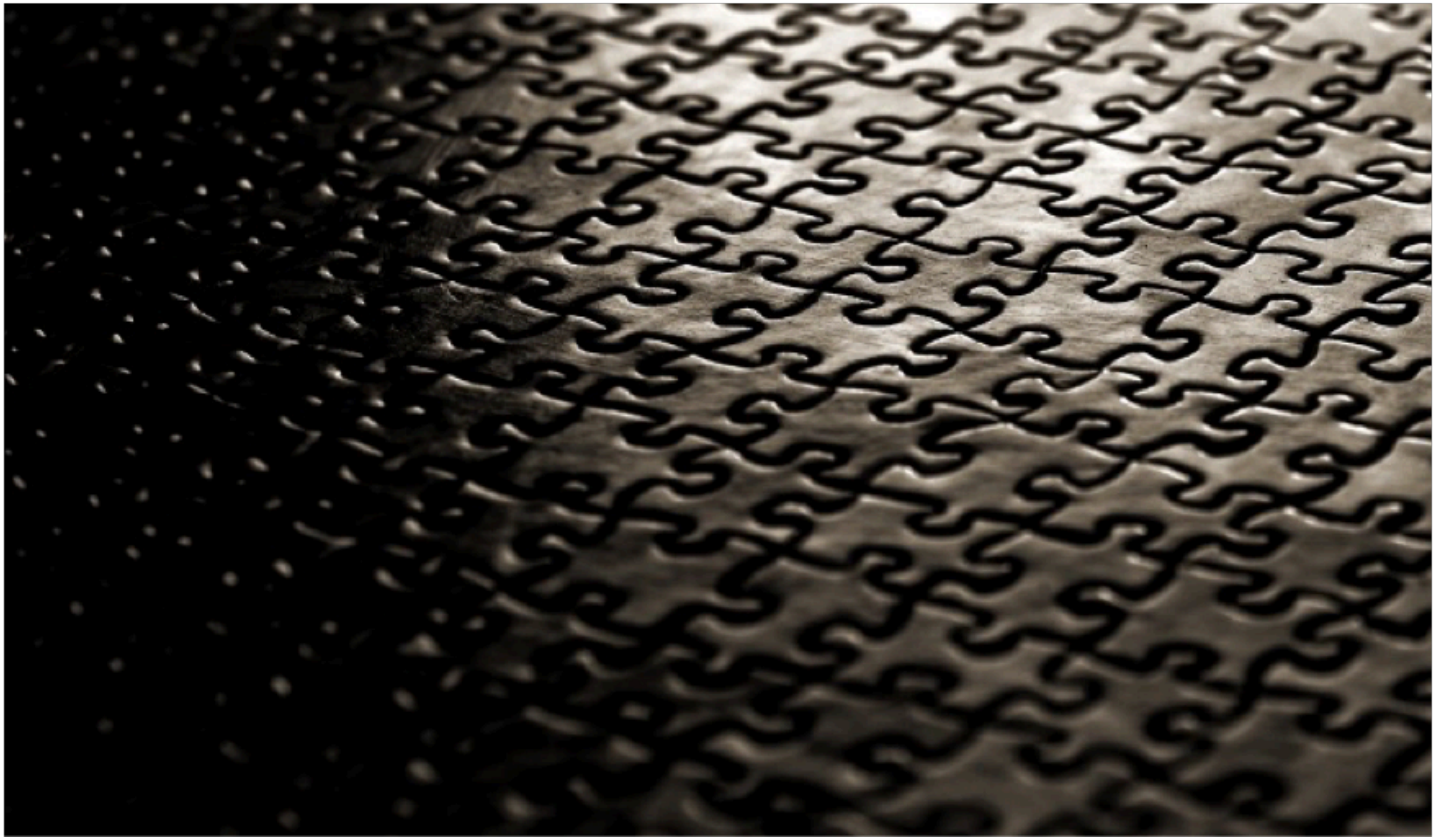
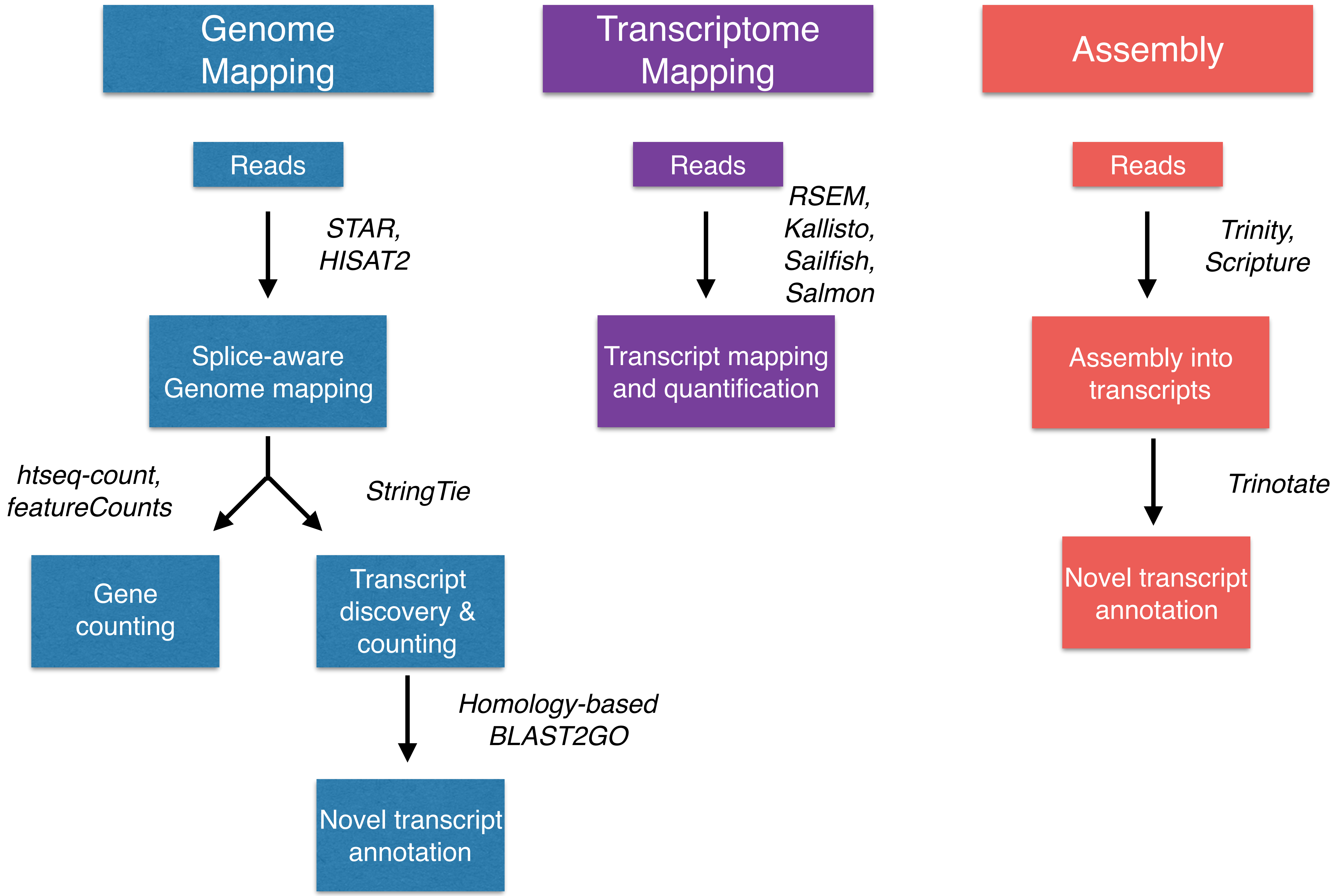
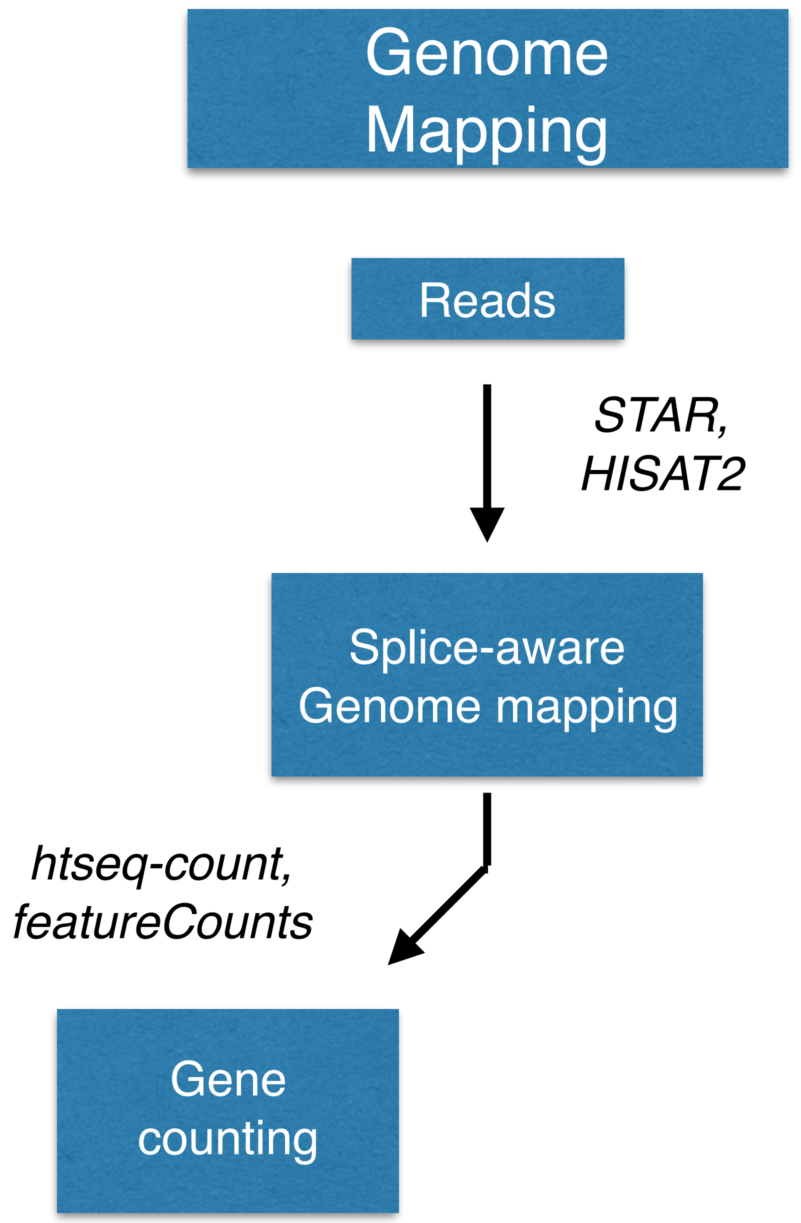


