October 18, 2014

SeqyClean User Manual

Ilya Y. Zhbannikov[1], Samuel S. Hunter[1,2], Matthew L. Settles[1,2,3]

[1]Institute for Bioinformatics and Evolutionary Studies, University of Idaho, Moscow, ID 83844-3051,
[2]Department of Biological Sciences, University of Idaho, Moscow, ID 83844-3051,
[2]Department of Bioinformatics and Computational Biology, University of Idaho, Moscow, ID 83844-3051,

Keywords:
Bioinfomatics, Genomics

Correspondence:
Ilya Y. Zhbannikov
Department of Bioinformatics and Computational Biology
University of Idaho
Moscow, ID 83844-30521
Email: zhba3458@vandals.uidaho.edu

Running Title:
SeqyClean User Manual

1 Introduction

We developed SeqyClean – a bioinformatics software pipeline for next-generation sequence cleaning. The first purpose of SeqyClean is to incorporate all aspects of NGS cleaning: adapter, contaminant, poly A/T and quality trimming into a single bioinformatics pipeline. SeqyClean successfully recognizes and removes technological components (adapters, primers, barcodes), contaminants and vector. SeqyClean provides a comprehensive flexible quality trimming by incorporation the LUCY© quality trimming algorithm to remove bad-quality and poly-A/T erroneous data. Below we conducted a survey of existing sequence preprocessing applications and briefly summarized it in Table 1. According to the Table 1 SeqyClean is the only application that is a full preprocessing pipeline for complete next-generation sequence cleaning. In addition, SeqyClean has more features: extension paired-end reads by overlap and duplicates removal, which we consider important for genome assembly because it reduces data space by discarding duplicated reads.

Table 1: Comparison of existing cleaning tools

Application	SFF	FastQ	Contaminants	Vector	Adapter	Quality	Poly A/T	Overlap	Duplicates
SeqyClean	X	X	X	X	X	X	X	X	X
AlienTrimmer		X	X	X	X*				
FASTX-Toolkit		X			X	X			
SeqClean		X	X			X	X		
Lucy				X		X	X		
SeqClean			X			X			
TrimEST							X		
VectorStrip				X					
VectorScreen				X					
Figaro				X					

*It considers adapter as a contaminant

2 Installation

2.0.1 How to download

SeqyClean is an open-source software application available from the Bitbucket for free under this link: https://bitbucket.org/izhbannikov/seqyclean/get/stable.zip. Save the file under some name you wish.

Unzip and compile:

`cd path_to_SeqyClean_directory`

`make`

`make test` (optional)

2.0.2 Usage

SeqyClean works on SFF files (454, Ion Torrent) and FASTQ Illumina (paired- and single-end reads).

SFF files

```
./seqyclean [options] -454 input_filename -o output_prefix
```

Main arguments

`-454 input_filename`	The filename of library to be cleaned. Can be in SFF or FASTQ formats. "-454" tells the program to clean Roche 454 reads
`-o output_prefix`	The files produced will start with the output_prefix followed by some formatted ending (see section "Output files: naming convention")

SFF options, extended

A complete set of options for cleaning 454 and Ion Torrent libraries is shown in Table 2

FastQ paired- and single-end libraries

Paired-end library

```
./seqyclean [options] -1 input_filename_R1 -2 input_filename_R2 -o output_prefix
```

The descriptions of main arguments for paired- and single-end libraries are shown in Table 3.

The descriptions of `[options]` for Illumina libraries are shown in Table 4.

Help

For help please use: `seqyclean -?  or --help`

Quick examples of usage

454 and Ion Torrent:

```
./seqyclean -v test_data/vectors.fasta -qual 30 25 -454 in.sff -o
cleaned_data/Small454Test_cleaned
```

In this example, we clean the library file in.sff that presumably contain vector genome; we apply the quality trimming with max_avg_error of 30 and max_err_at_ends of 25. The output files according to the provided output prefix Small454Test_cleaned are:

- Small454Test_cleaned.sff
- Small454Test_cleaned_SummaryStatistics.txt
- Small454Test_clened_SummaryStaristics.tsv
- Small454Test_cleaned_Report.csv

Example for paired-end reads: `./seqyclean -v test_data/vectors.fasta -c test_data/contaminants.fasta -qual -1 test_data/R1.fastq.gz -2`

`test_data/R2.fastq.gz -o cleaned_data/cleaned` Here, R1/2.fastq.gz are files of paired-end library that contains vector and contaminants; contaminant genomes are stored in contaminants.fasta, quality trimming will is performed with default parameters and the output in this case is:

- cleaned_PE1.fastq
- cleaned_PE2.fastq
- cleaned_SE.fastq
- cleaned_SummaryStatistics.txt
- cleaned_SummaryStatistics.tsv
- cleaned_Report.tsv

Example for single-end reads: `./seqyclean -U test_data/R1.fastq.gz -o cleaned_data/cleaned` Here, R1.fastq.gz is a single-end library file; only adaper trimming will be performed and the output in this case is:

- cleaned_SE.fastq
- cleaned_SummaryStatistics.txt
- cleaned_SummaryStatistics.tsv
- cleaned_Report.tsv

2.1 Output files: naming conventions (Description of seqyclean output)

Depending on the given parameters and the cleaning strategy, the name of output file can be different and has the formats described below.

