>:<palindrome clip threshold>:<simple clip threshold>:<minAdapterLength>:<keepBothReads>

minAdapterLength: In addition to the alignment score, palindrome mode can verify that a minimum length of adapter has been detected. If unspecified, this defaults to 8 bases, for historical reasons. However, since palindrome mode has a very low false positive rate, this can be safely reduced, even down to 1, to allow shorter adapter fragments to be removed.

keepBothReads: After read-though has been detected by palindrome mode, and the adapter sequence removed, the reverse read contains the same sequence information as the forward read, albeit in reverse complement. For this reason, the default behaviour is to entirely drop the reverse read. By specifying 'true' for this parameter, the reverse read will also be retained, which may be useful e.g. if the downstream tools cannot handle a combination of paired and unpaired reads.

SLIDINGWINDOW

Perform a sliding window trimming, cutting once the average quality within the window falls below a threshold. By considering multiple bases, a single poor quality base will not cause the removal of high quality data later in the read.

SLIDINGWINDOW:<windowSize>:<requiredQuality>

windowSize: specifies the number of bases to average across

requiredQuality: specifies the average quality required.

MAXINFO

Performs an adaptive quality trim, balancing the benefits of retaining longer reads against the costs of retaining bases with errors.

For many applications, the "value" of a read is a balance between three factors:

- Minimal read length: The read needs to be long enough that it can be uniquely located within the target sequence. Extremely short reads, which can be placed into many different locations within the target sequence, provide little value. The length required

before a read is likely to be unique depends on the size and complexity of the target sequence, but a typical target length would be in the order of 40 bases.
- Additional read length: There may be added value in retaining additional bases, beyond those needed to uniquely place a read. This is dependent primarily on the application. For pure counting applications, such as RNA-Seq, unique placement is sufficient. For assembly or variant finding tasks, additional bases provide extra evidence for or against putative results, and thus can be valuable.
- Error sensitivity: The downstream analysis can be more or less sensitive to errors within the data. This is determined by the tools and settings used. One extreme would be tools were a single base error would cause the entire read to be ignored, which favours aggressive quality trimming. The other extreme would be tools which can tolerate or even correct a large number of errors, which favour retaining as much data as possible.

Two user provided values are used, the "target read length", which affects the first scoring factor, and the "strictness" which affects the balance between the second and third factor.

Trimming is applied to the 3' end of the read, by considering these 3 factors at every possible position, as scoring as follows:

- Minimal read length: The difference between the putative read length and the target read length is scored using a logistic function. This means that reads shorter the target length are heavily penalized, but most of the scoring benefit can be achieved by reads only marginally longer than this target.
- Additional read length: The putative read length is scored linearly, and weighted by the "leniency" of the trimming, where leniency = (1 – strictness).
- Error sensitivity: The quality scores of the putatively retained bases are combined to calculate the probability that the read is error-free. This score is then weighted by the "strictness".

The combined score is calculated for each possible read length, and the optimal score is used to determine where the read should be trimmed. In practice, the different factors combine as follows:

- At very short read lengths, the minimal read length factor dominates. This will heavily penalize reads which are too short to be useful.
- Once the target read length has been achieved, the minimal read length factor penalty becomes a modest bonus. However, once the read is significantly longer than the target length, further bonuses from the minimal read length factor are limited, due to the logistic function.
- The additional read length factor then provides a modest benefit as additional bases are retained. This is countered by the increasing penalty as the 'error-free' probability drops with increasing read length. The balance between these two factors is controlled by the "strictness" parameter.
- For most reads, depending on the quality of the read and the "strictness" setting, the increasing penalty from the likelihood of error exceeds the bonus of retaining additional bases at some point, and the read is trimmed accordingly.