Aligning reads: tools and theory


Genome Mapping
Transcriptome Mapping
Assembly
Reads
STAR, HISAT2
Splice-aware Genome mapping
htseq-count, featureCounts
StringTie
Gene counting
Transcript discovery & counting
Homology-based BLAST2GO
Novel transcript annotation
Reads
RSEM, Kallisto, Sailfish, Salmon
Transcript mapping and quantification
Reads
Trinity, Scripture
Assembly into transcripts
Trinotate
Novel transcript annotation



Former “standard” approach for quantifying gene expression

Reference data files

Reference data files

- **Genome** assembly gives us the nucleotide sequence of the reference genome.

Reference data files

- **Genome** assembly gives us the nucleotide sequence of the reference genome.
    - It is in the form of a Fasta file

Reference data files

- **Genome** assembly gives us the nucleotide sequence of the reference genome.
    - It is in the form of a Fasta file
    - **Genome build/release** represents a version of the genome with improvements (i.e. gaps filled, mistakes corrected).

Reference data files

- **Genome** assembly gives us the nucleotide sequence of the reference genome.
  - It is in the form of a Fasta file
  - **Genome build/release** represents a version of the genome with improvements (i.e. gaps filled, mistakes corrected).
- **Transcriptome** gives us the complete set of transcripts