2.1.1 SFF (454, Ion Torrent)

After processing reads stored in SFF file, SeqyClean outputs a cleaned file by default in Standard Flowgam Format (SFF) and (if option `--fastq` was chosen) in FASTQ format. Also two report files: Prefix_SummaryStatistics.txt (which contains information about how many reads were processed, trimmed, discarded and some additional preprocessing information) and Output_prefix_Report.csv file which holds the detailed statistics for every read.

- Output_prefix.sff , .fastq (optionally)
- Output_prefix_Report.tsv
- Prefix_SummaryStatistics.txt
- Prefix_SummaryStatistics.tsv

2.2 FASTQ

After processing FASTQ reads, SeqyClean generates two (shuffled file and file with single-end reads) or three (PE1 and PE2 files that contain paired-end reads and one file with single-end reads) output files in FASTQ format. Optionally, if the *–overlap* flag was used, SeqyClean outputs single-end reads combined from paired-end reads by overlap into *Output_prefix_SE.fastq* file.

- Output_prefix_PE1.fastq
- Output_prefix_PE2.fastq
- Output_prefix_shuffled.fastq (if `--shuffle` flag was set)
- Output_prefix_SE.fastq
- Output_prefix_PE1_Report.tsv

- Output_prefix_PE2_Report.tsv
- Prefix_SummaryStatistics.txt
- Prefix_SummaryStatistics.tsv

2.2.1 Removing PCR duplicates (Description of the algorithm for removal of PCR duplicates)

Removing duplicates can be considered to mitigate the effects of PCR amplification bias introduced during the library construction. Removing duplicates also reduces the pool of reads before assembly and mapping and thereby reduces alignment time. We define duplicates as reads with identical sequence. The algorithm considers bases within the position from 10 to 25 in the read and does not look at the whole sequence (Figure 1). In present algorithm, detection of duplicates depends on library quality because it needs to compare the same regions of the read and its pair. If these regions is only one base different, they will be considered as distinct.

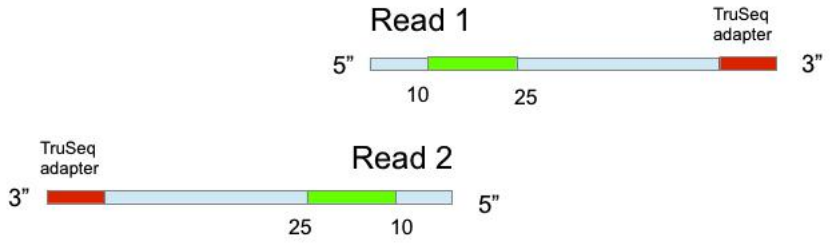

Figure 1: Paired-end Illumina sequences. Green regions are to be checked for duplicates.

2.3 Removal of non-interested sequences

The general workflow diagram of SeqyClean is shown in Figure 2 and described below. The workflow consists of several atomic steps: (1) Input data pre-processing; (2) Trimming poly A/T tails; (3) Vector and contaminants trimming; (4) Adapter trimming; (5) Quality trimming; (6) Extension

by overlap; (7) PCR duplicates removal; (8) Establishing clip points; (9) Generating output files and summary statistics. Stages 2, 3, 4, 5, 6, 7 are optional depending on chosen cleaning strategy.

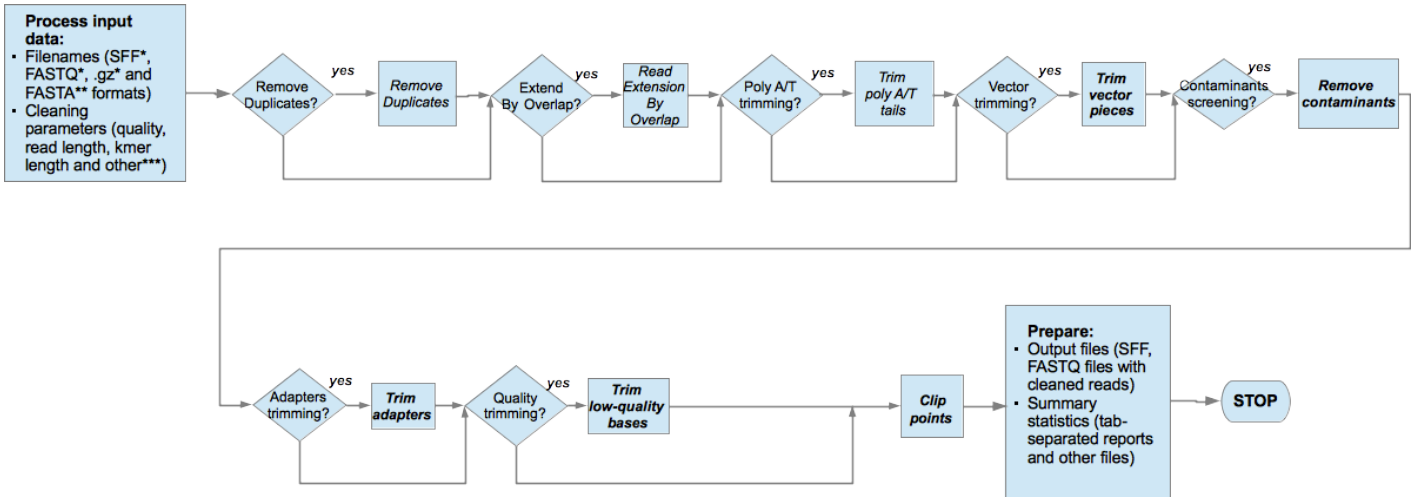

Figure 2: The workflow diagram for SeqyClean.

