Data preprocessing

Not only sequencing, but also data analysis costs money. Analyzing poor data wastes CPU time and interpreting the results from poor data wastes people time. The quality control step often shows that the data needs to be preprocessed before any downstream data analysis. The necessary data preprocessing steps highly depend on the type of library being sequenced (whole genome, transcriptome, 16S, metagenome, ...) and on the type of sequencing technology used to generate the data. The following guide should ensure that the data used for downstream analysis is not compromised of low-quality sequences, sequence artifacts, or sequence contamination that might lead to erroneous conclusions. However, there is no ''one-size-fits-all'' solution and each user must make informed decisions as to the appropriate parameters used for preprocessing.

Programs used:

TagCleaner - http://tagcleaner.sourceforge.net

PRINSEQ - http://prinseq.sourceforge.net

DeconSeq - http://deconseq.sourceforge.net

All programs used are freely available and provide a web-based and a standalone version.

Content:

Necessary resources for web versions

Hardware

Computer connected to the Internet

Software

Up-to-date Web browser (Firefox, Safari, Chrome, Internet Explorer, …)

Files

FASTA file with sequence data

QUAL file with quality scores (if available)

FASTQ file (as alternative format)

Notes

TagCleaner does also trim the quality data if provided as input (FASTQ format). DeconSeq does not make use of quality data and therefore does not require quality data as input. The web versions allow users to easily share and discuss the results with other people without transferring large data files. Results will be stored for one week, if not otherwise requested, on the web server using a unique identifier displayed on the result page.

Necessary resources for standalone versions

Hardware

Computer with a Linux/Unix or Mac OS X operating system

Software

Perl 5 (or higher)

TagCleaner standalone (available at http://tagcleaner.sourceforge.net)

PRINSEQ lite (available at http://prinseq.sourceforge.net)

DeconSeq standalone (available at http://deconseq.sourceforge.net)

Files

FASTA file with sequence data

QUAL file with quality scores (if available)

FASTQ file (as alternative format)

Notes

Standalone versions are preferable if multiple datasets have to be processed in a similar manner.

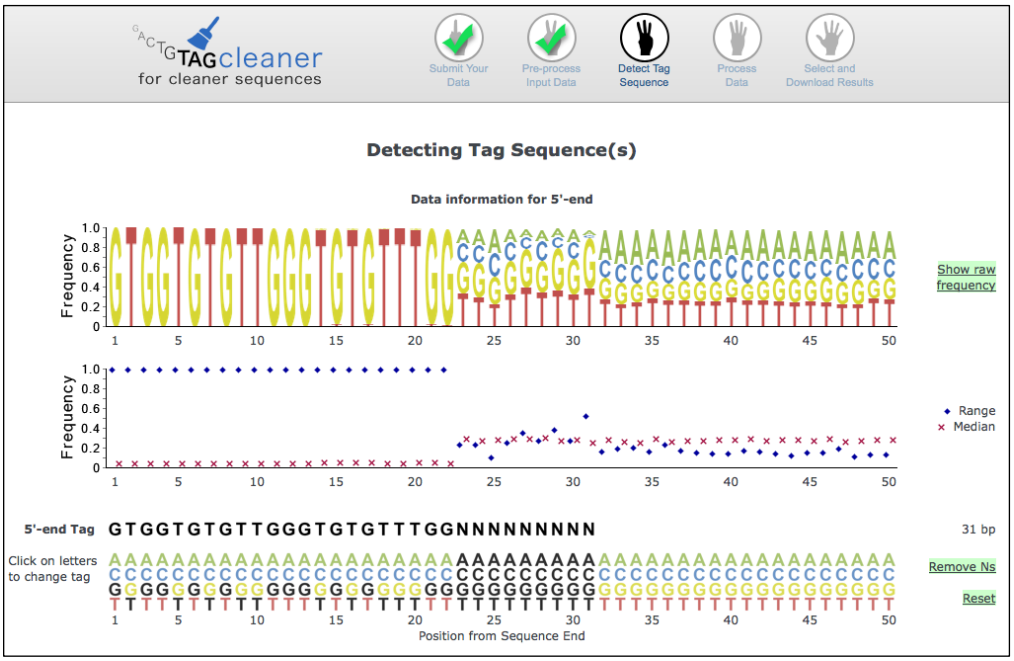
Upload data to the TagCleaner web version

To upload a new dataset in FASTA or FASTQ format to TagCleaner, follow these steps:

1. Go to http://tagcleaner.sourceforge.net
2. Click on "Use TagCleaner" in the top menu on the right (the latest TagCleaner web version should load)
3. Select your FASTA or FASTQ file
4. Select trim mode (trim tag sequences from both ends, 5'-end only, or 3'-end only)
5. Specify tag sequence for 5'-end and/or 3'-end (if available)
6. Click "Submit"

Notes

If the tag sequences are not available or are unknown, leave the fields free and TagCleaner will try to find the tag sequence. The predicted tag sequence is provided to the user and can be modified before any further steps are performed (see below).



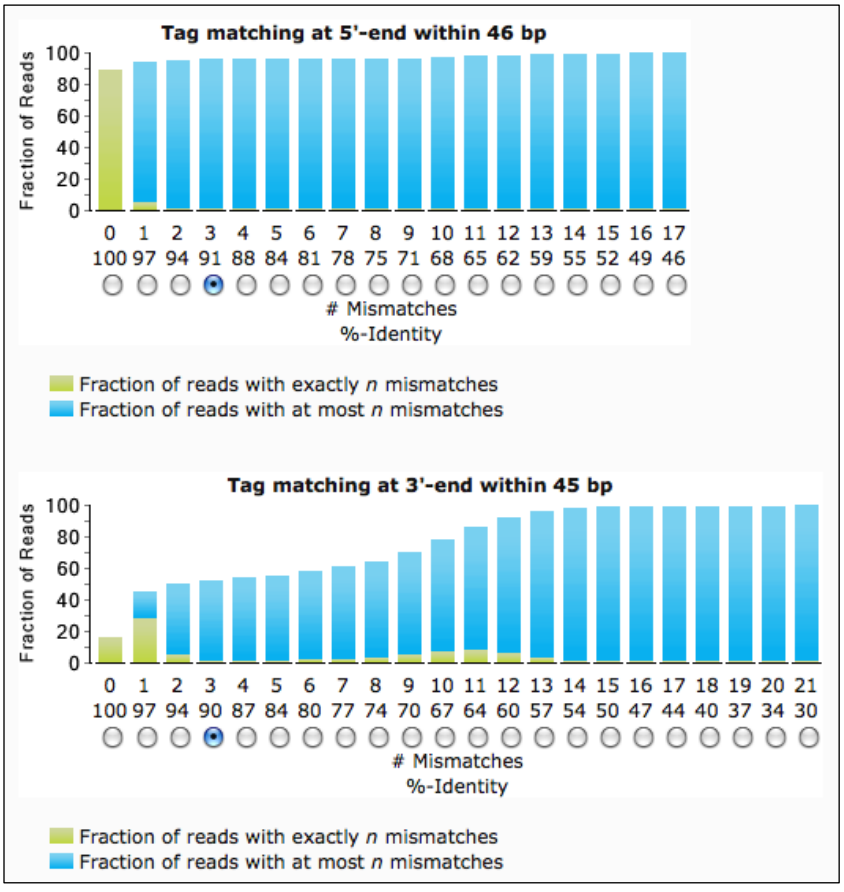
TagCleaner does not require the setting of filter parameters (such as maximum number of mismatches) before the data is processed. Instead, the filter parameters are set after the data is processed, which allows the user to choose parameters appropriate for their dataset and does not require them to submit and process the same data with modified parameters for several times.