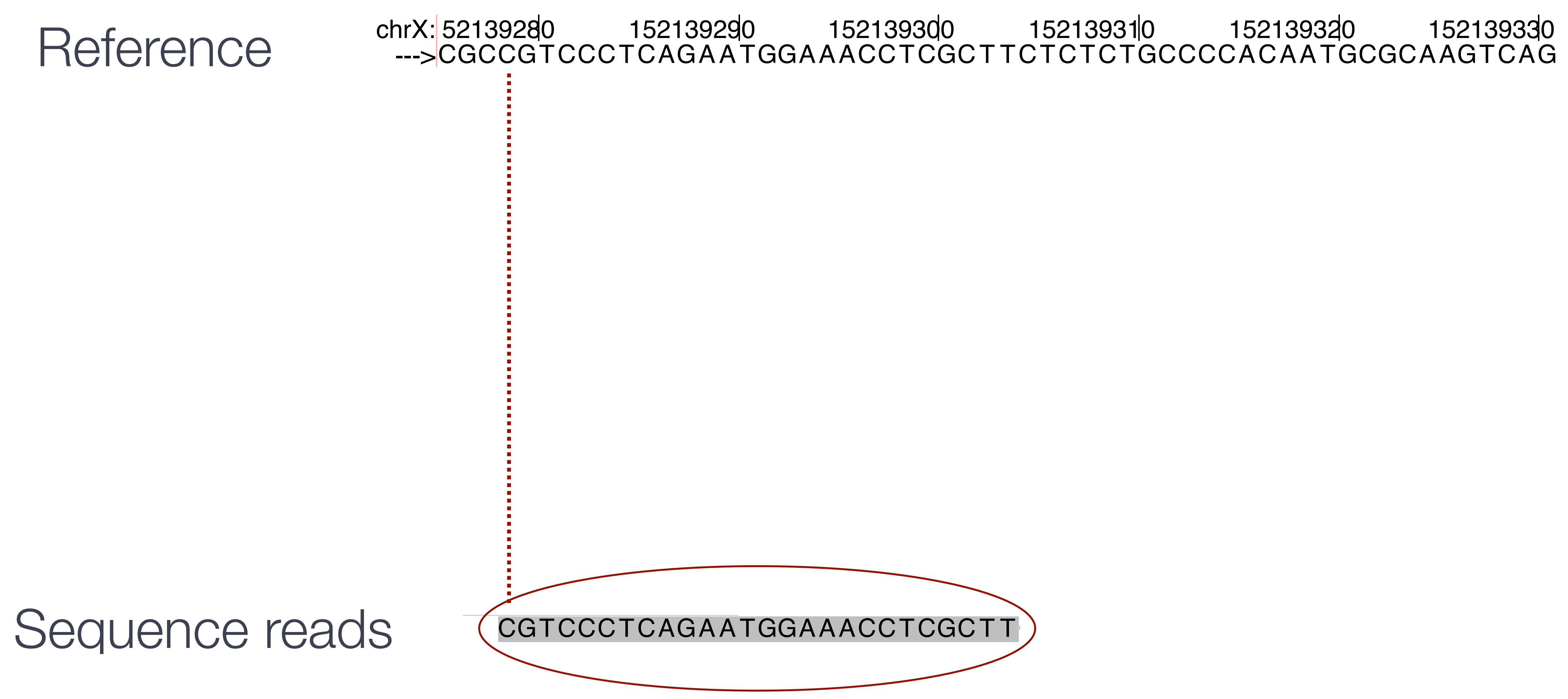
Reference data files

- **Genome** assembly gives us the nucleotide sequence of the reference genome.
  - It is in the form of a Fasta file
  - **Genome build/release** represents a version of the genome with improvements (i.e. gaps filled, mistakes corrected).
- **Transcriptome** gives us the complete set of transcripts
  - It is in the form of a GTF (gene transfer format) file

Reference data versions matter

- Make sure that all reference files used in an analysis are matched (i.e. genome file (FASTA), transcriptome file (FASTA, GTF)
    - Same build version
    - Same source (e.g. both from FlyBase)

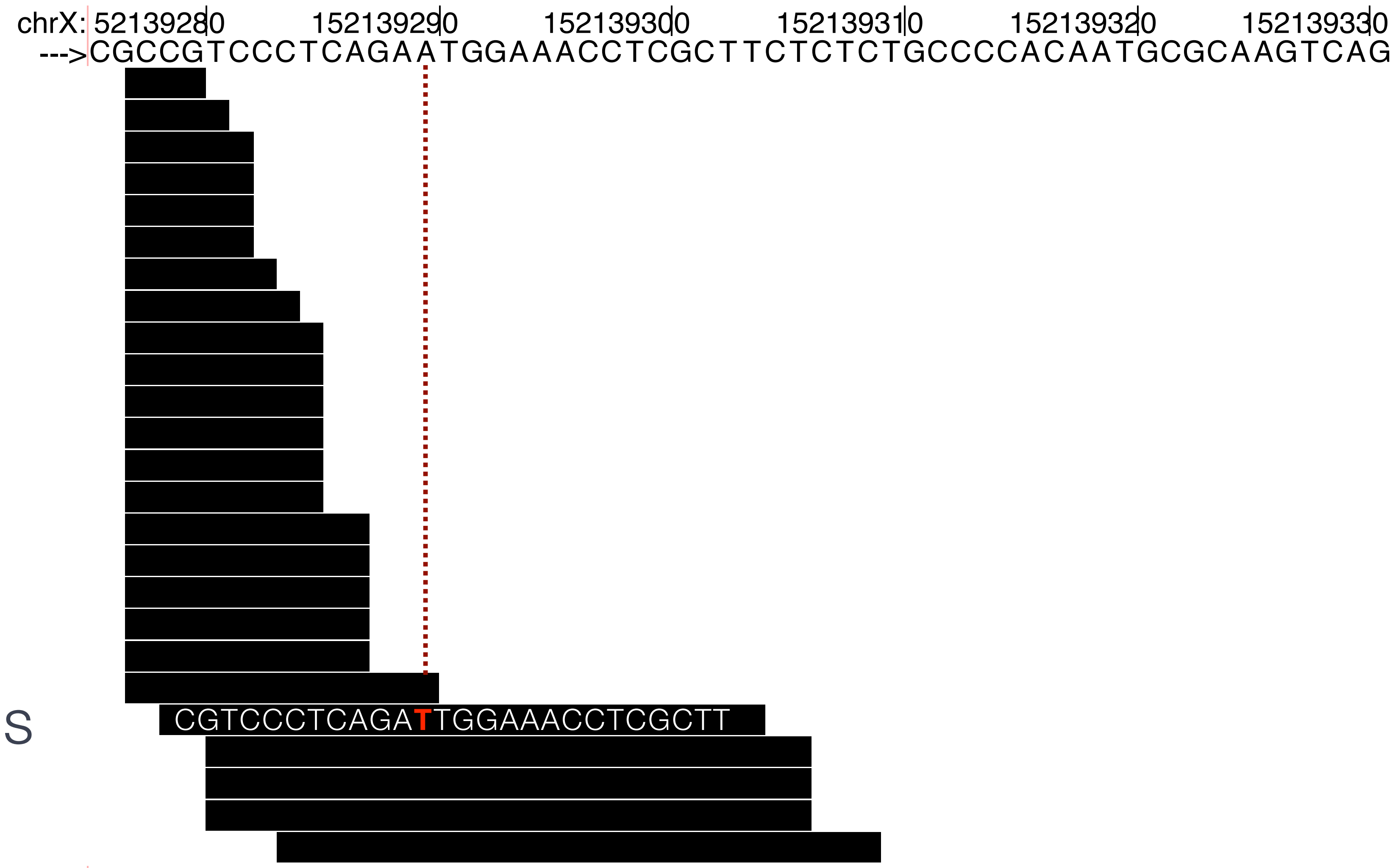
Goal: Finding where in the genome these reads originated from


Reference
chrX: 52139280 152139290 152139300 152139310 152139320 152139330
--->CGCCGTCCCTCAGAATGGAAACCTCGCTTCTCTCTGCCCCACAATGCGCAAGTCAG
Sequence reads
CGTCCCTCAGAATGGAAACCTCGCTT

