Trimming DNA Reads

Arvin Chireh[1,*]

1 Institutionen för Klinisk Neurovetenskap, Karolinska Institutet, Stockholm, Sweden

* arvin.chireh@gmail.com

Abstract

In modern DNA-sequencing, the quality of the base calls frequently drops towards the end of the sequencing reads. Therefore, there is a need for programs that can recognize this technical quality decline, and trim the ends of reads accordingly. The purpose of this project was to develop and test a new program for trimming of DNA sequences.

A program with two alternative trimming algorithms (tr1 and tr2) was written in Python. A fastq file produced with MISEQ sequencer, containing 250 000 reads, was used for evaluating the program. Mean sequence length and base call quality was assessed before and after the trimming, with corresponding histograms.

The tr1 algorithm trimmed 104 960 of 250 000 reads (42 %), reducing the mean sequence length from 169 to 149, increasing mean phred score from 29.4 to 29.9. The tr2 algorithm trimmed 34877 of 250 000 reads (14%), reducing mean sequence length from 169 to 157, improving mean phred score from 29.4 to 30. Only descriptive statistics was used.

In conclusion, the trimming program developed and tested here gives a marginal increase in mean base call quality, at the cost of reduced sequencing length. The algorithms were only tested on one dataset. Further evaluation is needed to properly benchmark the algorithms. Most likely, one should look for other, more efficient and more selective algorithms.

Introduction

Recent advances in DNA sequencing has made it possible to sequence DNA at an unprecedented pace. As a result, DNA sequencing has emerged as a valuable tool in biological research as well as clinical decision making. However, the current techniques have some limitations. For instance, base call quality frequently drops towards the end of each individual read from the sequencer. Therefore, there is a need for software than can recognize this quality decline towards the end of reads, and remove (trim) low quality bases. Quality, in this context, is described in terms of a phred score [1] , ranging between 0 and 40. The purpose of this work was to develop and evaluate a new method of trimming low quality bases without unnecessary removal of good quality bases.

Materials and methods

Test data

A fastq file generated from the MISEQ instrument (llumina, Inc), containing 250 000 DNA sequencing reads, was used as test data for assessing the effect of the algorithm.

An empty file, a fasta file and a fastq file with short sequences (15 base pair sequences) was used to check for the ability of the program to cope with suboptimal data.

Program

The program was written as an executable python script. Both algorithms, presented below, were written in the same script, giving the user the opportunity to choose algorithm when calling the script.

Several controls were built into the program. One control was written to check for empty files or wrong file types. Another control was inserted to skip trimming of too short sequences. No control for checking nucelotide validity (i.e. allowed base) was implemented, as it would slow down the program, and was not considered relevant for the purpose.

The "tr1" algorithm was designed to calculate the mean phred score and its standard deviation (SD) of the first 30 bases in each record, to use as an indicator. For each record in the fastq file, the program iterates through each base (base[i]) and checks if the mean phred score from i and onwards (seq[i:]) is below the quality threshold that was set (1 SD or more below the mean phred score of the first 30 bases). If the condition is met, the sequence is trimmed at that point and stored in a list of trimmed records. Sequences that are too short (less than 30 bases), or fail to fulfill criteria for trimming, are stored in the list of trimmed records as is. The program reports number of trimmed, not tirmmed (denoted as "untrimmed") and too short sequences.

The "tr2" algorithm is based on a similar approach. However, instead of checking the mean phred score from base[i] and onwards, the program looks at 10 consecutive bases from i and onwards (i.e. base[i], base[i+1] ... base[i+9]). If all these consecutive bases are below the quality threshold, the sequence is cut at base[i].

The program writes the new records to a fastq file.

Assessment

Mean sequencing length and mean quality (phred score), together with equivalent metrics for the last 20 bases in each record (i.e. the tails of the sequences) were assessed before and after trimming. Histograms were plotted to check the change of sequencing length and quality distributions.

