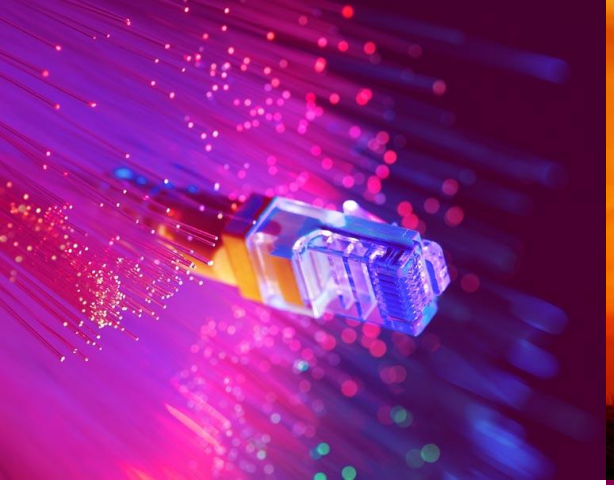


Single-cell RNA-seq data analysis using Chipster

14.3.2019

chipster@csc.fi

CSC – Suomalainen tutkimuksen, koulutuksen, kulttuurin ja julkishallinnon ICT-osaamiskeskus

Understanding data analysis - why?

- Bioinformaticians might not always be available when needed
- Biologists know their own experiments best
  - Biology involved (e.g. genes, pathways, etc)
  - Potential batch effects etc
- Allows you to design experiments better → less money wasted
- Allows you to discuss more easily with bioinformaticians

What will I learn?

- How to operate the Chipster software
- Analysis of single cell RNA-seq data
  - Central concepts
  - File formats
  - Analysis steps, practised in exercises
    1. DropSeq data preprocessing from raw reads to expression values
    2. Clustering analysis of 10X Genomics data with Seurat tools
    3. Integrated analysis of two samples with Seurat tools

Introduction to Chipster

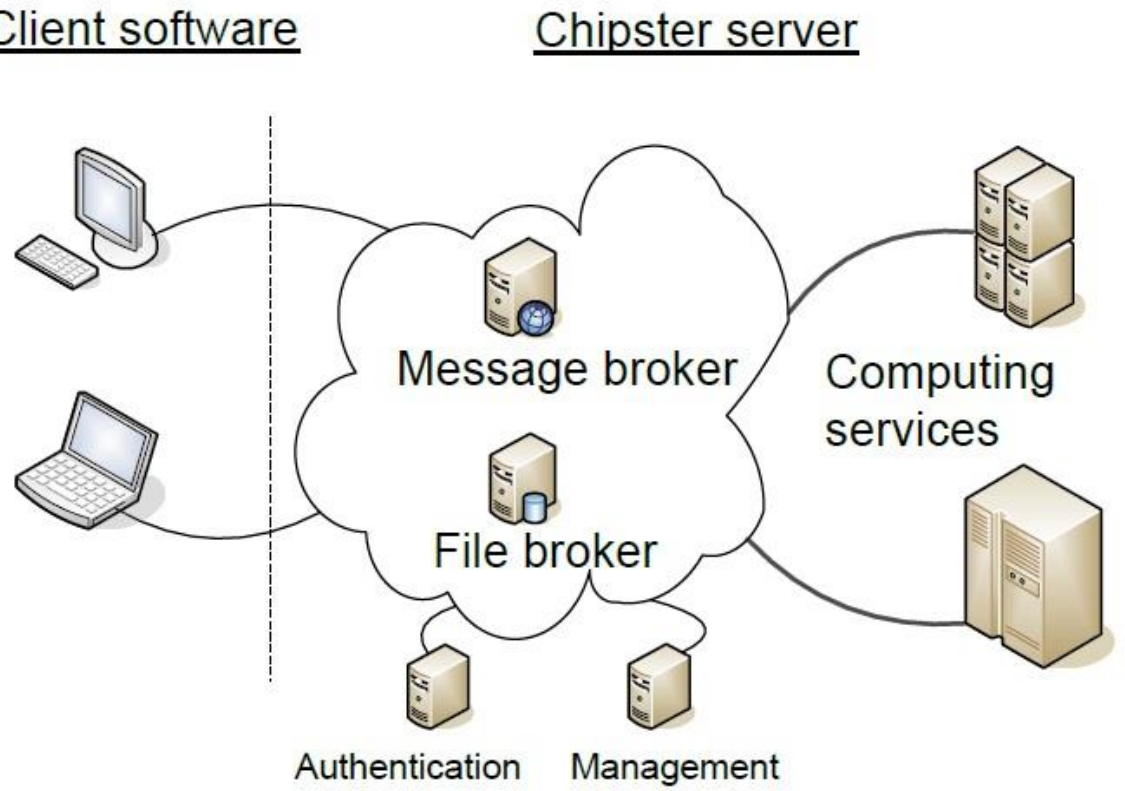
Chipster

- Provides an easy access to over 400 analysis tools
  - Command line tools
  - R/Bioconductor packages
- Free, open source software