A simple case of string matching

Reference

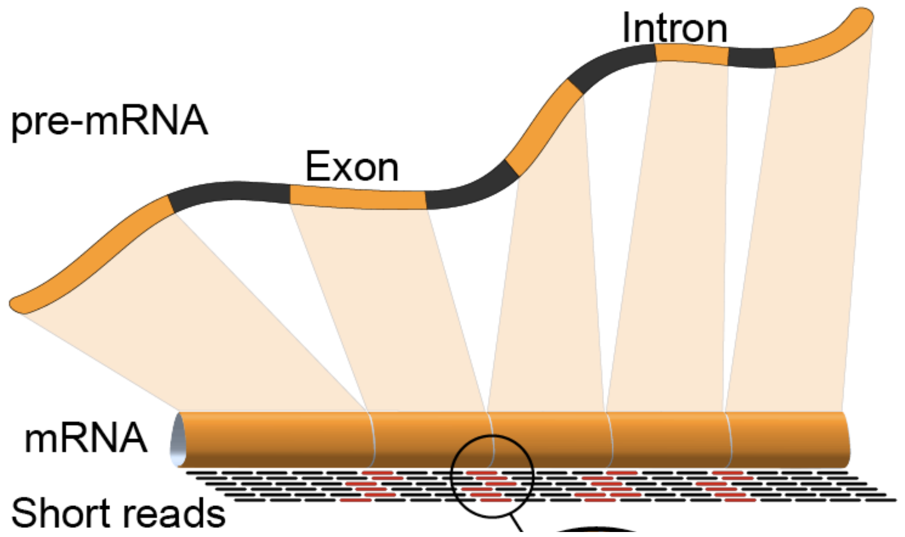
Sequence reads


chrX: 52139280
152139290
152139300
152139310
152139320
152139330
--->CGCCGTCCCTCAGAATGGAAACCTCGCTTCTCTCTGCCCCACAATGCGCAAGTCAG
CGTCCCTCAGATTGGAAACCTCGCTT

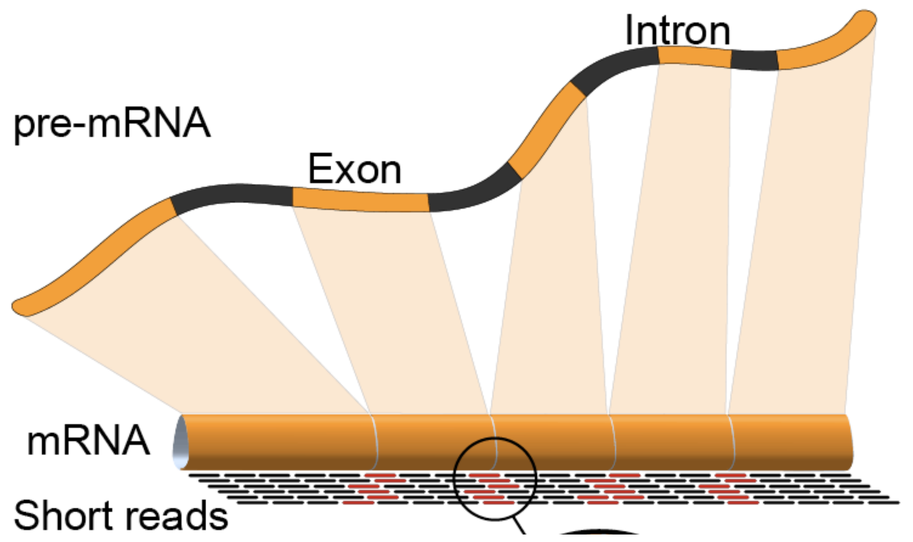
A simple case of string matching?

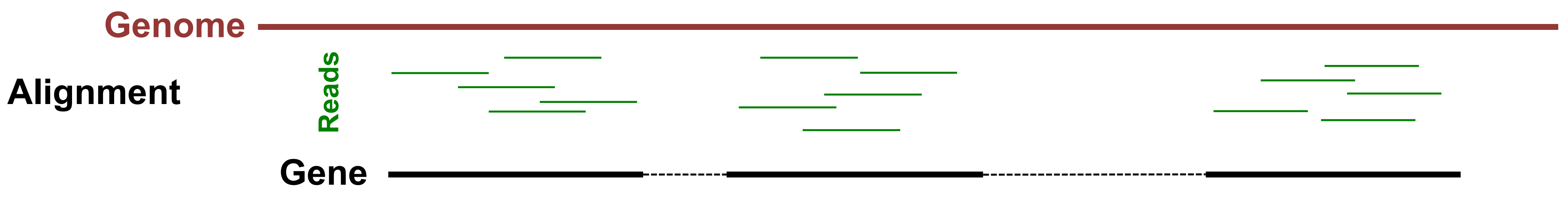
Non-comprehensive list of challenges

- Large, incomplete and repetitive genomes
- Short reads: 50-150 bp
    - Non-unique alignment
    - Sensitive to non-exact matching (variants, sequencing errors)
- Massive number of short reads
- Small insert size: 200-500 bp libraries
- Compute capacity for efficient base-to-base mapping