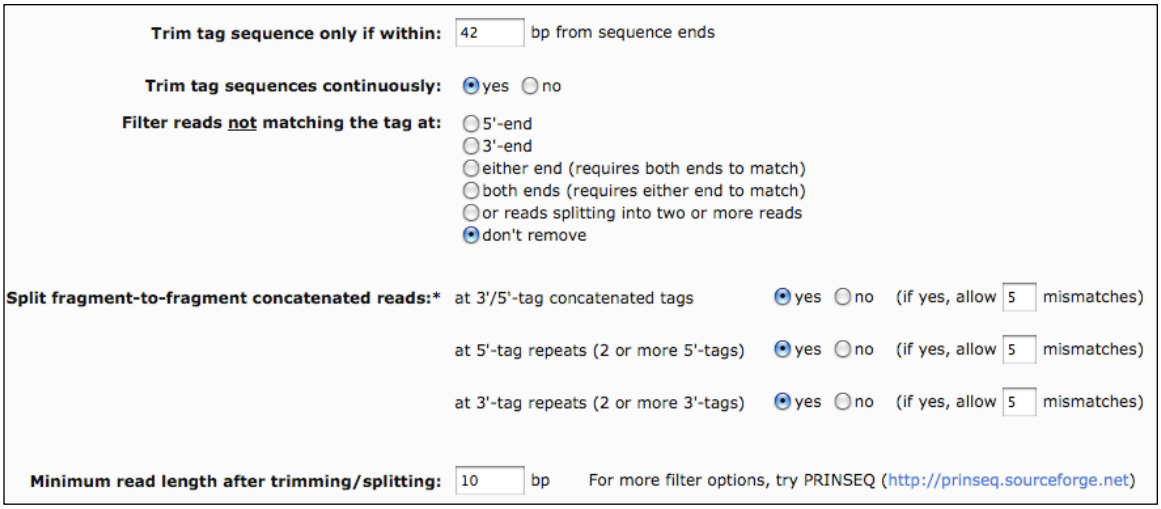
Tag sequence trimming

Tag sequence trimming should be performed before quality trimming and sequence dereplication. The trimming of low-quality bases at the ends might truncate the tag sequence and reduce the ability to recognize the remainder of the tag sequence. In those cases, large parts of the tag sequences might still remain for further analysis and data processing steps. Dereplication before trimming may miss duplicated sequences due to variations in the tag sequences that will be trimmed off later and would therefore require an additional dereplication step after the trimming.

The algorithm implemented in TagCleaner for the automatic detection of tag sequences assumes the randomness of a typical metagenome. Datasets that do not contain random sequences from organisms in an environment, but rather contain, for example, 16S data may cause incorrect detection of the tag sequences. However, the tag sequences will most likely be over-predicted and can be redefined by the user prior to data processing.

The independent definition of maximum allowed mismatches for the 5'- and 3'-end of the reads accounts for the differences in tag sequences due to the limitations of the sequencing method used to generate the datasets. The 3'-end will in most cases show a lower number of matching tag sequences with low number of mismatches due to incomplete or missing tags at the ends of incompletely sequenced fragments.





Trim tag sequence only if within

The sequence of the tag could occur not only at the sequence end, but also at any other position of the sequence. To assure that only tags are trimmed, the tag sequences can be defined to occur only at the ends allowing a certain number of variable bases. The default value for tag sequence of at least 10 bp is 1.5 times the tag sequence length.

Trim tag sequence continuously

The continuous trimming of tag sequences from the ends allows filtering of sequences mainly consisting of concatenated tag sequences.

Remove reads not matching tag sequence

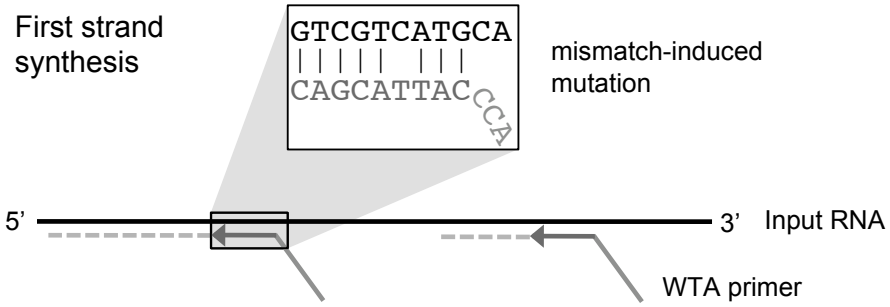
This filter can be applied to remove sequences that do not contain any tag sequence (with the defined number of maximum mismatches). This feature was originally designed to separate MID tagged sequences before the MID tags were made publicly available. The feature can also be used to further investigate sequences without a matching tag sequence.

