Bioinformatics, 2019, 0–0
doi: 10.1093/bioinformatics/xxxxx
Advance Access Publication Date: 15 May 2019
Applications Note

Sequence Analysis

A read mapping tool for high-throughput CRISPR library screening

Yige Zhao[1,*]

[1]Department of Systems Biology, Columbia University, New York, NY, USA.

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on May 15, 2019; revised on May 15, 2019; rejected on May 15, 2019

Abstract

Motivation: The fast development of gene editing tools leads to various genetic screening protocols which based on lentiviral library construction and high-throughput sequencing. Downstream analysis of such experiments always involves in mapping a subsequence within a read to a previously designed library, in order to compare whether the amount of an sgRNA follows expectation. Here I designed a tool suitable for mapping different types of CRISPR library.

Results: This tool estimates alternative sequences within read based on constant backbone sequences. It can be applied for single-end or paired-end sequencing read files, and also for one or multiple alternative sequences within reads. It uses hash table for locating target positions, which is efficient for large amounts of data.

Availability: Source code is available at https://github.com/YZhao96/lib_count. These scripts were implemented in Python 3.7, and runs as a command-line code under Linux or Unix.

Contact: yz3419@columbia.edu

Supplementary information: Sample data and other materials are available at Courseworks and https://github.com/YZhao96/lib_count.

1 Introduction

Regularly interspaced, short palindromic repeats (CRISPR)-associated nucleases, such as Cas9, are more and more frequently used in whole genome genetic screening, because of its convenience and high-efficiency (Peng *et al.*, 2015). Typically, guide RNA sequences are designed near protospacer adjacent motif (PAM) inside target genes, then corresponding oligos are synthesized and cloned into lentiviral vectors. A lentiviral library usually carries more than 10^5 types of sequences (Sanjana *et al.*, 2014) (GeCKOv2 human library, for example, contains 123,411 sgRNA sequences targeting 19,050 genes), which are later delivered into cells and stably integrated into genomes. These guide RNAs are constantly expressed in the cell line and lead to change of expression level of target genes. After some treatment, guide sequences targeting certain genes are enriched or depleted in the cell library, which helps with understanding of gene functions. (Figure S1)

Reads retrieved by polymerase chain reaction (PCR) are always highly similar with constant backbone sequences, except some alternative guide sequence parts. Efficiently mapping these alternative sequences to the designed library is of vital importance. There are several strategies regarding this purpose. First, constructing reference for the whole PCR fragments while using sequence alignment tools, such as Bowtie (DeJesus *et al.*, 2016; Langmead, 2010), is very wasteful, because the backbone parts are unimportant not required for mapping. Some studies use software such as cutadapt (Chen *et al.*, 2015; Martin, 2011), while doing semi-global alignment to truncate reads into the alternative sequences theoretically has $O(mn)$ time complexity, where m and n are read length and backbone sequence length. Besides, it's not very good to use if one read contains multiple alternative parts, which is common nowadays for paired guide RNA library, etc. The last strategy is extremely straight forward and efficient, and it's implemented within some software such as MAGeCK (Li *et al.*, 2014). It simply tries every possible position of a read for every possible length from the library, and directly get index from the hash table preprocessed from designed library. It takes at most $O(kl)$, where $k<m$ is the number of possible starting positions and l is number of possible lengths of library, while alternative sequence from different reads always start from adjacent positions ($k<10$) and guide sequence lengths only range from 19-23 nt (thus $l<6$). However, when an alternative part is short, for example, a 6 nt barcode with all possible nucleotide combinations, a single try of position often maps the sequence to an incorrect index.

I adapted the third strategy to locate alternative sequences based on the constant sequences rather than on themselves, which can be more widely applied in different library construction methods.

2 Methods

A complete procedure of mapping is divided into 4 parts. (Figure 1A) First, the constant backbone sequences are processed into a list hash table, with keys to be sequences of specified seed length and values to be its distance to the corresponding alternative part. Each library is also processed into a hash table, overall size of library and all possible lengths of the library sequences are recorded. Second, a subset of reads is used to deduce the approximate start position for the whole read file. In this step, seeds of backbone are searched thoroughly on the whole read. Then for each read in the read file, seeds of backbone are again searched in the read to locate the alternative sequence, but this time, only previously deduced range of start position are tried. Finally, using the positions of start / end pair, alternative parts are cut from the sequence and add counts to the same library sequence.