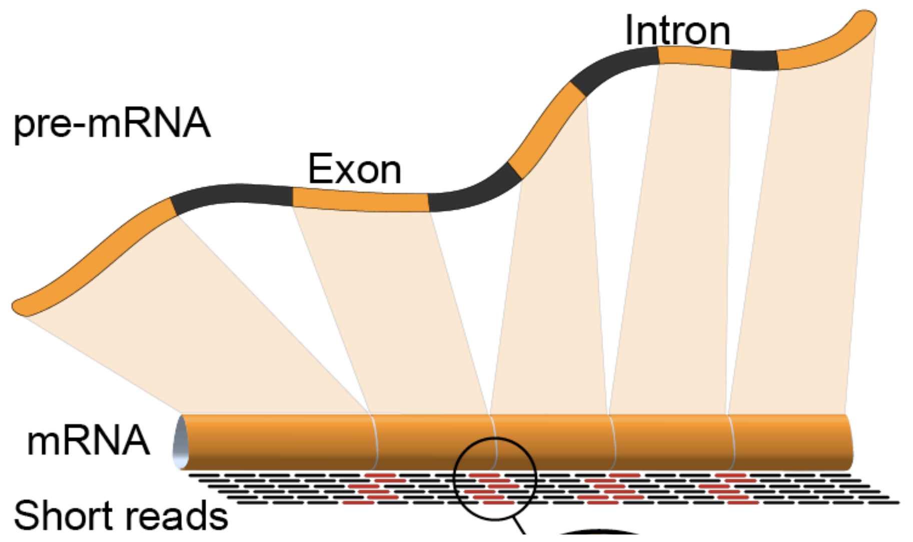


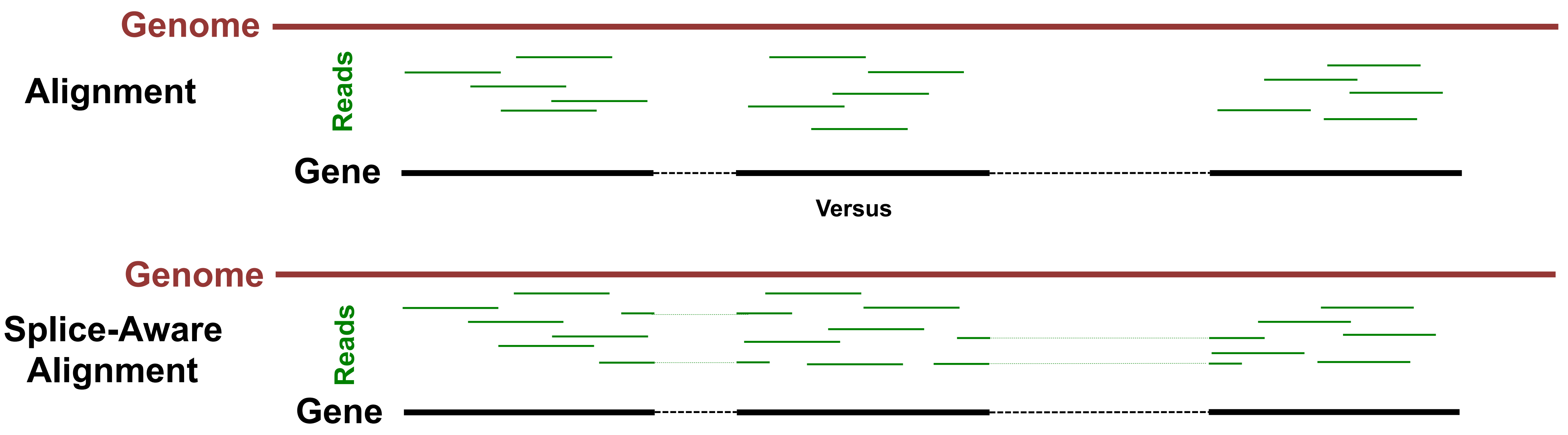
Splice-aware alignment





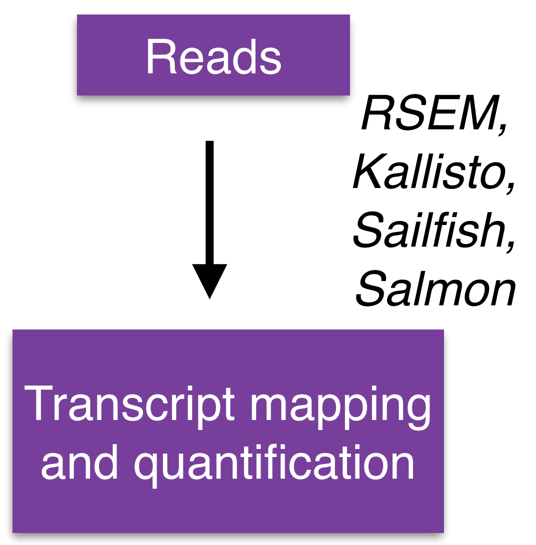
Splice-aware alignment





Splice-aware alignment

Transcriptome Mapping


Reads
RSEM, Kallisto, Sailfish, Salmon
Transcript mapping and quantification

The current standard

for quantifying gene expression

Why use lightweight alignment?

- Approaches avoid base-to-base alignment
- Faster, more efficient (~ >20x faster than alignment-based)
- Improved accuracy for transcript-level quantification
- Improvements in accuracy for gene-level quantification**
- Tools include: Kallisto (quasi-aligner), Sailfish (kmer-based), Salmon (quasi-aligner), RSEM

**doi: 10.12688/f1000research.7563.2

How does Salmon map reads?