Split fragment-to-fragment concatenated reads

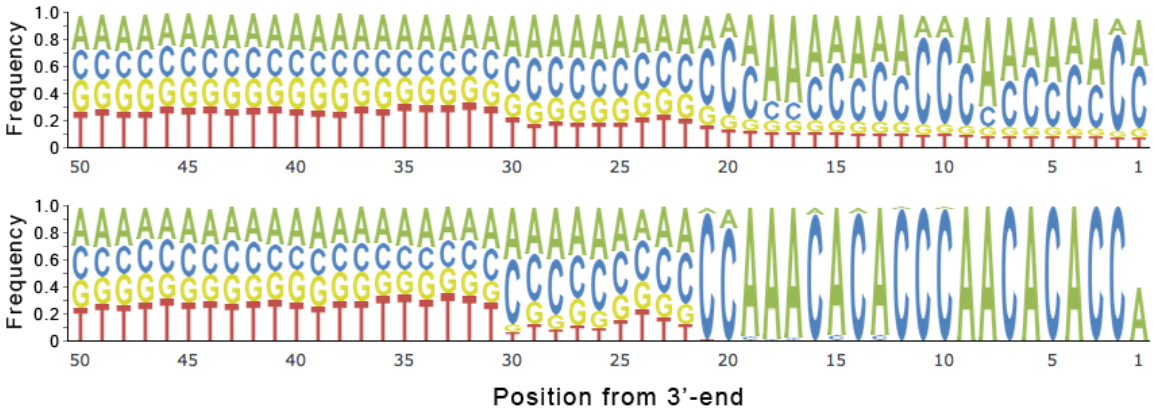
TagCleaner provides an additional feature of detecting and splitting *fragment-to-fragment concatenations*. This important pre-processing step removes tag contaminations inside the sequences, which may allow, for example, more accurate assemblies. The concatenated fragments may additionally present a source of error for annotation and taxonomic assignments, since fragments from different organisms may not be assigned correctly when concatenated.

Notes

TagCleaner is able to detect the quasi-random 3'-end of whole transcriptome amplification (WTA) primers. The user has the option whether or not to trim this part of the tag sequence by simply adding or removing the letter N from the end of the tag sequence. However, users are advised to trim the complete tag sequence. It is important to trim the random parts in order to account for mismatch-induced mutations that often happen when primers anneal to similar (but not identical) sequences with high enough affinity for binding (see below). Therefore, one cannot be certain that this part of the tag sequence represents the actual sequence of the sample.



The tag sequence prediction uses filtered base frequencies instead of raw base frequencies. This allows a more accurate prediction as it accounts for incomplete and shifted tag sequences. The following example shows the raw base frequencies (top) and the filtered base frequencies (bottom) for the 3'-ends.



Standalone version options

The standalone version does not provide graphical outputs, but all the functionality you might be used to from the web version. Starting with version 0.9, the web version is using the standalone version for all the calculations in the backend. The readme file contains information on the usage of the standalone version.

The following table contains all the options available for the standalone version.

Option/flag	Description	Default	Range
`-help` or `-h`	Print the help message; ignore other arguments		
`-man`	Print the full documentation; ignore other arguments		
`-version`	Print program version; ignore other arguments		
`-verbose`	Prints status and info messages during processing		
Input options			
`-fastq`	Input file in FASTQ format that contains the sequence and quality data		`FILE`
`-fasta`	Input file in FASTA format that contains the sequence data		`FILE`
`-qual`	Input file in QUAL format that contains the quality data		`FILE`
Output options			
`-out_format`	Output format 1 (FASTA only), 2 (FASTA and QUAL), 3 (FASTQ)	same as input	`[1,2,3]`
`-out`	By default, the output files are created in the same directory as the input file containing the sequence data with an additional "_tagcleaner_XXXX" in their name (where XXXX is replaced by random characters to prevent overwriting previous files). To change the output filename and location, specify the filename using this option. Example: use "file_passed" to generate the output file file_passed.fasta (fasta output) in the current directory		`STRING`

Option/flag	Description	Default	Range
-line_width	Sequence characters per line. Use 0 if you want each sequence in a single line. Use 80 for line breaks every 80 characters. Note that this option only applies to FASTA output files, since FASTQ files store sequences without additional line breaks.	60	INT
-stats	Prints the number of tag sequences matching for different numbers of mismatches. In combination with -split, the number of sequences with fragment-to-fragment concatenations is printed as well. Cannot be used in combination with -predict and will not perform any trimming. The output values are separated by tabs with the header line: "#Param Mismatches_or_Splits Number_of_Sequences Percentage Percentage_Sum". Cannot be used in combination with -predict and require -tag5 or -tag3.		
-predict	Use this option to have TagCleaner predict the tag sequences. It will attempt to predict the tag at either or both sites, if possible. The algorithm implemented for the tag prediction assumes the randomness of a typical metagenome. Datasets that do not contain random sequences from organisms in an environment, but rather contain, for example, 16S data may cause incorrect detection of the tag sequences. However, the tag sequences will most likely be over-predicted and can be redefined by the user prior to data processing. The tag sequence prediction uses filtered base frequencies instead of raw base frequencies. This allows a more accurate prediction as it accounts for incomplete and shifted tag sequences. The output values are separated by tabs with the header line: "#Param Tag_Sequence Tag_Length Percent_Explained". Cannot be used in combination with -tag3 or -tag5 or -stats.		
-log	Log file to keep track of parameters, errors, etc. The log file name is optional. If no file name is given, the log file name will be "inputname.log". If the log file already exists, new content will be added to the file.		FILE

Option/flag	Description	Default	Range
-info	This option will provide the trimming and splitting information in the header line after the sequence identifier. The following information is given and separated by a single space: initial length, length after trimming, 5'-end trimming position, 3'-end trimming position, number of mismatches at 5'-end, number of mismatches at 3'-end and number of sequences after splitting. In case of a splitting event, the number of mismatches at the 5'- and 3'-end will be the number of mismatches of the concatenated tags.		
-nomatch	This option allows to filter sequences that do not match the tag sequence at the ends or do not contain inner tags within the maximum number of allowed mismatches. The following options allow to filter reads not matching the tag at: (1) 5'-end, (2) 3'-end, (3) either end - requires both ends to match, (4) both ends - requires either end to match, or (5) reads splitting into two or more reads.		[1,2,3,4,5]
-minlen	Minimum read length of trimming and/or splitting		INT
-filtered	Output the sequences that would be filtered out instead of the sequences passing the filters. This includes sequences that e.g. are tag sequence repeats or do not fulfill the -nomatch option.		

