EMBL predoc course 2019 Day2 Strand-seq plotting pipeline

Hyobin Jeong

October 23, 2019

Contents

1 Introduction

In this practical we will learn how to perform pre-processing of Strand-seq data after the sequencing. We will start from the fastq file to align it to the human reference genome (hg38) and then make a chromosome overview plot based on Snakemake pipeline.

2 Download pipeline for the preprocessing of Strand-seq data

After the sequencing, we will get paired-end Strand-seq fastq files. As a pre-processing, we need to align the reads to the human reference genome (hg38), remove PCR duplicate, count reads for each of the bins in fixed windows. This will require multiple softwares and R script, so that workflow management tool such as Snakemake is useful to proceed convenient and reproducible processing. Snakemake is a python-based workflow management tool and it can run workflows consisting of multiple inter-dependent tasks. These can be computed either locally or on the cluster. In this session, we will download pipeline from the git repository, explore the overall script, and run the script to analyze Strand-seq data from RPE-1 WT.

```
## Copy the Snakemake pipeline (for the practical session)
cd /scratch/name_abcd/
mkdir Practical_WT
cd Practical_WT
cp /scratch/jeong/WGS/practical.tar.gz ./
tar -xvzf practical.tar.gz
cd practical/

##To browse the Snakefile
less Snakefile

##To close the Snakefile
q
```

Now, let's check the config file if we need to change file path or reference genome.

```
##To browse the config file

less Snake.config.json

##This is the config file where we specify the file path of code and reference genome

{
    "qc_plot" : "/g/korbel/meiers/tools/mosaicatcher/mosaicatcher/R/qc.R",
    "sv_plot" : "/g/korbel/meiers/tools/mosaicatcher/mosaicatcher/R/svs.R",
    "mosaic"  : "/g/korbel/meiers/tools/mosaicatcher/mosaicatcher/build/mosaicatcher",
    "genome"  : "/g/korbel/shared/datasets/refgenomes/human/GCA_000001405.15_GRCh38_no_alt_analysis_set.fna",
    "exclude" : "/g/korbel/meiers/tools/mosaicatcher/mosaicatcher/data/exclude/GCA_000001405.15_GRCh38_no_alt
}

##To close the config file
q
```

3 Add sequencing data to the pipeline

After downloading the pipeline, we need to tell the pipeline which data to use by copying/linking files into a fastq subdirectory. The paired-end raw data of RPE-1 WT can be downloaded using the following command.

```
## Copy the raw data of 25 cells from RPE-1 WT
cp /scratch/jeong/WGS/fastq.tar.gz ./
tar -xvzf fastq.tar.gz
ls fastq/

## Or to save the time and disk space, we can make soft link!
## https://www.cyberciti.biz/faq/creating-soft-link-or-symbolic-link/
mkdir fastq
cd fastq
ln -s {source-filename} {symbolic-filename}

## Let's link 25 libraries (two fastq files each)
ln -s /scratch/jeong/WGS/fastq/RPE1WTPE20401_1_sequence.txt.gz RPE1WTPE20401_1_sequence.txt.gz
ln -s /scratch/jeong/WGS/fastq/RPE1WTPE20401_2_sequence.txt.gz RPE1WTPE20401_2_sequence.txt.gz

## Rather than copy and paste 25 times, we can use following script
cp /scratch/jeong/WGS/fastq/make_link.sh ./
```

4 Execute the pipeline

Edit the Snakemake file to specify the sample name and other variables if necessary. Using snakemake -n you can see which jobs are going to be computed. Snakemake script can be launched locally or on the cluster.

```
## 1. Specify the R environment

## Open the Renvironment file
vi ~/.Renviron

## Make INSERT mode of vi editor
shift + I

## Copy and paste following line to the vi editor
R_LIBS=/g/korbel2/StrandSeq/Test_HJ/R_pipeline

##To save the edit and close vi editor
ESC
:wq


## 2. install required packages (cowplot, ggplot2, scales) in your default R environment
R
source("https://bioconductor.org/biocLite.R")
biocLite('cowplot')
```

```
## 3. Run pipeline using EMBL cluster

## Make sh script executable
chmod +x run_pipeline.sh

## Run sh script using EMBL server (you can change e-mail address)
sbatch -t 90:00:00  -N 1 -n 1 --mem=50000 --mail-type=FAIL,BEGIN,END \
--mail-user=hyobin.jeong@embl.de -o output.txt ./run_pipeline.sh
```