Results

The tr1 algorithm trimmed 104 960 of 250 000 reads (42 %), reducing the mean sequence length from 169 to 149, increasing mean phred score from 29.4 to 29.9, Table 1, Figure 1 . The tr2 algorithm trimmed 34877 of 250 000 reads (14%), reducing mean sequence length from 169 to 157, improving mean phred score from 29.4 to 30, Table 2, Figure 2.

Table 1. Statistics for tr1 algorithm Tail defined as the last 20 bases.

	Seq length, mean	phred score, mean	tail phred score, mean
Before trimming	169	29.4	28.6
After trimming	149	29.9	30.4

Discussion

In this work, I attempted to develop a software to trim low quality reads at the end of sequencing reads. The developed program contains two algorithms, however neither of

Table 2. Statistics for tr2 algorithm Tail defined as the last 20 bases.

	Seq length, mean	phred score, mean	tail phred score, mean
Before trimming	169	29.4	28.6
After trimming	157	30	29.1

these seem to do major improvements in overall quality. However, the expected quality improvement is highly dependent on the sequencing data. In our test data, the mean phred quality of the last 20 bases (i.e. tail) was not very different from the overall base call quality (29.4 mean for whole sequences, 28.6 mean for tails). As a result, one should not expect high changes in overall quality following a trimming of the tail. In fact, both algorithms succeeded in making the mean quality of the tail close to the overall mean, or even better, particularly tr1. As a disadvantage of the program design, all reported quality statistics are means of all bases pooled together. There was no pairwise comparison for each sequence.

More worrisome, only 42 % and 14 % of the reads were trimmed by tr1 and tr2, respectively. In other words, the algorithm failed to recognize and trim low quality bases in the tails of a majority of the sequences. This may or may not be dependent on the data set.

Notes on management of the project

This project was also an exercise with the purpose of improving the ability to handle a bioinformatics project. As a result, I read and implemented some of the ideas introduced by William Stafford Noble [2]. For instance, I used a modified version of the proposed directory structure, with logical and chronological sub-directories. Moreover, I kept a digital lab notebook in markdown format, with procedural notes coupled with timestamps. I also used git version control system to keep versions of my software, and looked at previous version at some points during the development process.

Noble suggest keeping a file with all command-lines that were used, and keep this file in parallel to the logbook. I personally find it easier to just keep those commands in the logbook as well, together with comments and other text. I find driver scripts too complicated at this point, and maybe not necessary for the scale of projects at this level.

I also added control code in my program to detect some primitive errors, as suggested by Noble and the assignment instructions. Obviously, these controls could have been more carefully designed, and cover a larger range of potential problems.

Conclusion

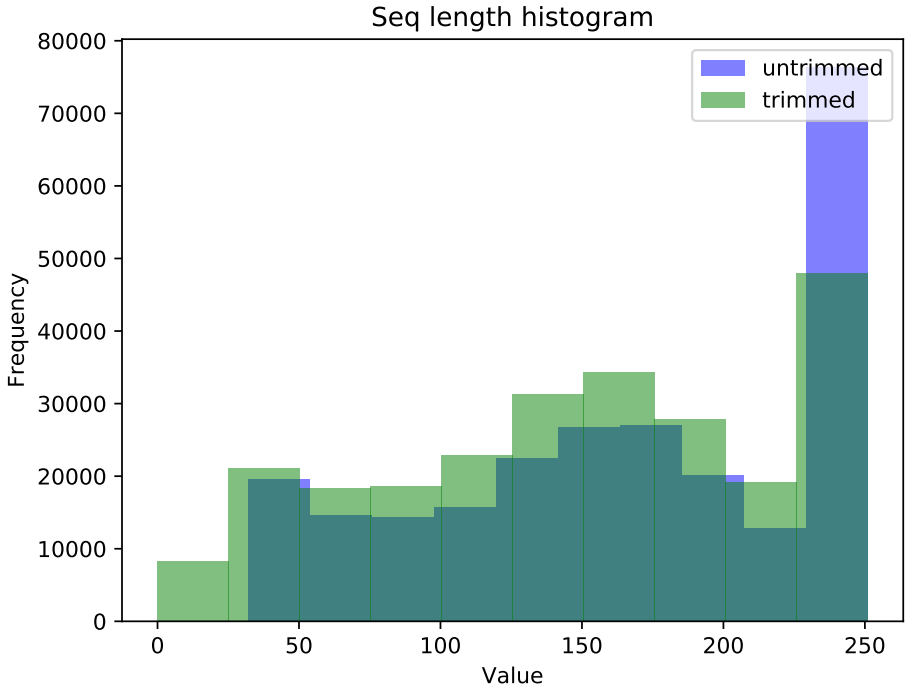
In conclusion, the algorithms used in my program offers a modest quality improvement at the cost of sequence data. I do recommend a search for more effective and selective algorithms. Moreover, the algortihms has to be tested on a further range of data sets before defintive conclusions can be made, perhaps with pairwise testing and/or other statistical testing.