Trim / Split options

-tag5	Tag sequence at 5'-end. Use option -predict if unknown.		STRING
-tag3	Tag sequence at 3'-end. Use option -predict if unknown.		STRING
-mm5	Maximum number of allowed mismatches at the 5'-end.	0	INT
-mm3	Maximum number of allowed mismatches at the 3'-end. *	0	INT
-trim_within	The sequence of the tag could occur not only at the sequence end, but also at any other position of the sequence. To assure that only tags are trimmed, the tag sequences can be defined to occur only at the ends allowing a certain number of variable bases.	1.5x tag length	INT

Option/flag	Description	Default	Range
	(cont.) The default value for -trim_within for a tag sequence of at least 10 bp is 1.5 times the tag sequence length. Example: Use -tag3 NNNNNNNNNCCAAACACACCCAACACACCAC -trim_within 60 to trim ATCCATTTCCCAAACACACCCAACACACCAC AAAAAAAAAAAAAAAACAAACAACACC		
-cont	Trim tag sequences continuously. This is helpful if you have sequence with tag sequence repeats or sequence that are concatenated tag sequences. Note that a high number of allowed mismatches and continuous trimming can cause over-trimming. Use more than 20% mismatches with continuous trimming only with caution.		
-split	This feature removes tag contaminations inside the sequences and splits fragment-to-fragment concatenations into separate sequences. The optional integer value specifies the maximum number of allowed mismatches for the internal (concatenated) tag sequence(s). This feature should be used with caution for inputs with only a 5' or 3' tag sequence (likely splits too many false positive that naturally occur for single tags compared to much longer concatenated 5' and 3' tags). The number of mismatches is used as maximum value for -stats. This option will cause a decrease in speed. sequence1-tag3-tag5-sequence2	0	INT
-split5r	This features is similar to -split, but instead of search for 3'-5'-tag repeats, it will search for 5'-tag repeats (2 or more). This option only applies if both -tag3 and -tag5 are specified. To split at a single 5'-tag, run the program without -tag3 and with -split. sequence1-tag5-tag5-sequence2	0	INT

Option/flag	Description	Default	Range
-split3r	This feature is similar to -split, but instead of search for 3'-5'-tag repeats, it will search for 3'-3'-tag repeats (2 or more). This option only applies if both -tag3 and -tag5 are specified. To split at a single 3'-tag, run the program without -tag5 and with -split. sequence1-tag3-tag3-sequence2	0	INT
-splitall	This feature is for convenience only and applies the integer value to all split options (-split, -split3r and -split5r). This option only applies if both -tag3 and -tag5 are specified and will overwrite all other split options with the given integer value. Combining all split options can split things like sequence1-tag3-tag3-tag5-tag5-tag5-sequence2 into sequence1 and sequence2.	0	INT

***** The independent definition for the 5'- and 3'-end of the reads accounts for the differences in tag sequences due to the limitations of the sequencing method used to generate the datasets. The 3'-end will in most cases show a lower number of matching tag sequences with low number of mismatches due to incomplete or missing tags at the ends of incompletely sequenced fragments.

Upload data to the PRINSEQ web version

To upload a new dataset in FASTA and QUAL format (or FASTQ format) to PRINSEQ, follow these steps:

1. Go to http://prinseq.sourceforge.net
2. Click on "Use PRINSEQ" in the top menu on the right (the latest PRINSEQ web version should load)
3. Click on "Upload new data"
4. Select your FASTA and QUAL files or your FASTQ file and click "Submit"

Notes

After clicking the submit button, a status bar (not progress bar) will be displayed until the file upload is completed. During the data processing, several progress bars will show the progress of the data parsing and statistics calculation steps. After the data is parsed and processed successfully, the user interface will provide a menu on the left and information and guidance in the main panel.

Possible problems

1. The PRINSEQ web interface does not load / is not visible.

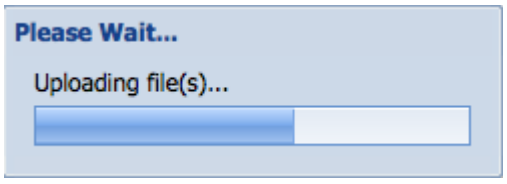
   You only see this and nothing else happens:

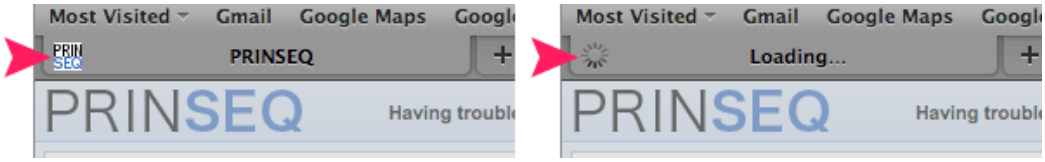
   Solution: Make sure that you have JavaScript activated in your browser, as this is required to load and use PRINSEQ's web interface.

2. The upload status bar does not disappear.

   After clicking on the submit button you see this and it does not disappear:

   Solution: The first thing to check is if the file is still uploading. The easiest way to do this is by checking the loading icon in your browser.



If you see the loading icon (right) instead of the PRINSEQ icon (left), your file is still uploading and you should give it more time. If you see the PRINSEQ icon instead of the loading icon, your file did not upload completely and this caused an error. If you have a slow connection to the Internet or try to upload large files, the connection to the web server can time out before the upload was completed. If you did not upload compressed files, try to compress your files with any of the supported compression algorithms (ZIP, GZIP, ...).

In rare cases, the issue can also be caused by certain Firefox plugins or extensions. If possible, use an alternative browser to test if this was the case. If the browser caused the problem, updating Firefox and the plugins/extension to the latest version might solve the problem.

Manage options

The DeconSeq web version allows users to manage the filter, trim and reformat options in different ways. There are features to save or reset the options currently set, to load previously saved options or to select one of the option pre-sets. Loaded options or selected pre-sets will only be applied after clicking on the "Set new options" button at the bottom. This allows users to review the options before actually setting them.

The feature to save options allows users to process different datasets with the same parameters and to record preprocessing parameters. For parameters that are likely to be used repeatedly in the future, users can request to add those parameters to the list of pre-set options.

Filter options

Length related

Sequences in the SFF files can be as short as 40 bp (shorter sequences are filtered during signal processing). For multiplexed samples, the MID trimmed sequences can be as short at 28 bp (assuming a 12 bp MID tag). Such short sequences can cause problems during, for example, database searches to find similar sequences. Short sequences are more likely to match at a random position by chance than longer sequences and may therefore result in false positive functional or taxonomical assignments. Furthermore, short sequences are likely to be quality trimmed during the signal-processing step and of lower quality with possible sequencing errors.