5 Explore the output folders and plots

After the function has finished, you will find the four different folders **bam**, **counts**, **plot chromosomes**, and **plot overview** containing all produced files and plots. You can explore the example output files with the following command.

```
##Example of result folders of RPE1 WT data analysis
ls /scratch/name_abcd_v1/Practical_WT/practical/
```

- **bam** This folder contains the bam files and index files after the bwa alignment. During the snakemake pipeline PCR duplicates were marked with SAM Flag but not removed from the bam files. Those bam files will be an input of the downstream analysis such as BreakpointR, StrandPhaseR, or MosaiCatcher.
- **counts** This folder has the count matrix (text files) of Strand-seq reads on the fixed sized bins. The numbers in the file name (20000, 50000, 100000, 200000, 500000) indicates the size (bp) or the window. The count matrix will show the number of Crick and Watson reads separately. Bins have a fixed width, starting from position 0 up to the end of the chromosome. Mapped reads were assigned to bins based on their start position and filtered according to the following criteria for read counting: non-primary and supplementary alignments are excluded; alignments with the QC failure flag are excluded; PCR duplicates are excluded; reads with mapping quality $<= 10$ are excluded. In case of paired-end data only the first read of each pair was used to avoid double-counting.This folder also give info matrix which contains the parameters p and r of the negative binomial (NB) distribution which is important for the SV classification in the downstream analysis. During parameter estimation, bins were excluded from the estimation process if their mean coverage across all cells was very low or if they showed a highly abnormal fraction across cells.
- **plot overview** This folder will show you five pdf files with different sizes of genomic windows. The first page of the pdf file shows the statistics of the analysis result such as the distribution of total number of reads per cell, duplication rate, or excluded bins per chromosomes. Afterwards, every pages show the overview of alignment result of each of the single-cells. The depth of Crick reads are depicted in the green color in the right side, and the depth of Watson reads are depicted in the orange color in the left side of each chromosome lines.
- **plot chromosomes** Under this folder, there will be subfolder with the name of sample you specified in the config file before running the pipeline, and it will contain pdf files per window size per chromosomes.

6 Check the mapped reads from count folder

To select good quality libraries we will check two output files in this section. 1) count info matrix 2) plot overview

For this, we need to open rstudio and set the path.

```
seneca.embl.de
setwd('/scratch/name_abcd_v1/Practical_WT/practical/counts')
```

In the rstudio console, we can run the following script to check the count info table.

```
##Check the count info table with the following command
countinfo <- read.table("testSample.200000.info", sep = '\t', header=T)
head(countinfo)
```

Using cell name and mapped columne, we can check the distribution of mapped read across single cell libraries and exclude the outlier cells which has extremely low mapped reads.

```
##Check the count info table with the following command
countmapped <- as.matrix(countinfo$mapped)
countmappedsort <- as.matrix(sort(countinfo$mapped))
cellnamesort <- countinfo$cell[sort(countmapped, index.return=TRUE)$ix]
rownames(countmappedsort) <- cellnamesort
barplot(t(countmappedsort), las=2, cex.names=0.5,
        cex.axis=0.5, ylab = "number of mapped read")
```

7 Challenge - diagnostic plot of ploidy

For this, we need to open rstudio and set the path.

```
seneca.embl.de
setwd('/scratch/name_abcd_v1/Practical_WT/practical/counts')
```

In the rstudio console, we can run the following script to check the number of w, c reads from count table. For this challenge, we will use the count table from 80 WT cells.

```
##Check the count info table with the following command
count_wc <- read.table("RPE1-WT.200000.txt.gz", sep = '\t', header=T)
head(count_wc)

tmp <- count_wc[count_wc$chrom=="chr11" & count_wc$start==10000000 & count_wc$end==10200000,]
barplot(sort(tmp$c/(tmp$c+tmp$w), decreasing = T), ylab="C/(W+C)")
```

