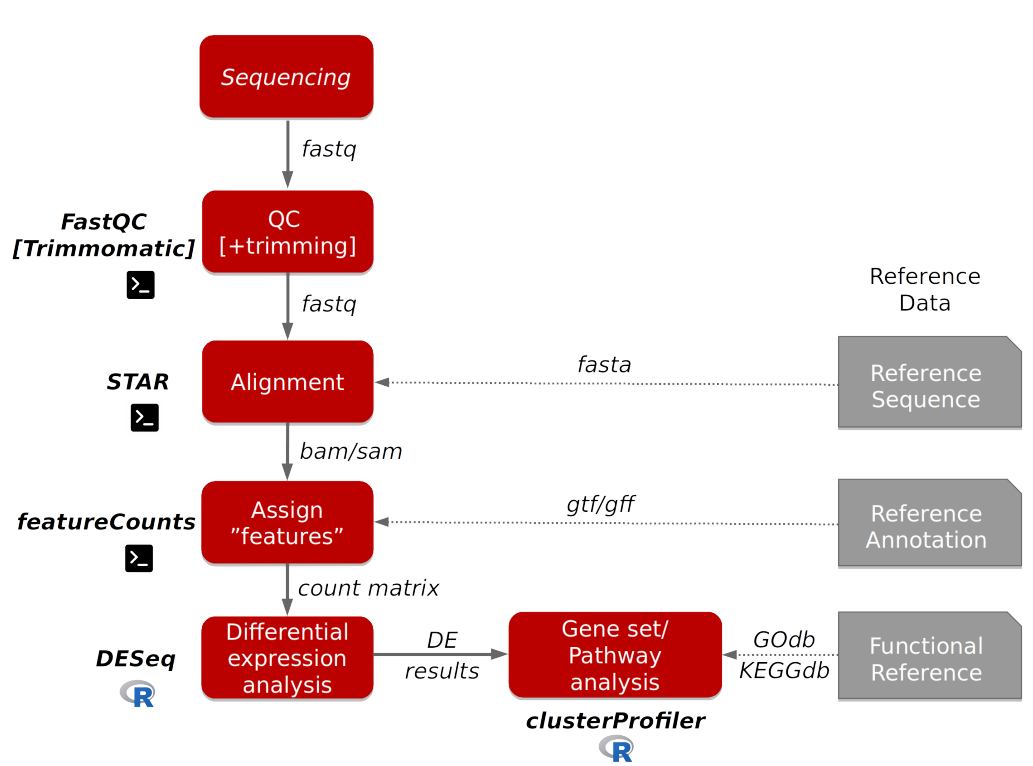
Intro to RNAseq alignment and the STAR aligner

Including indexing a genome with STAR

Jelmer Poelstra, MCIC Wooster

2021/02/17 (updated: 2021-02-25)

Overview of the analysis pipeline



Aligning RNAseq reads

The purpose of aligning reads, also known as *mapping*: **determine where in the genome each read originates from.**

Aligning RNAseq reads is more challenging than DNA reads: due to *splicing*, many reads can't be simply be mapped wholly to the genome.

Moreover, there is often *alternative splicing*, such that a single "correct" reference is simply not applicable.

