Sequence analysis

Jasmine: a Java pipeline for isomiR characterization in miRNA-Seq data

Xiangfu Zhong 1,2, Albert Pla1,2 and Simon Rayner1,2,3,*

1Department of Medical Genetics, University of Oslo, Oslo, 2Department of Medical Genetics, Institute of Clinical Medicine, Oslo University Hospital, Oslo and 3Hybrid Technology Hub – Centre of Excellence, Institute of Basic Medical Sciences, University of Oslo, Oslo, Norway

*To whom correspondence should be addressed.

Received on March 7, 2019; revised on September 13, 2019; editorial decision on October 26, 2019; accepted on October 30, 2019

Abstract

Motivation: The existence of complex subpopulations of miRNA isoforms, or isomiRs, is well established. While many tools exist for investigating isomiR populations, they differ in how they characterize an isomiR, making it difficult to compare results across different tools. Thus, there is a need for a more comprehensive and systematic standard for defining isomiRs. Such a standard would allow investigation of isomiR population structure in progressively more refined sub-populations, permitting the identification of more subtle changes between conditions and leading to an improved understanding of the processes that generate these differences.

Results: We developed Jasmine, a software tool that incorporates a hierarchal framework for characterizing isomiR populations. Jasmine is a Java application that can process raw read data in fastq/fasta format, or mapped reads in SAM format to produce a detailed characterization of isomiR populations. Thus, Jasmine can reveal structure not apparent in a standard miRNA-Seq analysis pipeline.

Availability and implementation: Jasmine is implemented in Java and R and freely available at bitbucket https://bitbucket.org/bipous/jasmine/src/master/.

Contact: simon.rayner@medisin.uio.no

Supplementary information: Supplementary data are available at Bioinformatics online.

1 Introduction

MicroRNAs (miRNA) are a class of small non-coding RNAs, which have significant function in gene regulation (Bartel, 2018). Nowadays, high-throughput sequencing (HTS) is the most common platform for investigating changes in miRNA populations between conditions (in the form of read data). However, HTS has revealed the presence of ‘miRNA-like’ reads, which differ from known miRNAs by one or more nucleotides. These miRNA-like isoforms (or isomiRs), were first proposed by Morin et al. (2008) and their functional significance is well established (see Nielsen et al., 2012). Many software tools have been developed for isomiRs profiling such as isomiR-SEA (Urgese et al., 2016) and Chimira (Vitsios and Enright, 2015)—a complete list of available tools is listed in Supplementary Table S2—and a recent study compared the performance of seven popular isomiR identification tools (Amstel et al., 2017). However, most tools only report isomiR reads, rather than classifying them in terms of the type of modification. Consequently, there is no easy way to profile the population either within, or across datasets, and it is not possible to relate studies that have used different analysis tools as many of them have their own embedded qualification criteria.

A further consideration is that miRBase (Kozomara et al., 2018), the standard miRNA reference database, catalogs miRNAs as well defined entities with a distinct start and stop positions producing a single sequence. Also, many entries differ from other entries by a single nucleotide. Finally, the authors caution that they provide ‘minimal gate-keeping’ for annotation and entries may be removed in subsequent releases. All of these points can confound analysis of miRNA datasets and need to be considered when characterizing isomiRs in NGS datasets. Thus, to identify miRNAs or isomiRs, a curated miRNA reference annotation and well-defined standard is needed for systematic isomiR annotation.

Here, we propose a comprehensive isomiR nomenclature and implement it in Jasmine, a Java based pipeline that can incorporate curated miRBase annotation (Zhong et al., 2019) and perform isomiR analysis on miRNA-Seq data.

2 Usage notes

Jasmine accepts unmapped (fastq/fasta format) or mapped (SAM format) reads as input. Mapping: (i) As a starting point it is assumed
that all short miRNA-like reads are potential products of miRNA biogenesis. (ii) Jasmine uses curated miRBase entries as mapping references, rather than genome or original miRBase entries, to reduce multiple and ambiguous mappings, so it can be applied to any species annotated in miBase. (iii) Reads are collapsed into unique sequences and pre-miRNAs are used as mapping reference. Thus, Jasmine can run on PC with limited CPUs and memory. *IsomiR analysis*: Jasmine incorporates a hierarchal isomiR nomenclature (Fig. 1), allowing progressively more refined characterization of isomiR populations in HTS studies.