We have practiced to make diagnostic plot for random region of chr11 in RPE1 WT. Can we find the region of Triploid region of chr10q in RPE1 WT? Hint: Triploid region is in chr10q (60.8Mb to chromosome end)

8 Session Info

```
toLatex(sessionInfo())
```

- R version 3.5.1 (2018-07-02), `x86_64-apple-darwin15.6.0`
- Locale: `C`
- Running under: `macOS High Sierra 10.13.3`
- Matrix products: default
- BLAS: `/System/Library/Frameworks/Accelerate.framework/Versions/A/Frameworks/vecLib.framework/Versions/A/libBL`
- LAPACK: `/Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRlapack.dylib`
- Base packages: base, datasets, grDevices, graphics, methods, parallel, stats, stats4, utils
- Other packages: BiocGenerics 0.26.0, GenomeInfoDb 1.16.0, GenomicRanges 1.32.7, IRanges 2.14.12, S4Vectors 0.18.3, breakpointR 1.0.0, breakpointRdata 1.0.0, cowplot 0.9.4, ggplot2 3.1.1, knitr 1.20
- Loaded via a namespace (and not attached): Biobase 2.40.0, BiocParallel 1.14.2, BiocStyle 2.8.2, Biostrings 2.50.2, DT 0.4, DelayedArray 0.6.6, DelayedMatrixStats 1.2.0, FNN 1.1.2.1, GenomeInfoDbData 1.1.0, GenomicAlignments 1.16.0, MOFAtools 0.99.0, Matrix 1.2-14, MultiAssayExperiment 1.6.0, R6 2.4.0, RColorBrewer 1.1-2, RCurl 1.95-4.12, Rcpp 1.0.1, Rhdf5lib 1.2.1, Rsamtools 1.34.0, SingleCellExperiment 1.2.0, SummarizedExperiment 1.10.1, XVector 0.20.0, assertthat 0.2.1, backports 1.1.4, beeswarm 0.2.3, bindr 0.1.1, bindrcpp 0.2.2, bitops 1.0-6, codetools 0.2-15, colorspace 1.4-1, compiler 3.5.1, corrplot 0.84, crayon 1.3.4, data.table 1.12.2, digest 0.6.19, doParallel 1.0.14, dplyr 0.7.6, dynamicTreeCut 1.63-1, edgeR 3.22.3, evaluate 0.11, foreach 1.4.4, ggbeeswarm 0.6.0, ggrepel 0.8.0, glue 1.3.1, grid 3.5.1, gridExtra 2.3, gtable 0.3.0, gtools 3.8.1, highr 0.7, htmltools 0.3.6, htmlwidgets 1.2, httpuv 1.5.1, igraph 1.2.2, iterators 1.0.10, jsonlite 1.6, later 0.8.0, lattice 0.20-35, lazyeval 0.2.2, limma 3.36.3, locfit 1.5-9.1, magrittr 1.5, matrixStats 0.54.0, mime 0.6, munsell 0.5.0, pheatmap 1.0.10, pillar 1.4.0, pkgconfig 2.0.2, plyr 1.8.4, promises 1.0.1, purrr 0.2.5, reshape2 1.4.3, reticulate 1.10, rhdf5 2.24.0, rjson 0.2.20, rlang 0.3.4, rmarkdown 1.10, rprojroot 1.3-2, rstudioapi 0.7, scales 1.0.0, scater 1.8.4, scran 1.8.4, shiny 1.3.2, shinydashboard 0.7.0, statmod 1.4.30, stringi 1.4.3, stringr 1.4.0, tibble 2.1.1, tidyr 0.8.1, tidyselect 0.2.4, tinytex 0.7, tools 3.5.1, tximport 1.8.0, vipor 0.4.5, viridis 0.5.1, viridisLite 0.3.0, withr 2.1.2, xfun 0.3, xtable 1.8-4, yaml 2.2.0, zlibbioc 1.26.0