BIOINFORMATICS APPLICATIONS NOTE Vol. 29 no. 4 2013, pages 494–496
doi:10.1093/bioinformatics/bts731

Genome analysis

Advance Access publication January 12, 2013

NGSUtils: a software suite for analyzing and manipulating next-generation sequencing datasets

Marcus R. Breese[1,2,3] and Yunlong Liu[1,2,3,*]

[1]Center for Computational Biology and Bioinformatics, [2]Center for Medical Genomics and [3]Department of Medical and Molecular Genetics, Indiana University School of Medicine, Indianapolis, IN 46202, USA

Associate Editor: Inanc Birol

ABSTRACT

Summary: NGSUtils is a suite of software tools for manipulating data common to next-generation sequencing experiments, such as FASTQ, BED and BAM format files. These tools provide a stable and modular platform for data management and analysis.

Availability and implementation: NGSUtils is available under a BSD license and works on Mac OS X and Linux systems. Python 2.6+ and virtualenv are required. More information and source code may be obtained from the website: http://ngsutils.org.

Contact: yunliu@iupui.edu

Supplemental information: Supplementary data are available at *Bioinformatics* online.

Received on July 17, 2012; revised on December 1, 2012; accepted on December 27, 2012

1 INTRODUCTION

With the broad implementation of next-generation sequencing (NGS) technology in biomedical research, genomics sequencing data are generated at an unprecedented rate. Such advances bring enormous challenges in data analysis, of which efficient, standardized and consistent analysis are critical. With the deluge of data brought about by these technological advances, the time required to analyze NGS datasets has become the limiting factor in high-throughput experiments (Richter and Sexton, 2009). In most cases, NGS data processing involves manipulating, converting and processing large standardized files, such as FASTQ (Cock *et al.*, 2010), BED (Kent *et al.*, 2002) and BAM (Li *et al.*, 2009) files. Frequently, initial pre-processing is required before using these files as input to mapping for assembly purposes. Once reads have been mapped, post-processing of these files is typically applied to filter out noisy mappings or calculate summary information.

The analysis of NGS experiments frequently consists of writing custom one-off scripts to manipulate data or calculate final results. This approach is inefficient and can lead to reproducibility errors. There are some existing tools that aid in the analysis of NGS data, such as BEDTools (Quinlan and Hall, 2010), VCFtools (Danecek *et al.*, 2011), samtools (Li *et al.*, 2009) and the FASTX toolkit (http://hannonlab.cshl.edu/fastx_toolkit/). To help augment the functionality of exiting tools and allow for pipeline assembly, the NGSUtils suite was developed, covering different aspects of NGS data analysis, including pre-processing, post-processing, filtering, format conversion and final result calculations.

2 IMPLEMENTATION

NGSUtils is organized into a series of modules based on the primary file format that is being manipulated. Each of the modules contains multiple programs, mainly written in Python. BAM files are written and read using the pysam library (http://code.google.com/p/pysam) and must be indexed using samtools. FASTQ, BED and GTF input files may be plaintext or gzip compressed. Both base- and color-space reads are supported. A summary of the major commands from each module is given below. A complete list of common commands is listed in Supplementary Table S1.

2.1 bamutils

This module is focused on manipulating BAM files. Four of the major commands are *basecall*, *expressed*, *filter* and *counts*.

The *basecall* command performs a similar function to the samtools *pileup* command, in that it calculates base calls for each position in the reference genome. Instead of the more complicated VCF format, the output is a human readable tab-delimited format. Consensus and minor base calls, the number of calls for each base, composition of insertions and strand bias are also calculated.

The *expressed* command is useful for finding regions of novel expression in a model-free manner. This operates by scanning the reference genome for regions with overlapping reads. This can be used for discovery of novel genes or non-coding RNAs.

The *filter* command is used for removing reads from a BAM file based upon a set of user-defined criteria. After reads have been mapped to a reference, it can be useful to post-process the resulting BAM file to remove noise from downstream analysis. Some of the supported criteria are number of allowed mismatches between a read and the reference (optionally counting indels of any length as 1), inclusion/exclusion of particular regions, a white or blacklist of reads and filtering based on any of the included extra 'tags'. Some of the common tags to filter by are AS (alignment score), IH (number of alignments in the file) and NM (edit distance). Mismatch filtering can also be

*To whom correspondence should be addressed.

performed in SNP aware manner, potentially limiting the effects of mapping bias in post-processing. The *stats* command can be a useful tool to help explore a dataset to see how reads might be affected by different filtering criteria.

Finally, the *counts* command is used to calculate the read counts for genes, BED regions, repeatmasker (Smit *et al.*, 1996–2010) defined repetitive elements or genome bins of a user-defined size. If given a GTF gene model, read counts for each exon or alternatively spliced fragment can be individually tallied, which may help the analysis of alternative splicing. Counts may be normalized to RPKM values (Mortazavi *et al.*, 2008) using a number of methods.