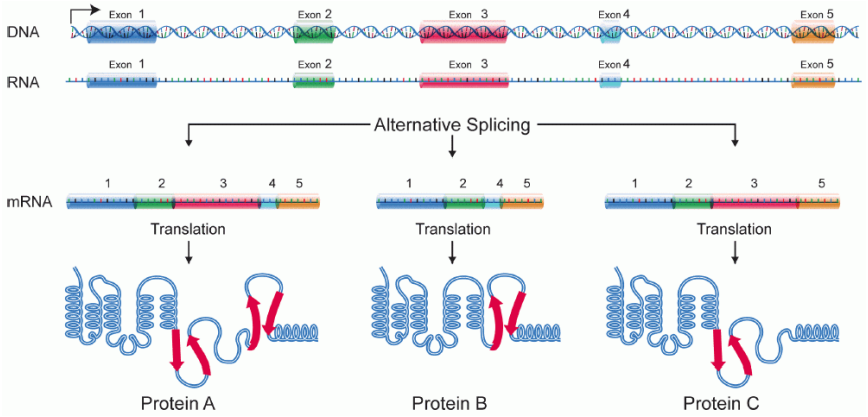
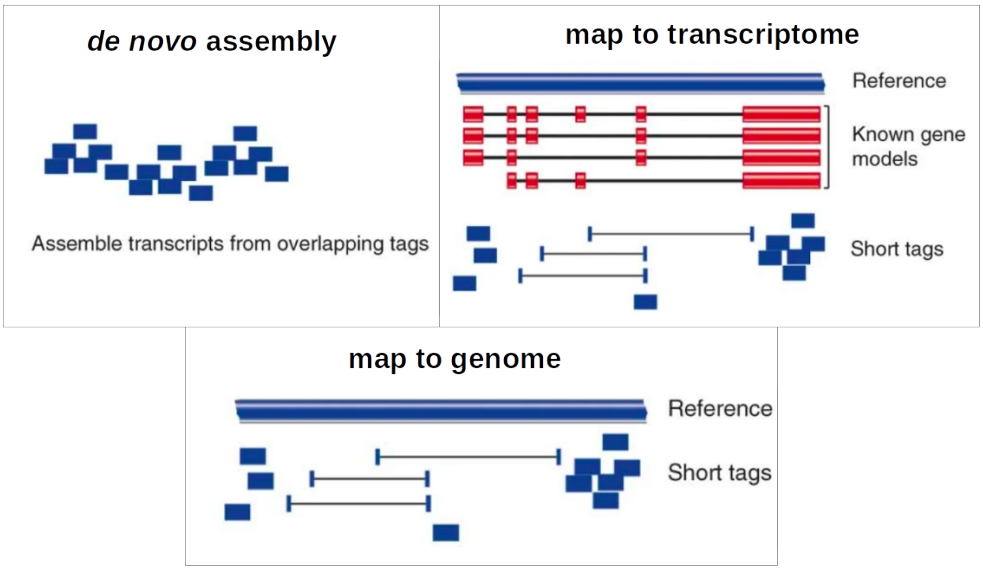


Figure from Wikipedia

Aligning RNAseq reads (cont.)

Three different strategies are possible – but mapping to the genome is preferable as long as a reference genome is available:



Figures from Cloonan & Grimmond 2010

Aligning RNAseq reads (cont.)

Therefore, RNAseq mappers have to be "splice-aware":

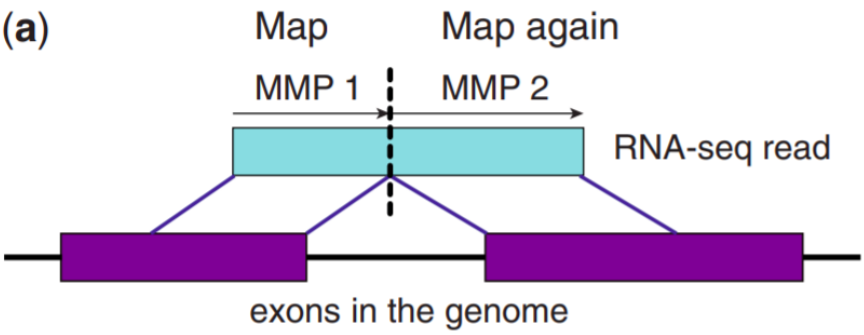


Figure from Dobin et al. 2013

Alignment output: SAM/BAM files

- SAM/BAM files contain the original reads *along with the genomic coordinates to which they were mapped* (and more information).
- We get 1 SAM/BAM file for each sample with both the forward and the reverse reads (not 2 separate files like for FASTQ files).
- SAM is an uncompressed text file and BAM is its binary, compressed counterpart.
    - SAM: Sequence Alignment/Map
    - BAM: Binary Alignment/Map
    - Most software can work directly with BAM files, and you can use programs like Samtools to view them.
- For subsequent analyses, BAM files usually also need to be *sorted* by genomic coordinates (e.g. with Samtools but also STAR).

The STAR aligner

About STAR:

- STAR is an acronym for "Spliced Transcripts Alignment to a Reference".
- Reference paper – Dobin et al. 2013, *Bioinformatics*: STAR: ultrafast universal RNA-seq aligner.
- This page has a nice quick, visual explanation of STAR's alignment strategy.
- It's a very fast aligner, but memory-intensive. Luckily we have OSC!

☐ Before we can perform the mapping, we also need to index the genome with STAR. Every mapper has its own mapping algorith and associated with that also its own way to create a genome dictionary.