2.3.1 Input data pre-processing step

The input data is preprocessed on the preprocessing stage. Preprocessing stage analyzes user's inputs and then calls cleaning methods. SeqyClean works on FASTQ, SFF and FASTA-formatted files the following types:

- Raw NGS library files (format SFF or FASTQ). Use *-454* in order to enter the 454 mode for cleaning Roche 454 or Ion Torrent reads; or *-1*, *-2* or *-U* to clean Illumina paired- or single-end reads.

- Vector/contaminants sequences in FASTA-formatted files. SeqyClean employs two separate FASTA file: one for contaminants and another one for vector. Such FastaA files can contain single or multiple sequences. Use flag *-c* to tell the program to use contaminant screening and flag *-v* for vector trimming.

2.3.2 *Contaminants and vector detection and trimming*

In many cases sample library contains foreign DNA at the same time [CITE]. Contaminants in DNA library are one of the main problems in sequence assembly and mapping. If bacteria vector is used (Figure 2.3.2) during the sample library preparation, it is possible that reads still have remnants of vector genome, which can also be considered a as contaminant. In order to screen contaminants, contaminant sequences must be provided in a separate file. In a reference file, contaminant sequences are reference sequences and reads are query sequence. It is up to user to decide which types of contaminants are likely to be in a library and thereby provide appropriate reference sequences. Algorithm for contaminants and vector trimming employs exact k-mer matching and works as follows. (1) Reference sequences along with their reverse complements are sampled into consecutive k-mers of n bases (15 bases by default) long and then each k-mer is stored within a hash table. Each dictionary item stores a key, which is a k-mer string and a value that represents a vector that contains items of 2-element length: pos, id, where *pos* is a position of the first base of a k-mer in a reference sequence and id represents a record id in a provided FASTA file. Searching for a contaminant within a read (which is a query) is performed by dictionary search for each k-mer in a read. In order to speedup the computation, a distance between two consecutive k-mers. Refer to Figure 2.3.2. There are two possible cases: (1) Vector trimming when vector sequence occupies only a part of a read. When vector coordinates are approximately found, then exact coordinates are obtained by applying a pairwise alignment of a vector site in the query sequence and corresponding area in the reference vector genome. The read is discarded if a vector is more than 80% of the read. (2) In case of contaminant screening the whole read considered as contaminant and discarded as soon as it meets several consecutive successful lookups.

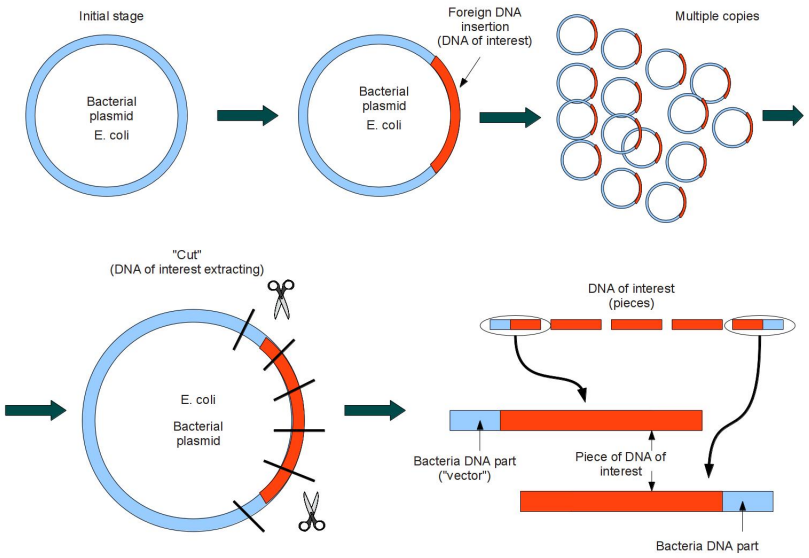

Figure 3: Bacteria vector cloning technology.

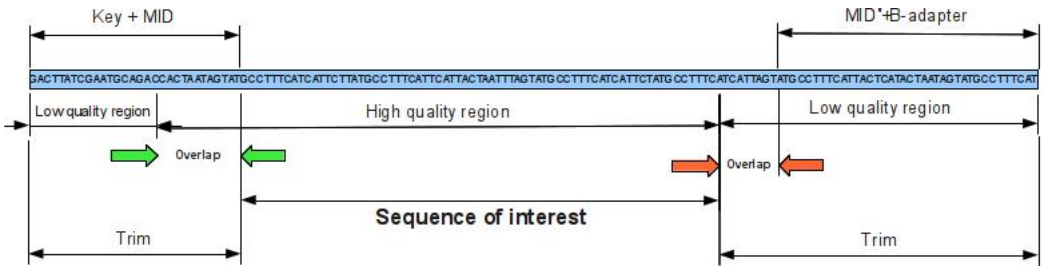

Figure 4: Artifacts within a read (Roche 454).

2.3.3 Adapter trimming

Next-Generation Sequencing technologies employ short synthetic nucleotides, adapters or primers to immobilize DNA pieces on to the inside surface of the flow cell channels. Depending on technology, adapters may be different types and attached to the ends of the DNA particle. Adapters can have a significant negative impact on genome assembly as it was previously shown in 2.3.2. Two separate algorithms are used to trim adapters: (1) Banded Pairwise alignment (Fickett, 1984)

is used for Roche 454 adapters. Banded pairwise alignment is a variation of Smith-Waterman algorithm with reduced omputational complexity by introduced a ”band” near the main diagonal in order to increase the alignment performance. (2) SSAHA kmer matching algorithm (Ning, Cox, & Mullikin, 2001), which is used to detect Illumina TruSeq adapters.