In some cases, sequences can be much longer than several standard deviations above the mean length (e.g. 1,500+ bp for a 500 bp mean length with a 100 bp standard deviation). Those sequences should be used with caution as they likely contain long stretches of homopolymer runs as in the following example. Homopolymers are a known issue of pyrosequencing technologies such as 454/Roche [1].

A rule of thumb for sequence length thresholds of longer-read datasets is to filter sequences shorter than 60 bp (20 amino acids) and longer than twice the mean length.

Quality score related

In addition to the decrease in quality across the read, regions with homopolymer stretches will tend to have lower quality scores. Huse *et al.* [1] found that sequences with an average score below 25 had more errors than those with higher averages.

Low quality sequences can cause problems during downstream analysis. Most assemblers or aligners do not take into account quality scores when processing the data. The errors in the reads can complicate the assembly process and might cause misassemblies or make an assembly impossible.

Most published thresholds for the sequence mean quality score range from 15 to 25.

GC content related

The GC content distribution of most samples should follow a normal distribution. In some cases, a bi-modal distribution can be observed, especially for metagenomic datasets. This filter is rarely used, but proved useful to separate sequences in a bi-modal distribution.

Ambiguity code related

Sequences can contain the ambiguous base N for positions that could not be identified as a particular base. A high number of Ns can be a sign for a low quality sequence or even dataset. If no quality scores are available, the sequence quality can be inferred from the percent of Ns found in a sequence or dataset. Huse *et al.* [1] found that the presence of any ambiguous base calls was a sign for overall poor sequence quality.

Ambiguous bases can cause problems during downstream analysis. Assemblers such as Velvet and aligners such as SHAHA2 or BWA use a 2-bit encoding system to represent nucleotides, as it offers a space efficient way to store sequences. For example, the nucleotides A, C, G and T might be 2-bit encoded as 00, 01, 10 and 11.

The 2-bit encoding, however, only allows to store the four nucleotides and any additional ambiguous base cannot be represented. The different programs deal with the problem in different ways. Some programs replace ambiguous bases with a random base (e.g. BWA [2]) and others with a fixed base (e.g. SHAHA2 and Velvet replace Ns with As [3]). This can result in misassemblies or false mapping of sequences to a reference sequence and therefore, sequences with a high number of Ns should be removed before downstream analysis.

Filtering out all reads containing Ns is only suggested if the loss can be afforded (e.g. high coverage datasets or low number of sequences with ambiguous bases). Filtering reads containing more than 1% of ambiguous bases is advised.

Data content related

To *select a subset* of all sequence in a dataset, the number of wanted sequences can be specified. The first X sequences passing all other specified filters can be selected this way.

The *sequence duplicates* can be defined using different methods. *Exact duplicates* are identical sequence copies, whereas *5' or 3' duplicates* are sequences that are identical with the 5' or 3' end of a longer sequence. Considering the double-stranded nature of DNA, duplicates could also be considered sequences that are identical with the *reverse complement* of another sequence.

Depending on the dataset and downstream analysis, it should be considered to filter sequence duplicates. The main purpose of removing duplicates is to mitigate the effects of PCR amplification bias introduced during library construction. In addition, removing duplicates can result in computational benefits by reducing the number of sequences that need to be processed and by lowering the memory requirements. Sequence duplicates can also impact abundance or expression measures and can result in false variant (SNP) calling.

PRINSEQ filters duplicates without allowing mismatches, as artificial duplicates tend to have the same errors and error-models are computationally more expensive. Programs such as cdhit-454 [4] use clustering techniques to identify near identical duplicates. However, those methods tend to miss duplicates identified by PRINSEQ due to the used clustering methods. For best results, duplicates should initially be filtered using PRINSEQ and then optionally using clustering methods.

For metagenomic datasets, the exact and 5' duplicates should be removed. The 3' and reverse complement duplicates can be removed as they do not provide additional information in database searches, but might be useful for variant discovery or assembly.

Sequence complexity related

Low-complexity sequences are defined as having commonly found stretches of nucleotides with limited information content (e.g. the dinucleotide repeat CACACACACA). Such sequences can produce a large number of high-scoring but biologically insignificant results in database searches. PRINSEQ calculates the sequence complexity using the DUST and Entropy approaches as they present two commonly used examples.

The *DUST* approach is adapted from the algorithm used to mask low-complexity regions during BLAST search preprocessing [5]. The scores are computed based on how often different trinucleotides occur and are scaled from 0 to 100. Higher scores imply lower complexity. A sequence of homopolymer repeats (e.g. TTTTTTTTT) has a score of 100, of dinucleotide repeats (e.g. TATATATATA) has a score around 49, and of trinucleotide repeats (e.g. TAGTAGTAGTAG) has a score around 32.

The *Entropy* approach evaluates the entropy of trinucleotides in a sequence. The entropy values are scaled from 0 to 100 and lower entropy values imply lower complexity. A sequence of homopolymer repeats (e.g. TTTTTTTTT) has an entropy value of 0, of dinucleotide repeats (e.g. TATATATATA) has a value around 16, and of trinucleotide repeats (e.g. TAGTAGTAGTAG) has a value around 26.

Sequences with a DUST score above 7 or an entropy value below 70 can be considered low-complexity. An entropy value of 50 or 60 would be a more conservative choice.

Custom filter parameters

The custom filter allows the specification of user defined filter using a two value system. Each new filter should be defined on a separate line and values should be separated by space. The first value defines the filter pattern (any combination of the letters "ACGTN"). The second value defines the number of repeats or percentage of occurrence of the filter pattern. The percentage values are defined by a number followed by the %-sign (without space). If no %-sign is given, it is assumed that the value specifies the number of repeats of the filter pattern.

Examples:

AAT 8	filters out sequences containing AATAATAATAATAATAATAATAAT anywhere in the sequence
T 70%	filters out sequences with more than 70% T's in the sequence
A 15	filters out sequences containing AAAAAAAAAAAAAAA anywhere in the sequence

Trim options

Trim by length/position

Sequences can be trimmed to a specific length or a fixed number of nucleotides can be trimmed from either end.

Trim tails

Poly-A/T tails can be trimmed from either end specifying a minimum tail length. All repeats of As or Ts with at least this length will be trimmed from the sequence ends. A small number of tails can occur even after trimming poly-A/T tails. For example, a sequence that ends with AAAAATTTTT and that has been trimmed for the poly-T will still contain the poly-A.

Trimming poly-A/T tails can reduce the number of false positives during database searches, as long tails tend to align well to sequences with low complexity or sequences with poly-A tails in the database.