STAR at OSC

- STAR is available in a module at OSC. The Owens cluster has a more recent version available, so make sure to log in to the Owens cluster.

```
$ cd /fs/project/PAS0471/teach/misc/2021-02_rn
```

```
$ module load star/2.6.0a
```

STAR options for genome indexing

Some options when generating a genome index with STAR are as follows:

Option	Meaning
`--runThreadN 18`	Use 18 cores/threads
`--runMode genomeGenerate`	The mode to index a genome
`--genomeDir mydir`	Destination dir for index
`--genomeFastaFiles my.fasta`	Path to the genome FASTA file
`--sjdbGTFfile my.gtf`	Path to the genome GTF file
`--sjdbOverhang ReadLength-1`	Length of the donor/acceptor sequence on each side of the junctions
`--genomeSAindexNbases 14`	Length of the SA pre-indexing string

☐ According to the documentation `genomeSAindexNbases` should be:

`log2(GenomeLength)/2 - 1)` => `log2(782520133)/2 - 1)` = 13.77 = 14

The key parts of a script to index the genome

```
#!/bin/bash
#SBATCH --account=PAS0471
#SBATCH --time=2:00:00
#SBATCH --cpus-per-task=18
module load star/2.6.0a

STAR --runThreadN "$SLURM_CPUS_ON_NODE" \
    --runMode genomeGenerate \
    --genomeDir "$index_dir" \
    --genomeFastaFiles "$ref_fa" \
    --sjdbGTFtagExonParentTranscript "$ref_gff" \
    --genomeSAindexNbases 13 \
    --sjdbOverhang ReadLength-1
```

Submit the script:

```
$ ref_fa=data/ref/Heinz1706_4.00/S_lycopersicum_c
$ ref_gff=data/ref/annot/ITAG4.1/ITAG4.1_gene_mod
$ index_dir=data/ref/Heinz1706_4.00/STAR_index

$ sbatch scripts/align/star_index.sh \
    "$ref_fa" "$ref_gff" "$index_dir"
#> Submitted batch job 12826056
```

Check the output

```
$ ls
#> slurm-STAR-index-12826056.out Log.out

$ cat slurm-STAR-index-12826056.out
#> Feb 10 14:53:19 ..... started STAR run
#> Feb 10 14:53:19 ... starting to generate Genom
#> Feb 10 14:53:34 ... starting to sort Suffix Ar
#> Feb 10 14:53:39 ... sorting Suffix Array chunk
#> Feb 10 14:55:09 ... loading chunks from disk,
#> Feb 10 14:55:28 ... finished generating suffix
#> Feb 10 14:55:28 ... generating Suffix Array in
#> Feb 10 14:56:15 ... completed Suffix Array ind
#> Feb 10 14:56:15 ... writing Genome to disk ...
#> Feb 10 14:56:15 ... writing Suffix Array to di
#> Feb 10 14:56:18 ... writing SAindex to disk
#> Feb 10 14:56:18 ..... finished successfully
```

Check the output (cont.)

STAR also automatically outputs a file `Log.out` with lots of information about the run:

```
$ less Log.out

#> STAR version=STAR_2.6.0a
#> STAR compilation time,server,dir=Mon Apr 23 13
#> ##### DEFAULT parameters:
#> versionSTAR                         20201
#> versionGenome                       20101   2020
#> parametersFiles                     -
#> sysShell                            -
#> runMode                             alignReads
#> runThreadN                          1
#> runDirPerm                          User_RWX
#> runRNGseed                          777
#> ...
```

```
$ wc -l Log.out
#> 416
```

Check the output (cont.)

Check the actual indexing output files:

```
$ ls -lh data/ref/Heinz1706_4.00/STAR_index

#> total 7.2G
#> -rw-r--r-- 1 jelmer PAS0471  116 Feb 10 14:53
#> -rw-r--r-- 1 jelmer PAS0471  246 Feb 10 14:53
#> -rw-r--r-- 1 jelmer PAS0471  130 Feb 10 14:53
#> -rw-r--r-- 1 jelmer PAS0471  130 Feb 10 14:53
#> -rw-r--r-- 1 jelmer PAS0471 748M Feb 10 14:56
#> -rw-r--r-- 1 jelmer PAS0471  734 Feb 10 14:56
#> -rw-r--r-- 1 jelmer PAS0471 6.1G Feb 10 14:56
#> -rw-r--r-- 1 jelmer PAS0471 374M Feb 10 14:56
```