2.4 Extending Illimina reads by combining overlapping reads

Illumina paired-end sequencing technology produces a very large amount of data, which is a set of relatively short reads. In spite of high quality and significant amount (up to 100 million) paired-end reads, their length (100-250 nucleotides) still makes genome assembly a difficult and challenging task. There is a potential benefit to take advantage of paired-end paradigm to extend the reads that may potentially overlap into one single-end sequence with length up to almost twice as the length of a pair. This is can be considered as extending paired-end reads by combining overlapping pairs and can be used before conducting the downstream analysis (Magoč & Salzberg, 2011). We adopted that algorithm for SeqyClean. The algorithm works as follows:

1. Align the pair of reads so that they overlap completely by the full length of the shorter read.
2. Repeat while the overlap is longer than minoverlap:
    a Calculate the overlap length.
    b Calculate the score for the overlap as the ratio between the number of mismatches and the overlap length.
    c If the score of the overlap is equal or bigger that the pre-defined mismatch threshold:
        – Calculate the new quality values of all bases within the overlap by taking maximum quality scores if bases are equal. Otherwise assign the new quality value as a subtraction of quality scores.

– Save the overlap and progress toward the next pair.

d Otherwise slide the reads apart by one base, reducing the overlap by one.

3. If the score is less than the mismatch threshold, report that no good overlap was found.

Overlapping reads have the ability of detection short parts of TruSeq adapters (9-14 bases) that may be missed by alignment. We propose an algorithm that increases the sensitivity of TruSeq (Illumina) adapter trimming. Our assumption is that if an adapter is found on one of the end of the read, the corresponding adapter should be found at the end of another read. In this case, by applying overlap alignment is possible to have a situation where adapters will be on the both end of the consensus sequence as it is shown in Figure 5.

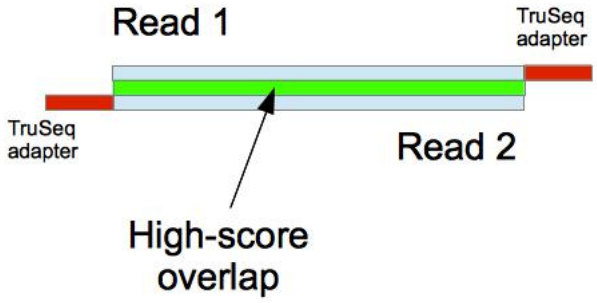

Figure 5: Adapter trimming by overlap. Green region has high overlap scores and this makes us to assume that the short remnats of TruSeq adapters exist on the both of the ends.

2.5 Quality trimming (Description of the algorithm for trimming poor quality sequence)

Errors in reads are unavoidable and assumption that reads are error-free is often misleading. In fact, majority of reads in DNA library have base-call errors and it is important to obtain as less errors as possible. The quality of nucleotide bases decreases toward the end of the read 2.5. To obtain nucleotide sequence and corresponding quality scores, a program known as a base caller converts the raw output from DNA sequencer into nucleotide bases, each of which is typically

assigned a quality score that estimates the probability that the base has been sequenced correctly. Quality scores are needed to trim the low quality ends of a read (Roche 454) and for determination a consensus sequence. SeqyClean utilizes quality trimming approach described in (Chou & Holmes, 2001) and employs the scoring model with an integer scale derived from the error probability p (where p is the probability that a particular base has been read correctly) via the formula $Q = -10log_{10}p$, a scoring scheme named Phred scheme. Nucleotides and corresponding Phred quality scores with other metadata are stored in ascii – delimited text FastQ format which Illumina employs (files are usually compressed with gzip) and binary Sff format which is used by Roche 454. Having the raw data, algorithm discards low-quality regions that may exist within a read by computing average quality of a window. Windows with quality lower than pre-defined threshold are trimmed. The minimum read length to keep is by default 50 nucleotide bases. In order to be able to process short Illumina reads with length about 30 bases, the size of the window can be adjusted.

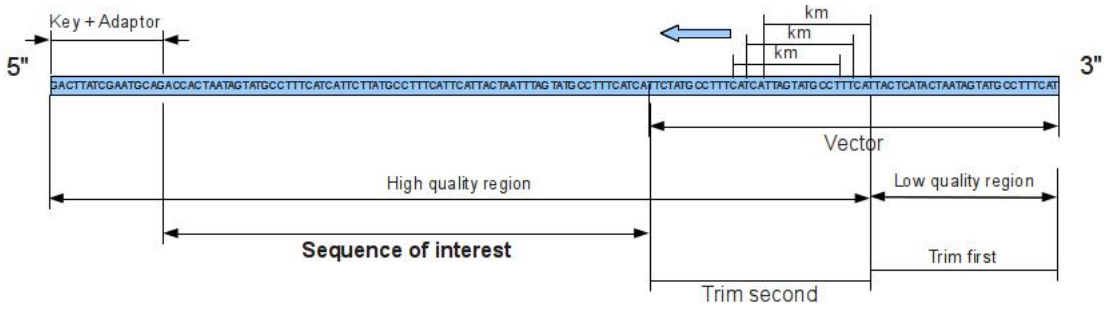

Figure 6: Quality trimming.