MAXINFO:<targetLength>:<strictness>

targetLength: This specifies the read length which is likely to allow the location of the read within the target sequence to be determined.

strictness: This value, which should be set between 0 and 1, specifies the balance between preserving as much read length as possible vs. removal of incorrect bases. A low value of this parameter (<0.2) favours longer reads, while a high value (>0.8) favours read correctness.

LEADING

Remove low quality bases from the beginning. As long as a base has a value below this threshold the base is removed and the next base will be investigated.

LEADING:<quality>

quality: Specifies the minimum quality required to keep a base.

TRAILING

Remove low quality bases from the end. As long as a base has a value below this threshold the base is removed and the next base (which as trimmomatic is starting from the 3' prime end would be base preceding the just removed base) will be investigated. This approach can be used removing the special illumina 'low quality segment' regions (which are marked with quality score of 2), but we recommend Sliding Window or MaxInfo instead

TRAILING:<quality>

quality: Specifies the minimum quality required to keep a base.

CROP

Removes bases regardless of quality from the end of the read, so that the read has maximally the specified length after this step has been performed. Steps performed after CROP might of course further shorten the read.

CROP:<length>

length: The number of bases to keep, from the start of the read.

HEADCROP

Removes the specified number of bases, regardless of quality, from the beginning of the read.

HEADCROP:<length>

length: The number of bases to remove from the start of the read.

MINLEN

This module removes reads that fall below the specified minimal length. If required, it should normally be after all other processing steps. Reads removed by this step will be counted and included in the ‘dropped reads’ count presented in the trimmomatic summary.

MINLEN:<length>

length: Specifies the minimum length of reads to be kept.

TOPHRED33

This (re)encodes the quality part of the FASTQ file to base 33.

TOPHRED33 (no further parameters)

TOPHRED64

This (re)encodes the quality part of the FASTQ file to base 64.

TOPHRED64 (no further parameters)

Examples

Paired End

java -jar trimmomatic-0.30.jar PE s_1_1_sequence.txt.gz s_1_2_sequence.txt.gz lane1_forward_paired.fq.gz lane1_forward_unpaired.fq.gz lane1_reverse_paired.fq.gz lane1_reverse_unpaired.fq.gz ILLUMINACLIP:TruSeq3-PE.fa:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36

This will perform the following **in this order**

- Remove Illumina adapters provided in the TruSeq3-PE.fa file (provided). Initially Trimmomatic will look for seed matches (16 bases) allowing maximally **2** mismatches. These seeds will be extended and clipped if in the case of paired end reads a score of **30** is reached (about 50 bases), or in the case of single ended reads a score of **10**, (about 17 bases).
- Remove leading low quality or N bases (below quality **3**)
- Remove trailing low quality or N bases (below quality **3**)
- Scan the read with a **4**-base wide sliding window, cutting when the average quality per base drops below **15**
- Drop reads which are less than **36** bases long after these steps

根据质量值修剪read首尾碱基

Single End

java -jar trimmomatic-0.30.jar SE s_1_1_sequence.txt.gz lane1_forward.fq.gz ILLUMINACLIP:TruSeq3-SE:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36

This will perform the same steps, using the single-ended adapter file. (Of course the **:30:** parameter has no effect, but a value has to be specified nevertheless)

The Adapter Fasta

Using one of the supplied Fasta Files

Illumina adapter and other technical sequences are copyrighted by Illumina, but we have been granted permission to distribute them with Trimmomatic. Suggested adapter sequences are provided for **TruSeq2 (as used in GAII machines) and TruSeq3 (as used by HiSeq and MiSeq machines)**, for both single-end and paired-end mode. These sequences have not been extensively tested, and depending on specific issues which may occur in library preparation, other sequences may work better for a given dataset. **As a rule of thumb newer libraries will use TruSeq3, but this really depends on your service provider.**

If you use FASTQC, the 'Overrepresented Sequences' report can help indicate which adapter file is best suited for your data.