3 Materials and methods

Jasmine is written in Java, implementing commonly used software tools to aid pre-analysis and processing (e.g. bowtie for mapping and Trimomatic for adapter trimming—see Supplementary Table S1 for details). Jasmine is split into four analysis modules for flexible and rapid analysis of multiple samples or projects (see work flow in Supplementary Fig. S1).

3.1 Input

Jasmine accepts three types of files as input: (i) raw read data in fastq/fasta format; (ii) adapter trimmed reads in fastq or fasta; (iii) collapsed reads in fasta format. Additionally, a configuration file in XML format is also required to launch the pipeline.

3.2 Analysis modules

The four analysis modules in Jasmine are: pre-processing, mapping, parsing and post-analysis.

1. Pre-processing: two primary sub steps are executed: (i) Adapter trimming via Trimomatic (Bolger et al., 2014) or cutadapt (Martin, 2011) to implement adapter trimming. (ii) Collapsing trimmed reads into unique reads using fastx_collaper and fastq_to_fasta from the FASTX-Toolkit. FastQC and MultiQC (Ewels et al., 2016) provide QC reports on trimmed reads. A summary report is generated by Jasmine for the step.

2. Mapping: The default built-in mapping approach uses bowtie (Langmead et al., 2009) to map reads against curated miRNA precursor sequences but users may define other tools for the mapping step if required. Since Jasmine focuses on isomiR analysis, at least one mismatch is recommended.

3. IsomiR parsing: The core module of Jasmine, it parses mapped reads (in SAM format) to identify isomiRs, assign them a proper isomiR name, and generate a detailed report. Reads are classified into two groups; (i) reads mapping to a unique location; (ii) reads mapping to multiple mapped locations within the provided mapping reference.

4. Post analysis: Provides a summary report using our isomiR nomenclature, including features such as polymorphism and templated extension and count tables for differential expression analysis.

3.3 Output

The output files comprise a miRNA count table, an isomiR count table, a polymorphism table, a templated extension count profile report and an isomiR report file. The latter provides details on isomiR reads, pre-miRNA reference, modifications and assigned isomiR name.

The count tables generated by Jasmine are suffixed with ‘Level’ according to following classification:

Level 1: (i) mature miRNAs; (ii) all different isoforms grouped together as a single isomiR group ‘isomiR’.

Level 2: (i) mature miRNAs; (ii) isomiRs further sub-classed by lengths: same length-, longer- and shorter- than reference miRNA.

Level 3: mature miRNAs and isomiRs sub-classed according to the 30 groups shown in Figure 1a.

Level 4: lowest level of isomiR description. Each isomiR is reported separately with all the information needed to reconstruct the sequence from the reference miRBase entry. For polymorphisms, whether the change occurs in the seed region and whether it corresponds to known single-nucleotide variants (SNVs) is also reported.

4 Using the Jasmine pipeline

4.1 Using Jasmine

The Jasmine pipeline is available as compiled jar file or compiling from source. Both are available via Bitbucket. An example XML...
configuration file and user manual are also provided. Jasmine can be run from the terminal using the command:

> java -jar Jasmine.jar jasmine_config.xml

where jasmine_config.xml is the configuration file for analysis.

4.2 Benefits and drawbacks of using Jasmine

The Jasmine pipeline implements the isomiR nomenclature described in Figure 1. The nomenclature in Jasmine is more comprehensive and flexible for describing isomiR populations in sequencing datasets, compared to other tools, see Table 1, Figure 1b and Supplementary Table S2. Users can select which level is suitable for their specific study.

Jasmine addresses multiple mapping issues by: (i) mapping reads against miRNA precursor sequences; (ii) merging miRNA annotation entries with identical sequences and (iii) calculating sequence similarity before isomiR profiling.

A potential drawback of mapping to precursor sequences is that reads from other genomic regions which are not annotated as miRNAs will not be mapped. However, this can be addressed by excluding miRNAs which can map to other regions. For example, miRNA sequences that map to tRNAs. Also, Jasmine seeks to software tools, miRNA annotation set and level of isomiR classification. This gives the user a high degree of flexibility in defining how to classify reads as isomiRs. Jasmine can be configured according to software tools, miRNA annotation set and level of isomiR classification. This gives the user a high degree of flexibility in an analysis, while retaining the ability to compare results across different studies.

Acknowledgements

Part of this work was presented at the NGS conference in Barcelona, Spain, 03–05 April 2017.

Funding

This work was supported by the Helse Sør-Øst grant 2016122 and Norwegian Research Council grant 274715.

Conflict of Interest: none declared.

References