2.6 Poly A/T tails trimming (Description of the algorithm for trimming poly A/T tails)

Expressed sequence tags (ESTs), are single-stranded DNA sequencing reads made from complementary DNA (cDNA) clone libraries derived from a known source (cells, tissues, or organs from different organisms). Sequencing a large number of these clones from such a library allows one to sample the set of expressed genes, or transcripts, in that particular tissue and experimental state,

thus providing a snapshot of the active genes under those defined conditions (Nagaraj, Gasser, & Ranganathan, 2007).

GenBank: DY008075.1

GenBank FASTA

IDENTIFIERS

dbEST Id: **35691756**
EST name: 19ACACYS_UP_022_A11_29OCT2004_095
GenBank Acc: DY008075
GenBank gi: 119423037

CLONE INFO
Clone Id: (5')
DNA type: cDNA

PRIMERS
PolyA Tail: Unknown

SEQUENCE

TGGTACGGTCAGATGCTTGCTAAAGGAGAAATAAATAGAGACATGGGTGATAGTATAAGC
GGAAAGGGAATGATTCAGGGTGTTTCTGCAGTGGGAGCGTTTTACCAACTGCTTAGTCAG
TCCAGCCTAAGTATATTGCATTCTGAAGAGAAGAAACCTGTGGCTCCGGTTGAATCATGT
CCTATTTTGAAAACACTCTACAAGATACTCATCACAAGAGAACAATCAACACAAGCGATT
CTGCAAGCATTAAGGGATGAAACACTGAATGACCCAAGAGACAGGATTGAGATTGCACAG
AGCCATGCATTCTACAGGCCTTCCCTTCTAGATCAGCCTTGATTAGTCTGTCATGGCTCA
TAATCCGAACTTCTAAGATCTTACTTGTGCAAACTGCAGATTCTGCTATGTTAAACATCA
TGTCTTAAAATTGATTGTTGTTCAGCCAAAAAAAAAAAAAAAAAAAAAAAAAAAAACAT
GTC

Entry Created: Dec 15 2006
Last Updated: Dec 15 2006

Figure 7: Poly-A tails occurs within a sequenced tag.

The raw DNA sequences obtained from an EST library contains poly-A/T tags that have to be removed before conducting the downstream analysis (see Figure 7). SecyClean provides this service by searching for the first minimum occurrence of 10 bases or longer poly-T fragment within the first initial search range of 50 bases inside the good region, then attempts to extend from this initial poly-T seed toward the center of the sequence, allowing no more than maximum error (three by default) mismatches between every min span (10) consecutive T bases in the search region. This can be done quickly since it is a linear time algorithm. The same algorithm is uses to the poly-A tail trimming at the other end of the sequence.

2.6.1 *Establishing clip points*

Clip point is a position within a read where all bases are discarded after this position. SeqyClean establishes the most conservative clip points, in other words it chooses the right-most clip point from the left and left-most clip point from the right side of the read. This implies, for example, that if quality clip points are located inside of vector clip points, SeqyClean uses the first ones as the actual trim points of the read (Figure 8).

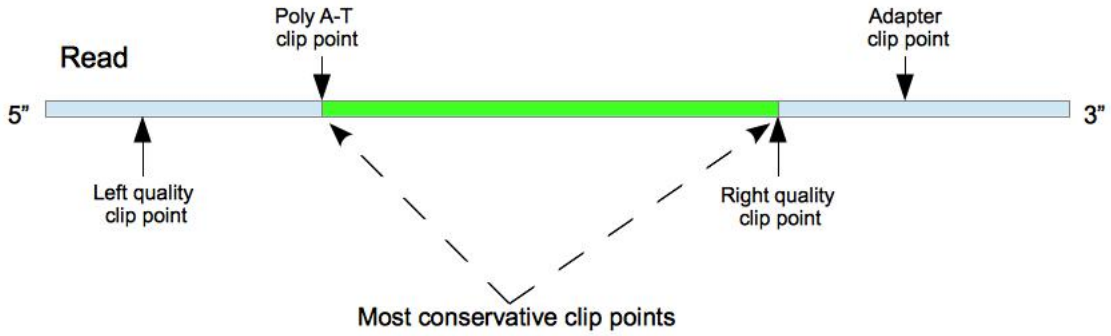

Figure 8: Establishing the most-conservative clip points. Green region is to be kept.

2.6.2 *Output files and summary statistics*

The output files are FastQ or SFF-formatted files containing either trimmed reads (FastQ-file) or reads with new established clip points (SFF-file). In addition, detailed statistics for each read and summary statistics are generated both in human- (TXT) and machine-readable (TSV) formats.

2.6.3 *Supported RLMIDs*

The set of supported Roche 454 RL MIDs is shown in Table 6.

3 Acknowledgements

This work was supported by IBEST COBRE, grant NIH/NCRR P20RR16448 and the University Research Office at the University of Idaho.