Lightweight alignment and quantification using Salmon

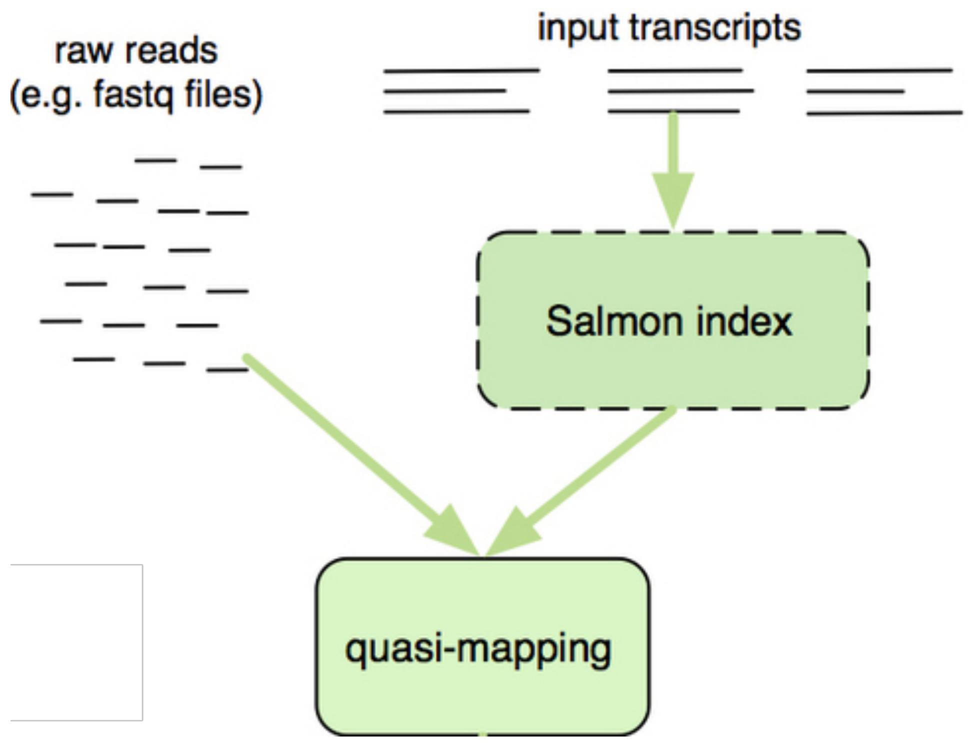


Image source: RNA-seq blog

Lightweight alignment and quantification using Salmon

- Reference = FASTA file of all **transcript sequences** for the organism

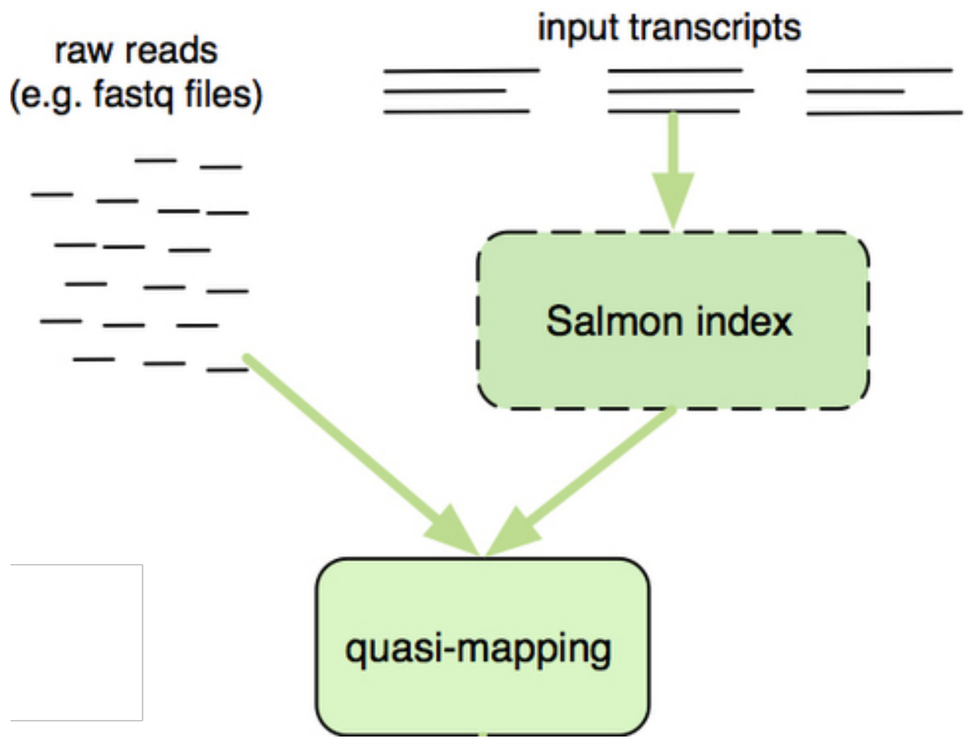


Image source: RNA-seq blog

Transcriptome reference file

- Mapping results are only as good as the quality of the reference transcriptome
- If one does not exist, it can be created using coordinates from a GTF file and the genome sequence file
- Reference data versions matter!
    - Stay consistent with the source and builds/releases being used

Lightweight alignment and quantification using Salmon

- Reference = FASTA file of all **transcript sequences** for the organism
- Reference **Index:** (2 components)
  - Suffix array
  - Hash table (mapping each transcript to its location in the SA)

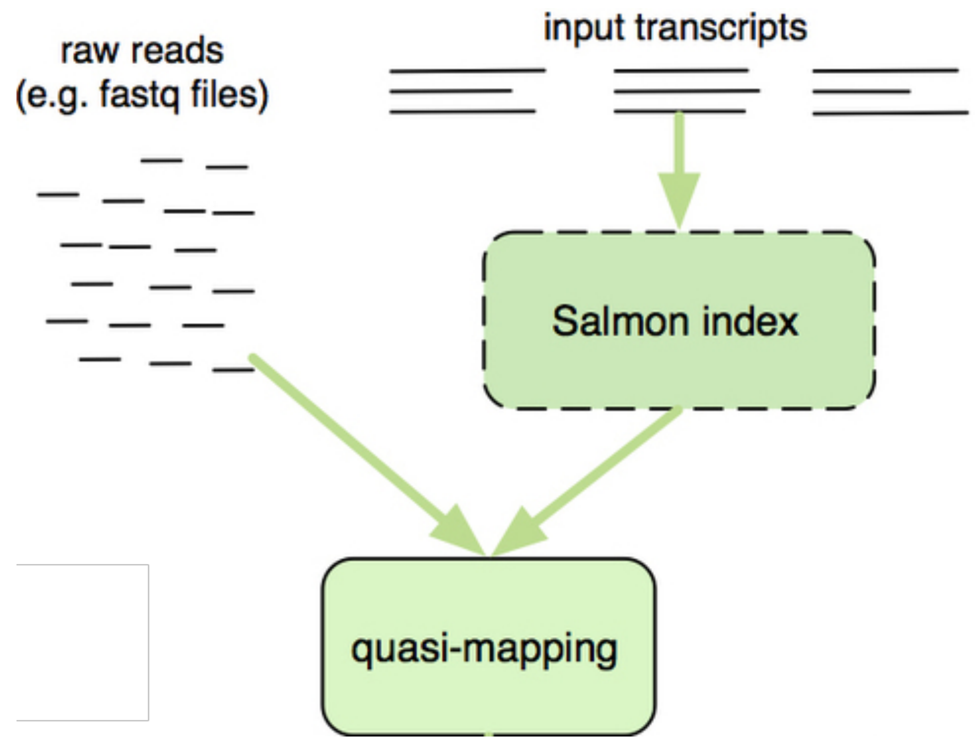


Image source: RNA-seq blog

Building an index

- Having an index of the reference sequence provides an efficient way to search
- Once index is built, it can be queried any number of times
- Every genome or transcriptome build requires a new index for the specific tool in question.

Commonly used indexing methods

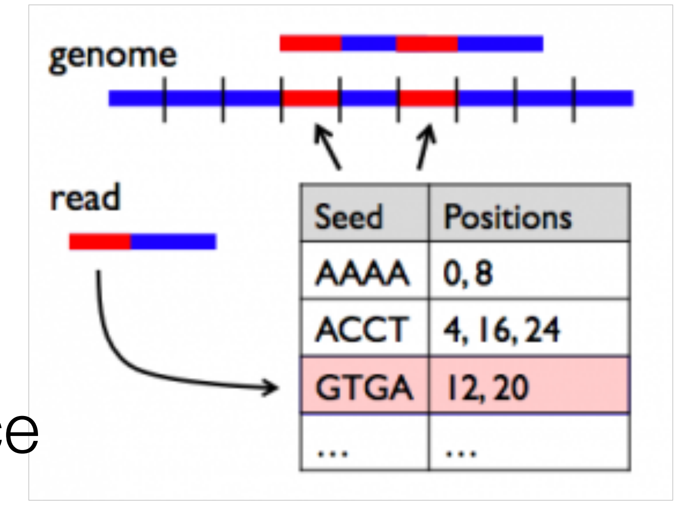
- Hash-based (Salmon, Kallisto)
- Suffix arrays (Salmon, STAR)
- Burrows-Wheeler Transform (BWA, Bowtie2)

Commonly used indexing methods

- Hash-based (Salmon, Kallisto)
- Suffix arrays (Salmon, STAR)
- Burrows-Wheeler Transform (BWA, Bowtie2)

Hash-based alignment (circa 1990)

- Pick k-mer size, build lookup of every k-mer in the reference mapped to its positions (the index)
- Break the query into k-mers
- Seed-and-extend strategy
- For BLAST, 100% match the query k-mer to reference then extend until score drops below 50%
- 0.1 - 1 sec per query; not feasible for NGS data



Seed	Positions
AAAA	0, 8
ACCT	4, 16, 24
GTGA	12, 20
...	...