"Illumina Single End" or "Illumina Paired End" sequences indicate single-end or paired-end TruSeq2 libraries, and the appropriate adapter files are "TruSeq2-SE.fa" and "TruSeq2-PE.fa" respectively.

"TruSeq Universal Adapter" or "TruSeq Adapter, Index …" indicates TruSeq-3 libraries, and the appropriate adapter files are "TruSeq3-SE.fa" or "TruSeq3-PE.fa", for single-end and paired-end data respectively.

Adapter sequences for TruSeq2 multiplexed libraries, indicated by "Illumina Multiplexing …", **and the various RNA library preparations are not currently included.**

We are planning to support additional library preparations, such as Nextera, in the future, based on user feedback.

Making custom clipping files

Trimmomatic uses two strategies for adapter trimming: Palindrome and Simple

With 'simple' trimming, each adapter sequence is tested against the reads, and if a sufficiently accurate match is detected, the read is clipped appropriately.

'Palindrome' trimming is specifically designed for the case of 'reading through' a short fragment into the adapter sequence on the other end. In this approach, the appropriate adapter sequences are 'in silico ligated' onto the start of the reads, and the combined adapter+read sequences, forward and reverse are aligned. If they align in a manner which indicates 'read-through', the forward read is clipped and the reverse read dropped (since it contains no new data).

Providing sequences for palindrome trimming

Naming of the sequences determines how they are used.

For 'Palindrome' clipping, a matched pair (or multiple matched pairs) of adapter sequences must be provided. The sequence names should both start with 'Prefix', and end in '/1' for the forward adapter and '/2' for the reverse adapter. The part of the name between 'Prefix' and '/1' or '/2' must match exactly within each pair.

Sequences which begin with ‘Prefix’ but do not form a matched pair are currently used in simple mode, but in future this will be treated as an error.

Providing sequences for simple trimming

All sequences not used in palindrome mode are used in simple mode.

Those with names ending in '/1' or '/2' will be checked only against the forward or reverse read respectively. All other sequences will be checked against both the forward and reverse read.

If you want to check for the reverse-complement of a specific sequence, you need to specifically include the reverse-complemented form of the sequence as well, with another name. As an example have a look at the TruSeq2-PE.fa file

```
>PCR_Primer1
AATGATACGGCGACCACCGAGATCTACACTCTTTCCCTACACGACGCTCTTCCGATCT
>PCR_Primer1_rc
AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGTAGATCTCGGTGGTCGCCGTATCATT
```

As one can see the Primer is included as forward as well as its reverse complement. Trimmomatic by default does not check for the reverse complement, as certain sequences are much more likely occur in one orientation that the other.

Changelog

- 0.30
  - Feature: Add ‘AVGQUAL’ trimmer
  - Bug Fix: Correct ‘half-window’ clipping from end of reads in SLIDINGWINDOW (bug introduced in 0.27)
  - Performance: Various improvements to simple alignment in ILLUMINACLIP
  - Internal: Clean up & refactor ILLUMINACLIP implementation.
- 0.27
  - Feature: Add support for ‘keep reverse’ and ‘minimum adapter length’ to palindrome mode in ILLUMINACLIP
  - Feature: Trimmomatic can be executed with –jar. The ‘old’ method, using the explicit class, continues to work.
  - Feature: Add ‘MAXINFO’ trimmer
  - Feature: Improve support for short adapters in simple mode in ILLUMINACLIP
  - Performance: Improve FastQ parser and serializer
- 0.25
  - Bug Fix: Handle concatenated Gzip files correctly
  - Bug Fix: Ensure input files are read only once, to allow streaming via shell
  - Bug Fix: Report failures correctly in multi-threaded mode
- 0.22
  - Feature: Support ZIP and BZIP2 format
  - Feature: Basic support for short adapters in simple mode in ILLUMINACLIP
- 0.20
  - Feature: Multithreading support
  - Feature: Summary trim statistics
  - Bug Fix: Incorrect alignment score in ILLUMINACLIP