References

454 Life Science, A. R. C. (2008). *Point-and-click tools for assembly, mapping and amplicon variant analysis.* Retrieved from http://454.com/products/analysis-software/index.asp

Chou, H., & Holmes, M. H. (2001). DNA sequence quality trimming and vector removal. *Bioinformatics/computer Applications in The Biosciences*, *17*, 1093–1104. doi: 10.1093/bioinformatics/17.12.1093

Criscuolo, A., & Brisse, S. (2013). Alientrimmer: A tool to quickly and accurately trim off multiple short contaminant sequences from high-throughput sequencing reads. *Genomics*(0), -. Retrieved from http://www.sciencedirect.com/science/article/pii/S0888754313001481 doi: http://dx.doi.org/10.1016/j.ygeno.2013.07.011

Eckalbar, W., Hutchins, E., Markov, G., Allen, A., Corneveaux, J., Lindblad-Toh, K., ... Kusumi, K. (2013). Genome reannotation of the lizard anolis carolinensis based on 14 adult and embryonic deep transcriptomes. *BMC Genomics*, *14*(1), 49. Retrieved from http://www.biomedcentral.com/1471-2164/14/49 doi: 10.1186/1471-2164-14-49

Falgueras, J., Lara, A., Pozo, N. F., Canton, F., Trabado, G. P., & Claros, M. G. (2010, January 20). SeqTrim: a high-throughput pipeline for pre-processing any type of sequence read. *BMC Bioinformatics*, *11*(1), 38+. Retrieved from http://dx.doi.org/10.1186/1471-2105-11-38 doi: 10.1186/1471-2105-11-38

Fickett, J. W. (1984). Fast optimal alignment. *Nucleic Acids Research*, *12*, 175–179. doi: 10.1093/nar/12.1Part1.175

Gurevich, A., Saveliev, V., Vyahhi, N., & Tesler, G. (2013). Quast: quality assessment tool for genome assemblies. *Bioinformatics*, *29*(8), 1072-1075. Retrieved from http://bioinformatics.oxfordjournals.org/content/29/8/1072.abstract doi: 10.1093/bioinformatics/btt086

Kim, U.-J., Shizuya, H., Sainz, J., Garnes, J., Pulst, S. M., de Jong, P., & Simon, M. I. (1995). Construction and utility of a human chromosome 22-specific fosmid library. *Genetic Analysis: Biomolecular Engineering*, *12*(2), 81 - 84. Retrieved from http://www.sciencedirect.com/science/article/pii/1050386295001220 doi: http://dx.doi.org/10.1016/1050-3862(95)00122-0

Lab, H. (n.d.). *FASTX Toolkit.* Retrieved from http://hannonlab.cshl.edu/fastx_toolkit/

Langmead, B., & Salzberg, S. L. (2012). Fast gapped-read alignment with bowtie 2. *Nat Meth*, *9*(4). Retrieved from http://dx.doi.org/10.1038/nmeth.1923 doi: 10.1038/nmeth.1923

Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., ... Subgroup, . G. P. D. P. (2009). The sequence alignment/map (sam) format and samtools. *Bioinformatics*. Retrieved from http://bioinformatics.oxfordjournals.org/content/early/2009/06/08/bioinformatics.btp352.abstract doi: 10.1093/bioinformatics/btp352

Magoč, T., & Salzberg, S. L. (2011). Flash: fast length adjustment of short reads to improve genome assemblies. *Bioinformatics*, *27*(21), 2957-2963. Retrieved from http://bioinformatics.oxfordjournals.org/content/27/21/2957.abstract doi: 10.1093/bioinformatics/btr507

Nagaraj, S. H., Gasser, R. B., & Ranganathan, S. (2007). A hitchhiker's guide to expressed sequence tag (est) analysis. *Briefings in Bioinformatics*, *8*(1), 6-21. Retrieved from http://bib.oxfordjournals.org/content/8/1/6.abstract doi: 10.1093/bib/bbl015

Ning, Z., Cox, A. J., & Mullikin, J. C. (2001). SSAHA: A Fast Search Method for Large DNA Databases. *Genome Research*, *11*, 1725–1729. doi: 10.1101/gr.194201

Ralser, M., Kuhl, H., Ralser, M., Werber, M., Lehrach, H., Breitenbach, M., & Timmermann, B. (2012). The saccharomyces cerevisiae w303-k6001 cross-platform genome sequence: insights into ancestry and physiology of a laboratory mutt. *Open Biology*, *2*(8). Retrieved from http://rsob.royalsocietypublishing.org/content/2/8/120093.abstract doi: 10.1098/rsob.120093

Scheetz, T. E., Trivedi, N., Roberts, C. A., Kucaba, T., Berger, B., Robinson, N. L., ... Casavant, T. L. (2003). ESTprep: Preprocessing CDNA Sequence Reads. *Bioinformatics/computer Applications in The Biosciences*, *19*, 1318–1324. doi: 10.1093/bioinformatics/btg159

Shizuya, H., Birren, B., Kim, U. J., Mancino, V., Slepak, T., Tachiiri, Y., & Simon, M. (1992). Cloning and stable maintenance of 300-kilobase-pair fragments of human dna in escherichia coli using an f-factor-based vector. *Proceedings of the National Academy of Sciences*, *89*(18), 8794-8797. Retrieved from http://www.pnas.org/content/89/18/8794.abstract doi: 10.1073/pnas.89.18.8794

White, J. R., Roberts, M., Yorke, J. A., & Pop, M. (2008). Figaro: a novel statistical method for vector sequence removal. *Bioinformatics/computer Applications in The Biosciences*, *24*, 462–467. doi: 10.1093/bioinformatics/btm632