Hash-based alignment (present day)

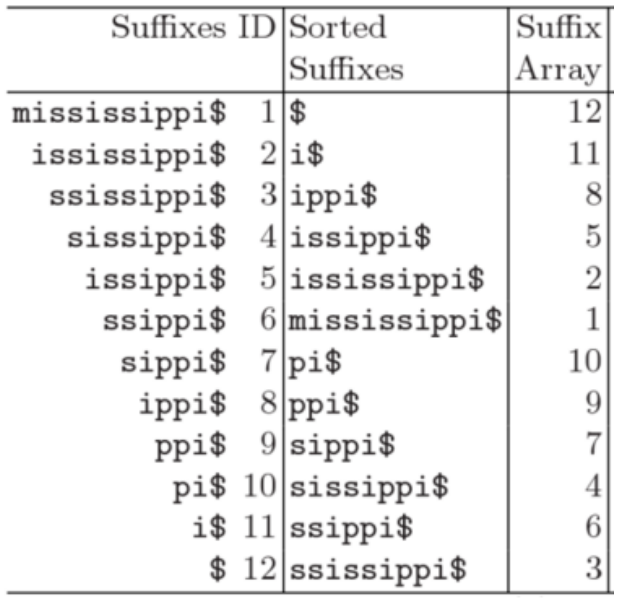
- Need to make some concessions on sensitivity by making adaptations for use on NGS data:
    - allow for mismatches and/or gaps (ELAND, MAQ, SOAP)
    - using multiple seeds (BLAT, ELAND2)
- Memory intensive and slower (~16GB RAM required for hg19)
- Simpler in design but more sensitive

Suffix arrays

- A sorted table of all suffixes (substrings) of a given string
- A suffix array will contain integers that represent the starting indexes of the all the suffixes of a given string, after the aforementioned suffixes are sorted
- Requires large amount of memory to load the suffix array and genome sequence prior to alignment
- Popular Tools:

  STAR (2012), Salmon

Let the given string be “`mississippi`”



Suffixes	ID	Sorted Suffixes	Suffix Array
mississippi$	1	$	12
ississippi$	2	i$	11
ssissippi$	3	ippi$	8
sissippi$	4	issippi$	5
issippi$	5	ississippi$	2
ssippi$	6	mississippi$	1
sippi$	7	pi$	10
ippi$	8	ppi$	9
ppi$	9	sippi$	7
pi$	10	sissippi$	4
i$	11	ssippi$	6
$	12	ssissippi$	3

The suffix array will be:
{12, 11, 8, 5, 2, 1, 10, 9, 7, 4, 6, 3}

Lightweight alignment and quantification using Salmon

- Reference = FASTA file of all **transcript sequences** for the organism
- Reference **Index:** (2 components)
  - Suffix array
  - Hash table (mapping each transcript to its location in the SA)
- Output:
  - **Abundance estimates** of reads mapping to each transcript listed in the reference

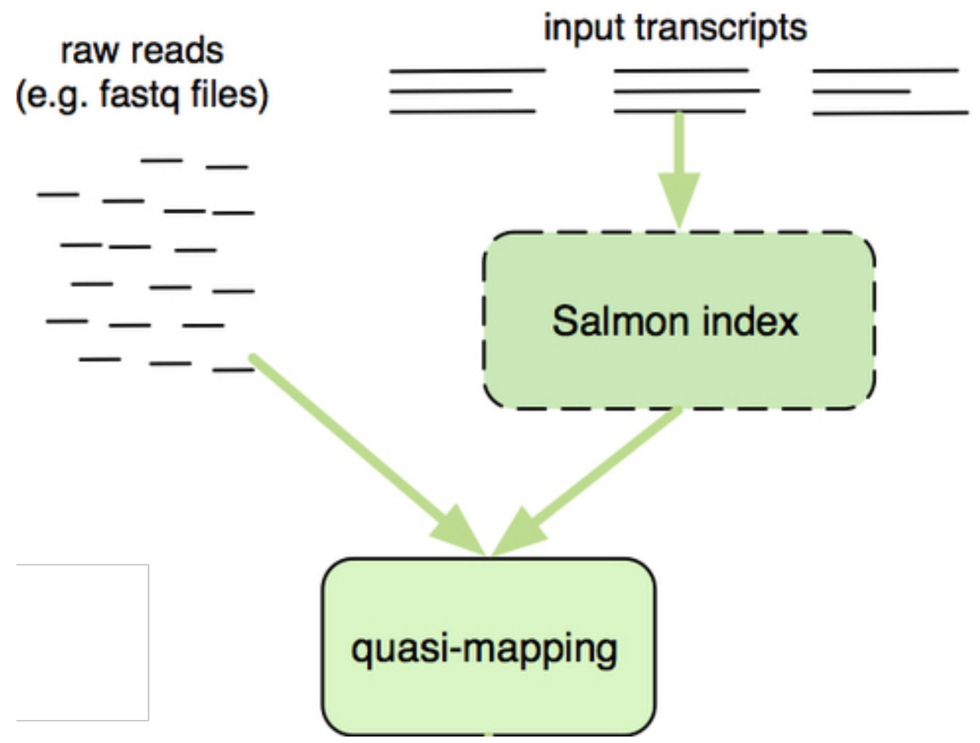


Image source: RNA-seq blog

Lightweight alignment and quantification using Salmon

- Reference = FASTA file of all **transcript sequences** for the organism
- Reference **Index:** (2 components)
  - Suffix array
  - Hash table (mapping each transcript to its location in the SA)
- Output:
  - **Abundance estimates** of reads mapping to each transcript listed in the reference

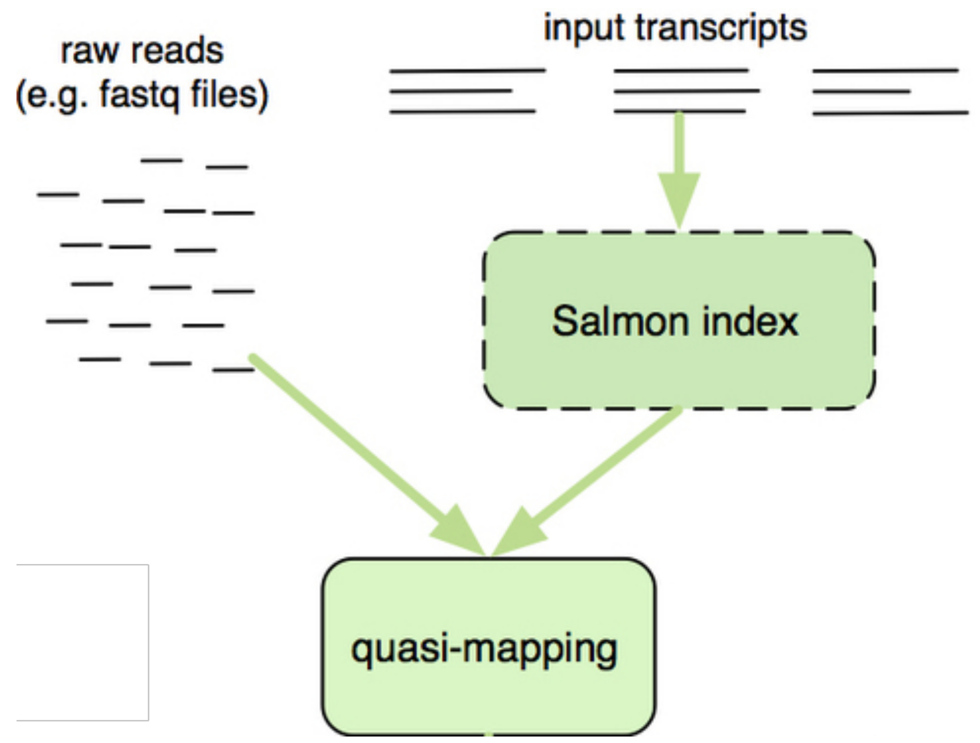
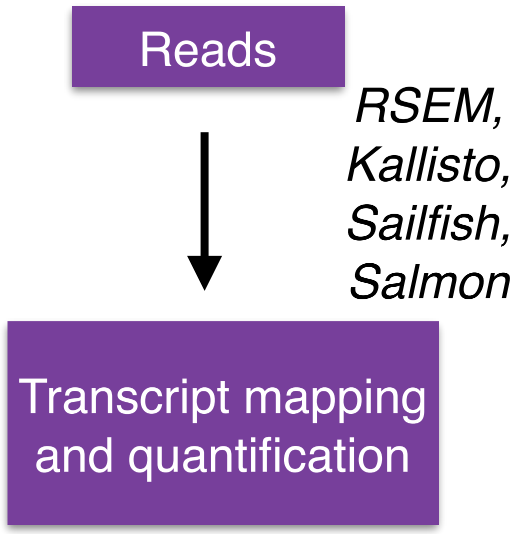