Trim ends by quality scores

As for Sanger sequencing, next-generation sequencers produce data with linearly degrading quality across the read. The quality scores for 454/Roche sequencers are PHRED-based since version 1.1.03, ranging from 0 to 40. Phred values are log-scaled, where a quality score of 10 represents a 1 in 10 chance of an incorrect base call and a quality score of 20 represents a 1 in 100 chance of an incorrect base call.

Sequences can be trimmed from either end using different rules applied to a sliding window. To stop at the first base that fails the rule defined, use a window size of 1. A bigger window size can trim sequences that might contain a high quality score in between low quality scores without stopping at the high quality score. To move the sliding window over all quality scores without missing any, the step size should be less or equal to the window size.

Note: The quality trimming during the signal processing step (see "Raw data processing" document) may not be sufficient. Trimmed sequences can end with low quality bases or even with ambiguous base N (approx. 1%). Reads with RLMIDs (Rapid library multiplex identifiers) may be trimmed in high quality regions as the default behavior will cause the reads to be trimmed at the first position the MID sequence matches, even if it is not the MID but a natural occurring match inside the read.

The parameters should be set to trim positions with a quality score below 20.

Reformat options

Reformat sequences

The sequence *characters case* can be changed between lower and upper case. This can be useful to, for example, remove soft-masking from sequences.

Sequences can be converted between RNA and DNA. This can be useful to, for example, convert RNA sequences into DNA before generating a BLAST nucleotide database.

Reformat header lines

The header line of a sequence in FASTA or FASTQ format contains the ">" symbol followed by the sequence identifier (sequence name) and an optional description separated by space. In many cases, the description in the sequence header is not used for any downstream analysis. Removing the description in the *sequence header* can significantly reduce the size of the FASTA or FASTQ file. The sequence header can contain information about the sequence that can be incorrect after data preprocessing (e.g. trimming) and therefore should be corrected or removed.

The *sequence identifiers* can be renamed to contain new information or to ensure that all sequences have a unique identifier, which is required by most programs used for downstream analysis. Sequence identifiers should not contain spaces, >, or quotes (which will be automatically removed by PRINSEQ). A counter is added to each identifier to assure its uniqueness. For example, "mySeq_10" will generate the IDs (in FASTA format):

>mySeq_10**1**
ACGTACGTACGT
>mySeq_10**2**
ACGTACGTACGT
>mySeq_10**3**
...

Standalone version options

The standalone version does not provide any graphical outputs and is designed for preprocessing purposes only. The only currently available standalone version is called "lite" as it does not require any non-core Perl modules for processing (no installation of additional Perl modules or third party programs is required). The readme file contains information on the usage of the standalone version.

The following table contains all the options available for the standalone version.

Option/flag	Description	Default	Range
`-help` or `-h`	Print the help message; ignore other arguments		
`-man`	Print the full documentation; ignore other arguments		
`-version`	Print program version; ignore other arguments		
`-verbose`	Prints status and info messages during processing		
Input options			
`-fastq`	Input file in FASTQ format that contains the sequence and quality data		`FILE`
`-fasta`	Input file in FASTA format that contains the sequence data		`FILE`
`-qual`	Input file in QUAL format that contains the quality data		`FILE`
`-si13`	Quality data in FASTQ file is in Solexa/Illumina 1.3+ format and should be scaled to Phred quality scores ranging from 0 to 40. (Not required for Solexa/Illumina 1.5+, Sanger, Roche/454, Ion Torrent, PacBio data.)		
Output options			
`-out_format`	Output format 1 (FASTA only), 2 (FASTA and QUAL), 3 (FASTQ)	same as input	`[1,2,3]`
`-out_good`	Change the output filename and location. The file extension will be added automatically. Output files are by default created in the same directory as the input file with an additional "_prinseq_good_XXXX" in their name (where XXXX is replaced by random characters to prevent overwriting previous files). Use "-out_good null" to prevent the program from generating the output file for data passing all filters. Example: use "file_passed" to generate the output file file_passed.fasta in the current directory		`STRING, null`

Option/flag	Description	Default	Range
-out_bad	Same as –out_good but for data not passing any filter		STRING, null
-log	Log file to keep track of parameters, errors, etc. The log file name is optional. If no file name is given, the log file name will be "inputname.log". If the log file already exists, new content will be added to the file.	Input name.log	STRING
Filter options			
-min_len	Filter sequence shorter than min_len		INT
-max_len	Filter sequence longer than max_len		INT
-range_len	Filter sequence by length range. Multiple range values should be separated by comma without spaces. Example: -range_len 50-100,250-300		STRING
-min_gc	Filter sequence with GC content below min_gc		INT [0..100]
-max_gc	Filter sequence with GC content above max_gc		INT [0..100]
-range_gc	Filter sequence by GC content range. Multiple range values should be separated by comma without spaces. Example: -range_gc 50-60,75-90		STRING
-min_qual_score	Filter sequence with at least one quality score below min_qual_score		INT
-max_qual_score	Filter sequence with at least one quality score above max_qual_score		INT
-min_qual_mean	Filter sequence with quality score mean below min_qual_mean		INT
-max_qual_mean	Filter sequence with quality score mean above max_qual_mean		INT
-ns_max_p	Filter sequence with more than ns_max_p percentage of Ns		INT [0..100]
-ns_max_n	Filter sequence with more than ns_max_n Ns		INT
-noniupac	Filter sequence with characters other than A, C, G, T or N		

Option/flag	Description	Default	Range
-seq_num	Only keep the first seq_num number of sequences (that pass all other filters)		INT
-derep	Type of duplicates to filter. Allowed values are 1, 2, 3, 4 and 5. Use integers for multiple selections (e.g. 124 to use type 1, 2 and 4). The order does not matter. Option 2 and 3 will set 1 and option 5 will set 4 as these are subsets of the other option. 1 (exact duplicate), 2 (5' duplicate), 3 (3' duplicate), 4 (reverse complement exact duplicate), 5 (reverse complement 5'/3' duplicate)		INT
-lc_method	Method to filter low complexity sequences		[dust, entropy]
-lc_threshold	The threshold value used to filter sequences by sequence complexity. The dust method uses this as maximum allowed score and the entropy method as minimum allowed value.		INT [0..100]
-lc_threshold	The threshold value used to filter sequences by sequence complexity. The dust method uses this as maximum allowed score and the entropy method as minimum allowed value.		INT [0..100]
Trim options			
-trim_to_len	Trim all sequence from the 3'-end to result in sequence with this length		INT
-trim_left	Trim sequence at the 5'-end by trim_left positions		INT
-trim_right	Trim sequence at the 3'-end by trim_right positions		INT
-trim_tail_left	Trim poly-A/T tail with a minimum length of trim_tail_left at the 5'-end		INT
-trim_tail_right	Trim poly-A/T tail with a minimum length of trim_tail_right at the 3'-end		INT
-trim_qual_left	Trim sequence by quality score from the 5'-end with this threshold score		INT
-trim_qual_right	Trim sequence by quality score from the 3'-end with this threshold score		INT

