Split-seq read processing protocol

Version 1.0.0 (3/14/2019)

Rebekka Wegmann

Snijder lab, ETH Zurich

www.snijderlab.org

Introduction

The following manual describes the usage of a script that processes raw sequencing output from Split-seq experiments into a digital gene expression matrix that will contain integer counts of the number of transcripts per single cell. This pipeline performs multiple analysis steps, including demultiplexing of the raw data by molecular (UMI) and cellular barcode, filtering cellular barcodes by a list of expected barcodes, alignment of reads to a reference genome, collecting basic QC metrics and counting UMIs per cell.

This manual provides an overview of the analysis steps performed by the Split-seq pipeline. Note that this pipeline is based on Drop-seq_tools developed by the McCaroll lab at Harvard Medical school. For a detailed description of how the individual components from the Drop-seq toolbox work, please refer to the Drop-seq_Alignment_Cookbook.

Software and hardware requirements

The Split-seq pipeline requires approximately 50 Gb of RAM (this is the amount needed by STAR to map reads to a human-sized genome). It is written in bash and was tested on Linux, but should run on any UNIX platform.

The following dependencies are provided as part of the toolbox:

- Drop-seq tools v. 2.1.0
- Picard

Additional dependencies:

- STAR v. 2.5 or higher
- Java
- Python 3.6 or higher with pySAM, h5py, itertools, numpy, pandas and matplotlib installed

Overview

Preparing metadata

To get from a raw, unmapped bam to a count matrix, it is necessary to have a set of reference datafiles which provide information about the genome sequence of the organism(s) in your experiment as well as the location of genomic features such as genes, transcripts, exons etc. The Split-seq pipeline uses exactly the same metadata bundle as Drop-seq tools. Briefly, you need to download a reference genome in fasta format and the corresponding annotation in GTF format (e.g. from Ensembl). All additional metadata can be derived from these two files using the create_Drop-seq_reference_metadata script that is provided as part of the Drop-seq tools. Please refer to the Drop-seq_Alignment_Cookbook for details.

Preparing the input file

The Split-seq pipeline expects a queryname-sorted BAM file containing both reads as input. There are various ways to generate such a file from the raw output of an illumine sequencer. For example, one could use Picard IlluminaBasecallsToSam or Illumina's bcl2fastq followed by Picard FastqToSam.

Stage 1: Tagging BAM with molecular and cellular barcodes

In this stage, the molecular and cellular barcodes are extracted from read 2 and added as tags to read 1. For a splitseq library, read 2 has the following structure:

[UMI, 10nt][rd2-BC, 8nt]GTGGCCGATGTTTCGCATCGGCGTACGACT***[rd1-BC,8nt]***ATCCACGTGCTTGAGAGGCCAGAGCATTCG***[RT-BC, 8nt]***

The UMI is a molecular barcode, and the reverse transcription (RT-BC), round 1 ligation (rd1-BC) and round 2 ligation (rd2-BC) together encode a cellular barcode (CBC).

The UMI and the three cellular barcodes are extracted by running Drop-seq tools TagBamWithReadSequenceExtended multiple times. Read pairs where any of the 4 barcode regions contain more than 1 base with a quality score below 10 are discarded.

At the end of this stage, the input bam file is converted to a new bam file that contains only read 1 (the transcript sequence) with the following tags added to each read:

- XM: 10 nucleotide UMI sequence, bases 1-10 f read 2
- XD: 8 nucleotide RT barcode, bases 87-94 of read 2
- XE: 8 nucleotide round 1 ligation barcode, bases 49-56 of read 2
- XF: 8 nucleotide round 2 ligation barcode, bases 11-18 of read 2

Stage 2: Trimming adapters and polyA tails

Here, the Drop-seq tools TrimStartingSequence and polyATrimmer are used to remove template switching oligo sequences from the 5' end of read 1 and polyA tails from the 3' end.

Stage 3: Barcode filtering and correction

At this stage, the extracted cellular barcodes (XD, XE, XF) are compared against a set of expected barcode sequences. By default, this set contains all of the primer sequences we have available in our lab (96 RT primers, 96 round 1 ligation barcodes and 96 round 2 ligation barcodes). You can change this behavior by providing a custom set of barcode lists and setting the –b parameter of the Split-seq pipeline to the folder where you stored your barcode list.

Barcode sequences within 1 edit distance of expected barcodes are assumed to have originated from a sequencing error and are corrected (i.e. set to the expected barcode sequence).

After this correction, any read that still contains one or more unexpected sequences is discarded.

The remaining reads are tagged with the complete CBC (XC), wich is obtained by concatenating XD,XE and XF and written to a new file *unaligned_tagged_BC_filtered.bam.*

In addition to filtering and tagging, this stage also produces a summary plot showing the distribution of reads across the barcoding plates as well as the cumulative fraction of reads per CBC.

Stage 4: Alignment

STAR is used to map reads onto a reference genome. STAR can only use fastq files as input, therefore the unaligned_tagged_BC_filtered.bam file is converted to fastq before the alignment.

Stage 5: Recover molecular and cellular tags and tag bam with gene names and function

In a first step, the bam tags for the UMI and CBC that were lost during the fastq conversion are recovered by sorting the STAR output file by queryname and merging it with *unaligned_tagged_BC_filtered.bam*. During this merge, only the best alignment is kept and all secondary alignments are discarded.

Next, the Drop-seq tools TagReadWithInterval and TagReadWithGeneFunction add additional tags to the aligned bam file. These tags encode information about where the read maps: gn = gene name, gs = gene strand, gf = gene function. For backward compatibility with older versions of Drop-seq tools, these functions also write the gene name to the XG tag and the function to the XF tag.

The final output of this stage, *gene_function_tagged.bam* is an aligned bam file containing tags for the UMI (XM), CBC (XC) and gene information (gn, gs, gf, XG, XF).

Stage 6: Creating digital expression matrix

The Drop-seq tools DigitalExpression function is run twice to calculate digital expression: Once counting only exonic reads, and a second time counting both intronic and exonic reads. We found that for Split-seq, the content of pre-mRNA is higher than for related scRNA-seq protocols, therefore we generally use the count matrix with both intronic and exonic counts for downstream analysis.

Running the script

To run the pipeline with default settings, use the following command:

```
/path/to/splitseq_toolbox/Split-seq_pipeline.sh –g /path/to/STAR_indices –r
/path/to/reference.fasta input.bam
```

Overview of all input options

-g <genomedir>	Directory of STAR genome directory. Required.
-r <referencefasta>	Reference fasta of the Drop-seq reference metadata bundle. Needs to be in the same folder as the other metadata files. Required.
-d <dropseq_root>	Directory containing Drop-seq executables. Default: Subdirectory of the splitseq toolbox.

-o <outputdir>	Where to write output bam. Default: current directory.
-t <tmpdir>	Where to write temporary files. Default: current directory.
-s <STAR_path>	Full path of STAR. Default: STAR is found via PATH environment variable.
-b <barcode_dir>	Full path to directory where the list of expected barcodes is stored. Default: subdirectory of the splitseq toolbox.
-n <num_cells>	Estimated number of cells in the library. Only affects visualization of barcode filtering results. Default: 500.
-p	Reduce file I/O by pipeline commands together. Requires more memory and processing power.
-e	Echo commands instead of executing them. Cannot use with -p.