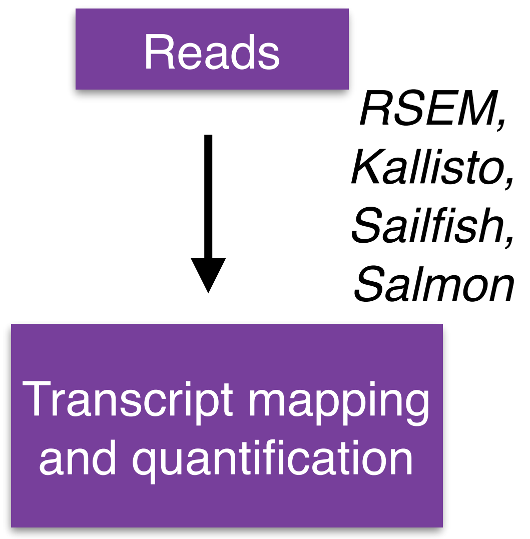
Image source: RNA-seq blog

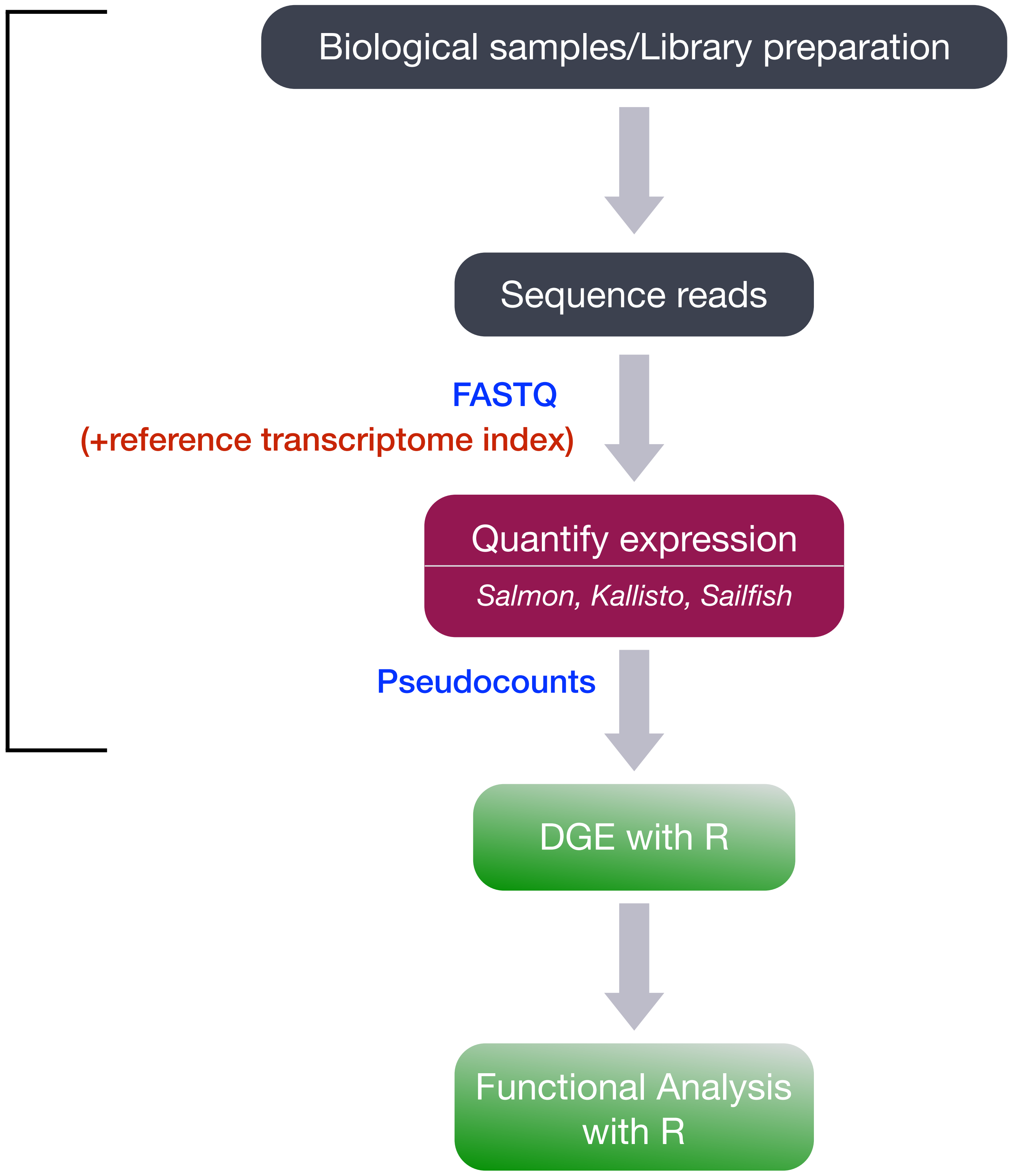
Note that we don't get the <u>genomic coordinates</u> of where each read is mapping with this approach!

Transcriptome Mapping


Reads
RSEM, Kallisto, Sailfish, Salmon
Transcript mapping and quantification

Transcriptome Mapping




Biological samples/Library preparation
Sequence reads
FASTQ
(+reference transcriptome index)
Quantify expression
Salmon, Kallisto, Sailfish
Pseudocounts
DGE with R
Functional Analysis with R