Option/flag	Description	Default	Range
-trim_qual_type	Type of quality score calculation to use	min	[min,mean, max,sum]
-trim_qual_rule	Rule to use to compare quality score to calculated value. Allowed options are lt (less than), gt (greater than) and et (equal to)	lt	[lt,gt,et]
-trim_qual_window	The sliding window size used to calculate quality score by type. To stop at the first base that fails the rule defined, use a window size of 1.	1	INT
-trim_qual_step	Step size used to move the sliding window. To move the window over all quality scores without missing any, the step size should be less or equal to the window size.	1	INT
Reformat options			
-seq_case	Changes sequence character case to upper or lower case		[upper, lower]
-dna_rna	Convert sequence between DNA and RNA. Allowed options are "dna" (convert from RNA to DNA) and "rna" (convert from DNA to RNA).		[dna,rna]
-line_width	Sequence characters per line. Use 0 if you want each sequence in a single line. Use 80 for line breaks every 80 characters. Note that this option only applies to FASTA output files, since FASTQ files store sequences without additional line breaks.	60	INT
-rm_header	Remove the sequence header. This includes everything after the sequence identifier (which is kept unchanged)		
-seq_id	Rename the sequence identifier. A counter is added to each identifier to assure its uniqueness		STRING
Summary statistic options *			

Option/flag	Description	Default	Range
-stats_info	Outputs basic information such as number of reads (reads) and total bases (bases).		
-stats_len	Outputs minimum (min), maximum (max), range (range), mean (mean), standard deviation (stddev), mode (mode) and mode value (modeval), and median (median) for read length.		
-stats_dinuc	Outputs the dinucleotide odds ratio for AA/TT (aatt), AC/GT (acgt), AG/CT (agct), AT (at), CA/TG (catg), CC/GG (ccgg), CG (cg), GA/TC (gatc), GC (gc) and TA (ta).		
-stats_tag	Outputs the probability of a tag sequence at the 5'-end (prob5) and 3'-end (prob3) in percentage (0..100).		
-stats_dupl	Outputs the number of exact duplicates (exact), 5' duplicates (5), 3' duplicates (3), exact duplicates with reverse complements (exactrevcom) and 5'/3' duplicates with reverse complements (revcomp), and total number of duplicates (total). The maximum number of duplicates is given under the value name with an additional "maxd" (e.g. exactmaxd or 5maxd).		
-stats_ns	Outputs the number of reads with ambiguous base N (seqswithn), the maximum number of Ns per read (maxn) and the maximum percentage of Ns per read (maxp). The maxn and maxp value are not necessary from the same sequence.		
-stats_all	Outputs all available summary statistics.		

* The summary statistic values are written to STDOUT in the form: "parameter_name statistic_name value" (without the quotes). For example, "stats_info reads 10000" or "stats_len max 500". Only one statistic is written per line and values are separated by tabs.

Upload data to the DeconSeq web version

To upload a new dataset in FASTA or FASTQ format to DeconSeq, follow these steps:

1. Go to http://deconseq.sourceforge.net
2. Click on "Use DeconSeq" in the top menu on the right (the latest DeconSeq web version should load)
3. Select your FASTA or FASTQ file
4. Select the retain and remove (optional) database(s)
5. Click "Submit"

Notes

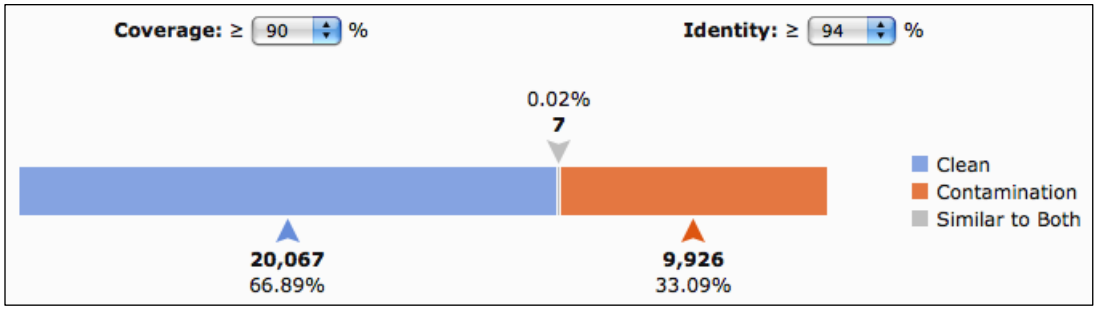
Step 4 allows selecting two types of reference databases. The remove databases are the databases used to screen for contaminants. The retain databases are used to eliminate redundant hits with higher similarity to non-contaminant genomes (e.g. viral sequences in the human genome for viral metagenomes).

Sequence contaminant removal

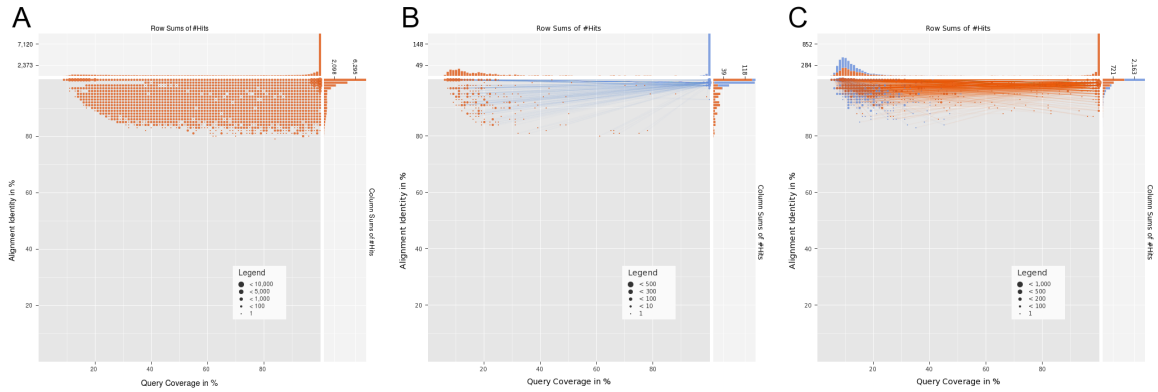
Sequences obtained from impure nucleic acid preparations may contain DNA from sources other than the sample. Those sequence contaminations are a serious concern to the quality of the data used for downstream analysis, possibly causing erroneous conclusions.