2.2 fastqutils

The FASTQ format is a common output format from NGSs, consisting of read sequences and the quality of individual *base-calls*. These then serve as the input for mapping programs to translate raw reads into mapped positions. For high-quality mapping results, it is important to pre-process the input FASTQ files. The *fastqutils* module contains a variety of commands for converting to and from many different input formats, as well as performing initial quality control pre-processing. The *fastqutils split* command is useful for efficient parallel processing, and it splits a FASTQ file into sub-files in a paired-end aware manner. The *fastqutils trim* command will scan each read and remove user-defined 5′-and 3′-linker sequences using a Smith–Waterman (SW) (Smith and Waterman, 1981) alignment algorithm. For quality-control purposes, the *fastqutils filter* command can remove or truncate reads based on a number of user-defined criteria. Supported criteria include the number of wildcard calls (Ns), read length, trimming bases from the 3′-end of a read based on read quality, truncating a read when the average basecall quality of a sliding window falls below a threshold and removing adaptor sequences (non-SW aligned). Additionally, for paired-end datasets, there is a filter for removing reads that after previous filtering may be missing their partner read. Each of the filtering criteria is processed in a chain; therefore, the results of one filter feed directly into the next, allowing for efficient filtering with multiple criteria in one step.

2.3 bedutils

BED files contain coordinates for regions of a genome. In the context of NGS experiments, these can correspond to binding sites, as found in ChIP-seq or CLIP-seq, or regions targeted for enrichment, such as in exome enrichment. There are two primary commands for BED files: *reduce* and *refcount*. The *reduce* command merges overlapping BED regions into one long contiguous region. The *refcount* command takes one 'reference' BED file and one or more 'input' BED files. For each region in the reference BED file, the input files are scanned for overlapping regions. These are then tallied, and a count for each input file is produced. One example where this might be useful is a ChIP-seq experiment, where the data are a BED file for each sample containing 100 bp regions that were detected. Using the *reduce* command, all of the samples could be merged together into one reference set of found regions. This could then be used as the reference BED file. Next, using each of the original sample BED files as inputs, *refcount* could be used to find regions that were enriched in one group of samples versus the other.

2.4 gtfutils

A common format for genome annotation is the GTF format (http://mblab.wustl.edu/GTF22.html). GTF annotations can be easily obtained for most organisms from many sources, including the UCSC Genome Browser (Kent *et al.*, 2002). In the *bamutils* module, GTF files are used for read counting and mapping statistics. The *gtfutils* module includes commands for reading and enhancing GTF gene model annotations. The *gtfutils junctions* command takes a GTF annotation file and a genome FASTA file and output all possible splice-junctions for all genes. Potential splices from all exons within a gene are outputted, including those from multiple isoforms, if isoform annotations have been added to the GTF file. One use for this potential splice-junction file is mapping junction-spanning RNA-seq reads. To facilitate RNA-seq read mapping, each sequence in the junction library is named in such a way that files mapped to the junction library can be readily converted back to genomic reference coordinates, including gaps in alignments, if necessary. This conversion is handled by the *bamutils convertregion* command.

3 CONCLUSIONS

Individually, each of these programs performs a small job in the analysis of NGS experiments. However, collectively, this suite of tools acts as building blocks that can be combined into larger pipelines for data exploration and analysis. With modules that operate from FASTQ pre-processing through BAM post-processing and RPKM calculations, NGSUtils compliments existing tools and provides unique functionality that helps each step of an NGS data analysis pipeline.

ACKNOWLEDGEMENT

The authors thank Howard J. Edenberg for valuable discussions about the application of these tools.

Funding: National Institutes of Health (AA017941, CA113001, GM088076); Indiana Biobank, whose operations are subsidized by the Indiana Clinical and Translational Sciences Institute, which is supported by an NCRR Clinical and Translational Sciences Award (in part) (U54-RR025761. Anantha Shekhar, PI) and an NCRR construction grant (C06-RR020128-01. R.S. Fife, PI, K. Cornetta, Co-I).

Conflict of Interest: none declared.

REFERENCES

Cock,P.J.A. *et al.* (2010) The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. *Nucleic Acids Res.*, **38**, 1767–1771.

Danecek,P. *et al.* (2011) The variant call format and VCFtools. *Bioinformatics*, **27**, 2156–2158.

Kent,W.J. *et al.* (2002) The human genome browser at UCSC. *Genome Res.*, **12**, 996–1006.

Li,H. *et al.* (2009) The sequence alignment/map format and SAMtools. *Bioinformatics*, **25**, 2078–2079.
Mortazavi,A. *et al.* (2008) Mapping and quantifying mammalian transcriptomes by RNA-Seq. *Nat. Methods*, **5**, 621–628.
Quinlan,A.R. and Hall,I.M. (2010) BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics*, **26**, 841–842.
Richter,B.G. and Sexton,D.P. (2009) Managing and analyzing next-generation sequence data. *PLoS Comput. Biol.*, **5**, e1000369.
Smit,A. *et al. RepeatMasker Open-3.0*. 1996–2010 <http://www.repeatmasker.org>.
Smith,T.F. and Waterman,M.S. (1981) Identification of common molecular subsequences. *J. Mol. Biol.*, **147**, 195–197.