In the second and third step, for a specific read, each subsequence with length of the seed is used to get the distance from the backbone hash. If succeeds, alternative part starting position within this read can be calculated by the distance and current read position. Iterate this process over and over again until more than a certain number of seed-length subsequence support the same start position, and use that position as result. Positions of each alternative part is calculated in pairs, first from left to right to find the 3' end, then from this 3' end backwards, from minimum library length away, to find the 5' end. (Figure 1B)

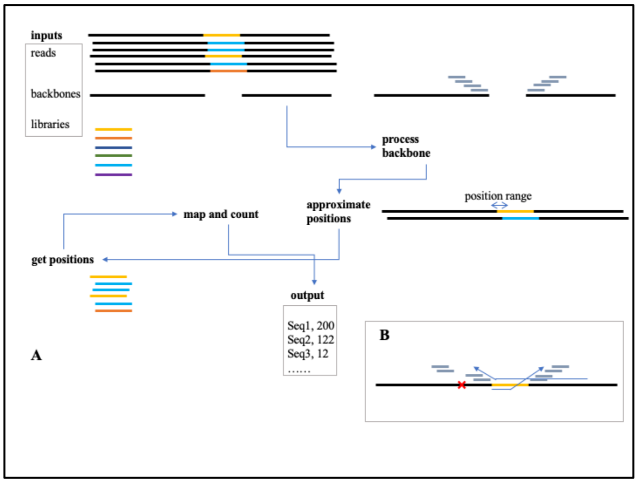

Fig. 1. **Methods Overview.** A. Sequence library mapping procedures. B. Locating an alternative part within a read.

3 Results

The previously algorithms are together implemented into a single Python 3.7 script which can be used by command line input to allow easy use. It takes fastq/fasta file, backbone file and library csv file as inputs, and outputs a csv file indicating counts of each library sequence.

3.1 Multiple Applications for different CRISPR library construction strategies

By specifying parameters, this tool can be used these purposes: (Figure S2)

- Single-end sequencing, with one or more alternative parts, strand-specific or not.
- Paired-end sequencing, with one or more alternative parts, located in same or different end.
- For multiple alternative parts, whether or not these different libraries are grouped (sequence 1 in library 1 can only be paired with sequence 1 in library 2) or randomly combined.

Such flexible settings suit common sgRNA or pgRNA libraries, as well as libraries studying gene interaction, which may have randomly combined libraries. The software was tested on sgRNA library (SRR7975589) where 20 nt guide sequence is followed by a constant sequence of ~19 nt and then a second barcode sequence of 6 nt length. (Zhu *et al.*, 2019)

3.2 Parameters settings and performance

The seed length and minimum match number need to be specified case by case. In general, if we want the it to be more error-tolerant, we can set seeds to be short; if we want the position to be more accurate, we can set minimum match number to be big. The programs work extremely well when there's no mutations to achieved 100% accuracy for all set of parameters. I simulated different data files based on GeCKOv2 (Sanjana *et al.*, 2014) with different number of indels and single nucleotide change on backbone, each containing 10^5 sequences and measured how much can they be mapped. (Supplementary Table S1, Figure S3) The backbone constant sequence length is 14-16 nt. Unmatchable results come from two parts, either such mutation is not tolerant, or a wrong position is picked. Insertions or deletions are generally less tolerant than single nucleotide variance. On average, the algorithm takes about 13s on personal computer to do mapping for 10^6 sequences. Performance also varies when different library and backbone sequences. It has a time complexity of around *O(ku)*, where *u* denotes the minimum matching number.

3.3 Compare with cutadapt

Theoretically, this tool outperforms cutadapt (Martin, 2011) on speed, which uses semi-global alignment. In real test for simulated sequences without mutations, cutadapt-2.3 uses around 1.4 s for trimming 10^5 sequences on the same machine, which is similar to this tool. However, cutadapt sometimes only trim either the 5' or 3' end, and may be required to run for a few rounds in order to get the cleanest data. Further, it cannot always get the right position. The rate of match for sequences without mutations is 98% for cutadapt, while this software has a rate of 1.

Acknowledgements

I thank all the classmates and TAs in Computational Genomics Spring 2019 of Columbia University for discussions. Especially I thank Dr. Itsik Pe'er for his patience and enthusiasm during lectures, as well as his useful suggestions regarding this project.

Funding. This work was supported by the Department of Computer Science, Columbia University, for Google Cloud virtual environment availability.

Conflicts of Interest: none declared.

References

Chen,S. *et al.* (2015) Genome-wide CRISPR screen in a mouse model of tumor growth and metastasis. *Cell*, **160**, 1246–1260.

DeJesus,R. *et al.* (2016) Functional CRISPR screening identifies the ufmylation pathway as a regulator of SQSTM1/p62. *Elife*, **5**, e17290.

Langmead,B. (2010) Aligning short sequencing reads with Bowtie. *Curr. Protoc. Bioinforma.*, **32**, 11.7. 1-11.7. 14.

Li,W. *et al.* (2014) MAGeCK enables robust identification of essential genes from genome-scale CRISPR/Cas9 knockout screens. *Genome Biol.*, **15**, 554.

Martin,M. (2011) Cutadapt removes adapter sequences from high-throughput sequencing reads. *EMBnet J.*, **17**, 10–12.

Peng,J. *et al.* (2015) High-throughput screens in mammalian cells using the CRISPR-Cas9 system. *FEBS J.*, **282**, 2089–2096.

Sanjana,N.E. *et al.* (2014) Improved vectors and genome-wide libraries for CRISPR screening. *Nat. Methods*, **11**, 783–784.

Zhu,S. *et al.* (2019) Guide RNAs with embedded barcodes boost CRISPR-pooled screens. *Genome Biol.*, **20**, 20.