This project was a great opportunity to improve management of a bioinformatic project. The ideas of William Stafford Noble were considered and to a large degree implemented in this work.

References

1. Ewing B, Green P. Base-calling of automated sequencer traces using phred. II. Error probabilities. . Genome Res. 1998 Mar;8(3):186-94.

Fig 1. Tr1 Algorithm Histograms showing distributions of sequencing length and quality, for algorithm tr1. Untrimmed = Before trimming. Trimmed = After trimming.



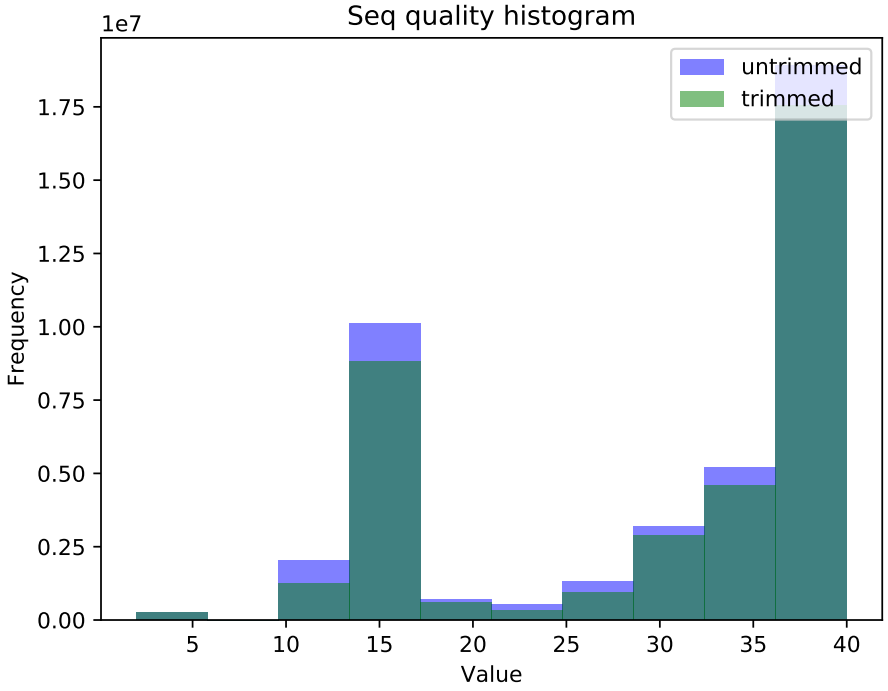
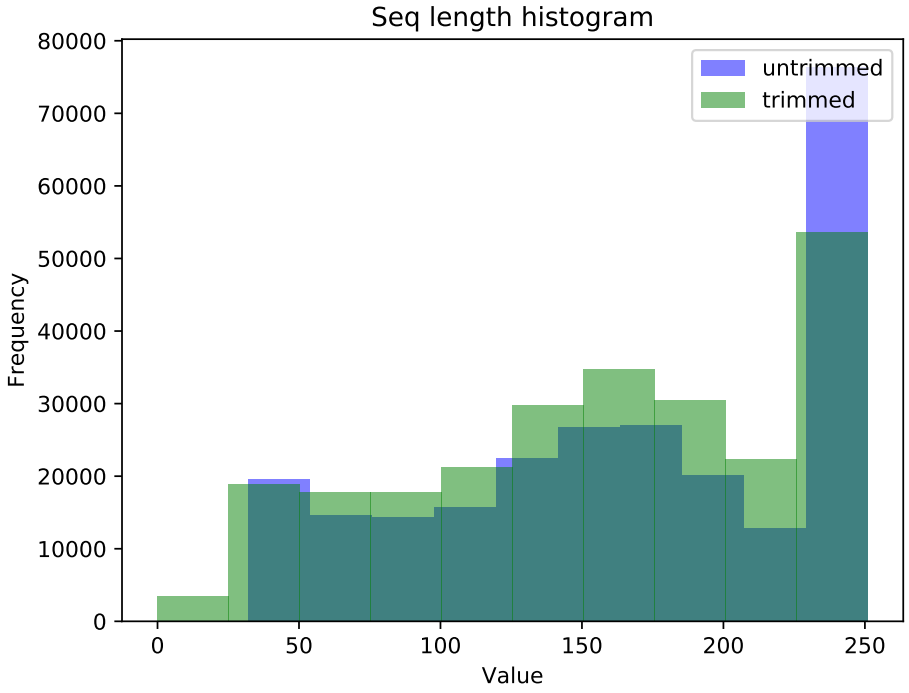
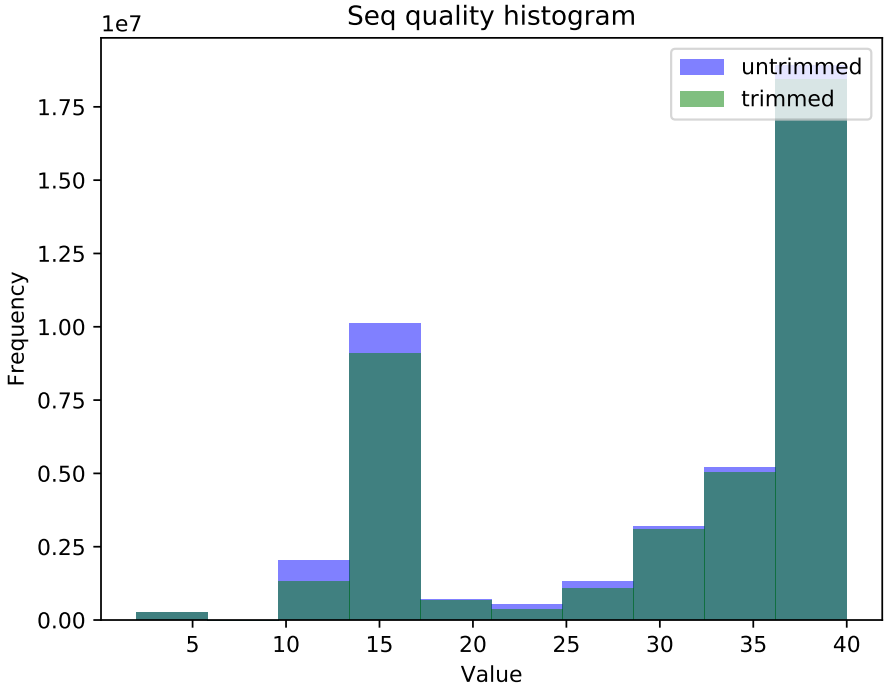


Fig 2. Tr2 Algorithm Histograms showing distributions of sequencing length and quality, for algorithm tr2. Untrimmed = Before trimming. Trimmed = After trimming.





2. William Stafford Noble. A Quick Guide to Organizing Computational Biology Projects. PLoS Comput Biol. 2009 Jul; 5(7): e1000424.