The dinucleotide abundance approach used by PRINSEQ to identify if a dataset contains contamination does not allow the identification and removal of single contaminant sequences. Sequence similarity seems to be the only reliable option to identify single contaminant sequences. However, BLAST searches against the human reference genome are slow and lack corresponding regions (gaps, variants, …). Furthermore, novel sequences were found in every new human genome sequenced [6]. DeconSeq allows the automated identification and removal of sequence contamination in longer-read datasets (>150 bp mean read length) using an algorithm tens of times faster than BLAST [7].

DeconSeq uses the query sequence coverage and alignment identity to identify sequences that are similar to a contaminant sequence in the remove databases. The identity is a measure for how similar the query sequence to the reference sequence is and the coverage is a measure of how much of the query sequence is similar to the reference sequence. If a retain database is selected, query sequences are classified as "Similar to Both" if they are similar to sequences in the remove and retain database using the coverage and identity thresholds. All other query sequences can be classified as "Clean" or "Contamination", as shown below.



DeconSeq generate Coverage vs. Identity plots to guide users in their selection of threshold values. The plots show the number of matching reads for different query coverage and alignment identity values. The number of matching reads with a specific coverage and identity value defines the size of each dot in the plots. Red dots represent matching reads against the remove databases and blue dots against retain databases. The column and row sums at the top and right of each plot allow an easier identification of the number of sequences that match for a particular threshold value.



The plots for matching reads against the remove databases do not show matching reads that additionally have a match against the retain databases (A). Results for reads matching against both databases are shown in a second plot where dots for a single read are connected by lines. If the match against the remove database is more similar, then the line is colored red, otherwise blue. In B, for example, the majority of sequences is more similar to the retain databases and in C the majority is more similar to the remove databases.

Standalone version options

The standalone version does not provide any graphical outputs and databases. The databases used for contamination screening have to be generated as described in the readme file distributed with the source code. The readme file also contains information on the usage of the standalone version.

Option/flag	Description	Default	Range
`-help` or `-h`	Print the help message; ignore other arguments		
`-man`	Print the full documentation; ignore other arguments		
`-version`	Print program version; ignore other arguments		
Input/Output options			
`-show_dbs`	Prints a list of available databases		
`-f`	Input file in FASTA or FASTQ format that contains the query sequences		`FILE`
`-dbs`	Name of remove database(s) to use. Names are according to their definition in the config file. Separate multiple database names by comma without spaces. Example: `-dbs hs1,hs2,hsref`	`hsref`	`STRING`
`-dbs_retain`	Name of database(s) to use for cross-check. Query sequences with hit against any `-dbs` will be compared to these databases. Databases have to be different from names in `-dbs`. Names are according to their definition in the config file. Separate multiple database names by comma without spaces. Example: `-dbs_retain bact,vir`		`STRING`
`-out_dir`	Directory where the results should be written. If the directory does not exist, it will be created.	Current directory	`STRING`
`-group`	If dbs_retain is set, then this option can be used to group the sequences similar to `-dbs` and `-dbs_retain` databases with either the clean or the contamination output file. If `-group` is not set and `-dbs_retain` is set, then three separate files will be generated. Use 1 for grouping "Clean + Both" and 2 for grouping "Contamination + Both".		`[1,2]`

Option/flag	Description	Default	Range
-no_seq_out	Prevents the generation of the FASTA/FASTQ output file for the given coverage and identity thresholds. This feature is e.g. useful for the web-version since -i and -c are set interactively and not yet defined at the data processing step.		
-keep_tmp_files	Prevents from unlinking the generated tmp files. These usually include the id file and the .tsv file(s). This feature is e.g. useful for the web-version since .tsv files are used to dynamically generate the output files.		
-id	Optional parameter. If not set, ID will be automatically generated to prevent from overwriting previous results. This option is useful if integrated into other tools and the output filenames need to be known.		STRING
Alignment options			
-i	Alignment identity threshold in percentage. The identity is calculated for the part of the query sequence that is aligned to a reference sequence. For example, a query sequence of 100 bp that aligns to a reference sequence over the first 50 bp with 40 matching positions has an identity value of 80%.		INT [1..100]
-c	Alignment coverage threshold in percent. The coverage is calculated for the part of the query sequence that is aligned to a reference sequence. For example, a query sequence of 100 bp that aligns to a reference sequence over the first 50 bp with 40 matching positions has an coverage value of 50%.		INT [1..100]
-S	Chunk size of reads in bp as used by BWA-SW	10000000	INT
-z	Z-best value as used by BWA-SW	1	INT
-T	Alignment score threshold as used by BWA-SW	30	INT

References

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

2. Li H, Durbin R: **Fast and accurate short read alignment with Burrows-Wheeler transform**. *Bioinformatics* 2009, **25**:1754-1760.

3. Li H, Homer N: **A survey of sequence alignment algorithms for next-generation sequencing**. *Brief Bioinform* 2010.

4. Niu B, Fu L, Sun S, Li W: **Artificial and natural duplicates in pyrosequencing reads of metagenomic data**. *BMC Bioinformatics* 2010, **11**:187.

5. Morgulis A, Gertz EM, Schäffer AA, Agarwala R: **A fast and symmetric DUST implementation to mask low-complexity DNA sequences**. *J. Comput. Biol* 2006, **13**:1028-1040.

6. Li R, Li Y, Zheng H, Luo R, Zhu H, Li Q, Qian W, Ren Y, Tian G, Li J, Zhou G, Zhu X, Wu H, Qin J, Jin X, Li D, Cao H, Hu X, Blanche H, Cann H, Zhang X, Li S, Bolund L, Kristiansen K, Yang H, Wang J, Wang J: **Building the sequence map of the human pan-genome**. *Nat. Biotechnol* 2010, **28**:57-63.

7. Li H, Durbin R: **Fast and accurate long-read alignment with Burrows-Wheeler transform**. *Bioinformatics* 2010, **26**:589-595.