Table 2: Options for cleaning 454 and Ion Torrent libraries

-v vector_file	This option does vector trimming. If you choose this option, the program assumes the file of vector sequences provided in vector_file. This file must be given in FASTA format. Example: ./seqyclean -v vectors.fa -454 in.sff -o Test
-c file_of_contaminants	This option is used for contaminants screening. If you choose this option, the program assumes the reference genome provided in file_of_contaminants. This file must be given in FASTA format. When SeqyClean recognizes contaminants in the sequence, the whole sequence gets discarded. Note: contaminant reference sequences must be provided! Example: ./seqyclean -c contaminants.fa -454 in.sff -o Test
-m file_of_RL_MIDS	This option works in SFF mode only. Use this option to provide your own RLMIDs. SeqyClean will use them and will not use those provided by default. Example: ./seqyclean -m file_of_custom_RL_MIDS -454 in.sff -o Test
-k k_mer_size	Use this option in order to specify a size of k-mer. Default k-mer size is 15 bases. Example: ./seqyclean -k 18 -454 in.sff -o Test
-kc k_mer_size	Special k-mer size for contaminant screening. Use this option only if you want to have different k-mer sizes for contaminant dictionary. Example: ./seqyclean -kc 25 -454 in.sff -o Test
-f k-mer overlap	For Roche 454 only. This option is intended to tweak an overlap between two consecutive k-mers. By default the length of overlap it is set to 1 bp. Example: ./seqyclean -f 10 -454 in.sff -o Test
-t number_of_threads	Specifies a number of threads in order to take advantage from using a multicore system. Example: ./seqyclean -t 16 -454 in.sff -o Test
-qual [max_avg_error max_error_at_ends -w0 <value> -w1 <value>]	LUCY parameters for quality trimming. Default values for max_avg_error and max_error_at_ends are [20 20]. "w0" and "w1" are window parameters. Examples: ./seqyclean -qual -454 in.sff -o Test ./seqyclean -qual 19 -w0 23 -454 in.sff -o Test ./seqyclean -qual 28 -w0 40 -w1 5 -454 in.sff -o Test ./seqyclean -qual 25 30 -454 in.sff -o Test
--qual_only	Use --qual_only parameter if you want to do only quality trimming. Example: ./seqyclean --qual_only -qual -454 in.sff -o Test
--fastq	If input is given in SFF format, by default the output will be also in SFF format. Use this option if you want to have FASTQ format on the output in addition to SFF. Example: ./seqyclean --fastq -454 in.sff -o Test
--keep_fastq	Use this option only if you want to keep original FASTQ file from your input SFF Example: ./seqyclean --keep_fastq -454 in.sff -o Test
-minimum_read_length value	Use this option in order to define the minimum number of base pairs when read is still considered as acceptable. If after the cleaning process the read has a length which is less than minimum_read_length parameter, the read will be discarded. By default, the minimum_read_length is set to 50 base pairs. Example: ./seqyclean -minimum_read_length 100 -454 in.sff -o Test
-polyat [cdna] [cerr] [crng]	This option provides trimming of poly A/T tails from nucleotide sequences. cdna – tail length (10 by default); cerr – maximum number of errors per tail (3 by default); crng – range to search poly A/T tails (50 by default) Examples: ./seqyclean -polyat -454 in.sff -o Test ./seqyclean -polyat 12 5 67 poly_test.fastq.gz -o Test_polyAT
--no_ts_adapter_trim	This option turns off adapter trimming.

Table 3: Paired- and single-end main arguments

`-1 input_filename_R1`
 	
`-2 input_filename_R2`
	The filenames of the library to be cleaned. Must be in FASTQ formats only (the program also accepts zipped (.gz) FASTQ files).

`-U input_filename`	The filenames of the library to be cleaned. Can be in FASTQ formats only (the program also accepts .gz files).
`-o output_prefix`	The files produced will start with the output_prefix followed by some formatted ending (see section "Output files: naming convention")

Table 4: Illumina options

--shuffle	With this option SeqyClean will combine output paired-end libraries into one single file named <output_prefix>_shuffled.fastq. However, SeqyClean still does keep single-end reads (reads without corresponding pairs) in <output_prefix>_SE.fastq file. Example: ./seqyclean --shuffle -1 R1.fastq -2 R2.fastq -o Test
-v vector_file	This option is used for vector trimming. If you choose this option, the program assumes the reference genome provided in vector_file . This file must be given in FASTA format. Example: ./seqyclean -v vectors.fa -1 R1.fastq -2 R2.fastq -o Test
-c file_of_contaminants	This option is used for contaminants screening. If you choose this option, the program assumes the reference genome provided in file_of_contaminants . This file must be given in FASTA format. When SeqyClean recognizes contaminants in the sequence, the whole sequence gets discarded. Example: ./seqyclean -c contaminants.fa -1 R1.fastq -2 R2.fastq -o Test
-k k_mer_size	Use this option in order to specify a size of k-mer. Default size is 15 bases. In Illumina mode this option defines a size of kmer that will be used as a dictionary word size. Example: ./seqyclean -k 14 -1 R1.fastq.gz -2 R2.fastq.gz -o Test
-kc k_mer_size	Special k-mer size for contaminant screening. Use this option only if you want to have different k-mer sizes for contaminant dictionary. Sometimes this option is useful because it prevents false detection of contaminants when program discards too many reads. Example: ./seqyclean -kc 31 -1 R1.fastq.gz -2 R2.fastq.gz -o Test
-qual max_avg_error max_error_at_ends	LUCY parameters for quality trimming. if "-qual" is set that means you have to provide max_avg_error and max_error_at_ends. Otherwise default values [20 20] will be used. Examples: ./seqyclean -qual -1 R1.fastq.gz -2 -1 R2.fastq.gz -o Test ./seqyclean -qual 30 25 -1 R1.fastq.gz -2 R2.fastq.gz -o Test
--qual_only	Use --qual_only parameter if you want to do only quality trimming. Example: ./seqyclean --qual_only -qual -1 R2.fastq.gz -2 R2.fastq.gz -o Test
-minimum_read_length value	Use this option in order to define the minimum number of base pairs when read is still considered as acceptable. If after cleaning process the read has length which is less than minimum_read_length parameter such read will be discarded. By default, the minimum_read_length is set to 50 base pairs. Note: in this case no adapter/vector/contaminants cleaning is performed. Example: ./seqyclean -minimum_read_length 100 -1 R1.fastq.gz -2 R2.fastq.gz -o Test
-polyat [cdna] [cerr] [crng]	This option provides trimming of poly A/T tails from nucleotide sequences. Parameters: cdna – tail length (10 by default) cerr – maximum number of errors per tail (3 by default) crng – range to search poly A/T tails (50 by default) Examples: ./seqyclean -polyat -454 in.sff -o Test ./seqyclean -polyat 15 4 55 poly_test.fastq.gz -o Test_polyAT
-overlap <value>	This option provides overlap of read 1 and read 2. The consensus single-end sequences go to the ¡output_prefix¿_SE.fastq file. Parameter: <value¿ – the minumum length of overlap in bases. By default it is set to 10 bases.
-ot value	This option sets the similarity threshold for adaprer trimming by overlap. By default its value is set to 0.9.
-adapter_length value	This option sets the maximum adapter length for adaprer trimming by overlap. By default its value is set to 40 bases.
--ow	This option allows SeqyClean overwrite output files.
--no_ts_adapter_trim	This option turns off adapter trimming.

Table 5: Quality parameter values along with corresponding real error

Phred value for the pair {maximum_average_error, maxim_error_at_ends}	Corresponding error probability	Phred value for the pair {maximum_average_error, maxim_error_at_ends}	Corresponding error probability
1,1	0.2057	21,21	0.9921
2,2	0.3690	22,22	0.9937
3,3	0.4988	23,23	0.9950
4,4	0.6019	24,24	0.9960
5,5	0.6838	25,25	0.9968
6,6	0.7488	26,26	0.9975
7,7	0.8005	27,27	0.9980
8,8	0.8415	28,28	0.9984
9,9	0.8741	29,29	0.9987
10,10	0.9000	30,30	0.9990
11,11	0.9206	31,31	0.9992
12,12	0.9369	32,32	0.9994
13,13	0.9499	33,33	0.9995
14,14	0.9602	34,34	0.9996
15,15	0.9684	35,35	0.9997
16,16	0.9749	36,36	0.9997
17,17	0.9800	37,37	0.9998
18,18	0.9842	38,38	0.9998
19,19	0.9874	39,39	0.9999
20,20	0.9900	40,40	0.9999

Table 6: Supported RLMIDs by default

#	Left MID	Right MID	#	Left MID	Right MID
RL1	ACACGACGACT	AGTCGTGGTGT	RL19	ATAGTATACGT	ACGTATAGTAT
RL2	ACACGTAGTAT	ATACTAGGTGT	RL20	CAGTACGTACT	AGTACGTGCTG
RL3	ACACTACTCGT	ACGAGTGGTGT	RL21	CGACGACGCGT	ACGCGTGGTCG
RL4	ACGACACGTAT	ATACGTGGCGT	RL22	CGACGAGTACT	AGTACTGGTCG
RL5	ACGAGTAGACT	AGTCTACGCGT	RL23	CGATACTACGT	ACGTAGTGTCG
RL6	ACGCGTCTAGT	ACTAGAGGCGT	RL24	CGTACGTCGAT	ATCGACGGACG
RL7	ACGTACACACT	AGTGTGTGCGT	RL25	CTACTCGTAGT	ACTACGGGTAG
RL8	ACGTACTGTGT	ACACAGTGCGT	RL26	GTACAGTACGT	ACGTACGGTAC
RL9	ACGTAGATCGT	ACGATCTGCGT	RL27	GTCGTACGTAT	ATACGTAGGAC
RL10	ACTACGTCTCT	AGAGACGGAGT	RL28	GTGTACGACGT	ACGTCGTGCAC
RL11	ACTATACGAGT	ACTCGTAGAGT	RL29	ACACAGTGAGT	ACTCACGGTGT
RL12	ACTCGCGTCGT	ACGACGGGAGT	RL30	ACACTCATACT	AGTATGGGTGT
RL13	AGACTCGACGT	ACGTCGGGTCT	RL31	ACAGACAGCGT	ACGCTGTGTGT
RL14	AGTACGAGAGT	ACTCTCGGACT	RL32	ACAGACTATAT	ATATAGTGTGT
RL15	AGTACTACTAT	ATAGTAGGACT	RL33	ACAGAGACTCT	AGAGTCTGTGT
RL16	AGTAGACGTCT	AGACGTCGACT	RL34	ACAGCTCGTGT	ACACGAGGTGT
RL17	AGTCGTACACT	AGTGTAGGACT	RL35	ACAGTGTCGAT	ATCGACAGTGT
RL18	AGTGTAGTAGT	ACTACTAGACT	RL36	ACGAGCGCGCT	AGCGCGCGCGT