- What can I do with Chipster?
  - analyze and integrate high-throughput data
  - visualize data efficiently
  - share analysis sessions
  - save and share automatic workflows

Technical aspects

- **Client-server system**
  - Enough CPU and memory for large analysis jobs
  - Centralized maintenance
- **Easy to install**
  - Client uses Java Web Start
  - Server available as a virtual machine image



Chipster

Open source platform for data analysis

- Home
- Getting access
- Analysis tool content
- Screenshots
- Manual
- Tutorial videos
- Course material
- Cite
- FAQ
- Contact

- For developers:
  - Open source project
  - Tool editor

Welcome to Chipster

Chipster is a user-friendly software for analyzing high-throughput data such as NGS and microarrays. It contains over 400 analysis tools and a large collection of reference genomes. Users can save and share automatic analysis workflows, and visualize data interactively using for example the built-in genome browser. Chipster's client software uses Java Web Start to install itself automatically, and it connects to computing servers for the actual analysis. Chipster is open source, and the server environment is available as a virtual machine image free of charge. If you would like to use Chipster running on CSC's server, you need a user account.

Launch Chipster v3.14

...or launch with more memory: 3 GB or 6 GB

If you have trouble launching Chipster, read this

News and resources:

- 24.9.2018 Video tutorials for single cell RNA-seq data analysis available!
- 13.9.2018 Version 3.14 released
- 17.4.2018 RNA-seq tutorial for differential expression analysis
- 19.8.2014 RNA-seq data analysis guidebook with Chipster instructions
- News archive

Training:

- 14.-15.3.2019 Single cell RNA-seq data analysis, CSC
- 11.12.2018 Community analysis of amplicon sequencing data, Evira
- 19.9.2018 Single cell RNA-seq data analysis, CSC
- 4.-5.9.2018 RNA-seq data analysis, University of Oulu
- 8.8.2018 Community analysis of amplicon sequencing data, JyU
- 26.-28.6.2018 Expression data analysis, DKFZ
- 9.2.2018 Single cell RNA-seq data analysis, CSC
- 16.1.2018 Webinar: VirusDetect pipeline

Chipster 3.14.2

File Edit View Workflow Help

Datasets

- drseq_read_2_fastqc.html
- tagging_and_trimming_summary.txt
- tagging_and_trimming_histograms.pdf
- drseq_read_1.fq.gz
- drseq_read_1_unaligned.bam
- drseq_read_1.bam
- drseq_read_1.bam.bai
- hisat.log
- drseq_read_1_fastqc.html
- drseq_merged_tagged.bam
- drseq_merged_tagged.bam.bai
- inflectionPoint.pdf
- cell_readcounts.txt.gz
- synthesis_stats_summary.txt
- digital_expression.tsv
- cleaned.bam

Analysis tools

Microarrays | NGS | Misc

- Quality control
- Preprocessing
- Utilities
- Matching sets of genomic regions
- Alignment
- Variants
- RNA-seq
- Small RNA-seq
- Single cell RNA-seq
- ChIP- and DNase-seq
- 16S rRNA sequencing
- CNA-seq

- Preprocessing DropSeq FASTQ files
- Merge aligned and unaligned BAM
- Estimate number of usable cells
- Create digital gene expression matrix
- Seurat -Setup and QC
- Seurat -Filtering, regression and detection of variable genes
- Seurat -PCA
- Seurat -Clustering
- Seurat -Visualize biomarkers
- Seurat -Combine two samples and perform CCA
- Seurat -Integrated analysis of two samples
- Seurat -Find conserved cluster markers and DE genes in two samples
- Seurat -Visualize genes with cell type specific responses in two samples

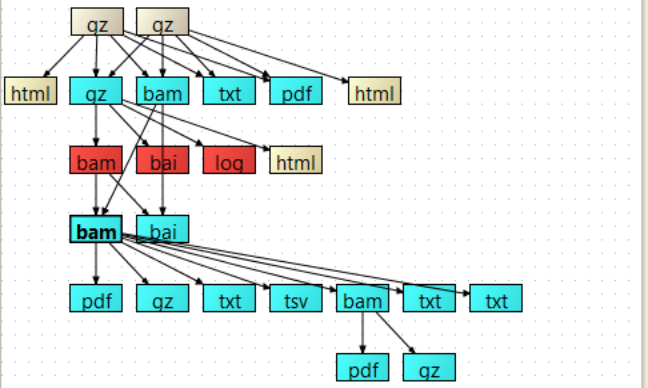
Show parameters | Run

Corrects bead synthesis errors and extracts gene expression values from a BAM file where reads have been tagged with gene names. The resulting Digital Gene Expression matrix, DGE, can be used for further analysis with the Seurat tools.

More help | Show tool sourcecode

Workflow

Fit



Visualisation

Maximise | Detach | Close

drseq_merged_tagged.bam

574 MB, Wed Jan 02 08:47:02 EET 2019

(Click here to add your notes)

Created with Chipster 3.14.1

Analysis history

Single cell RNA-seq / Merge aligned and unaligned BAM

Reference genome	Mus_musculus.GRCm38
Reference GTF	Mus_musculus.GRCm38.92

BAM viewer

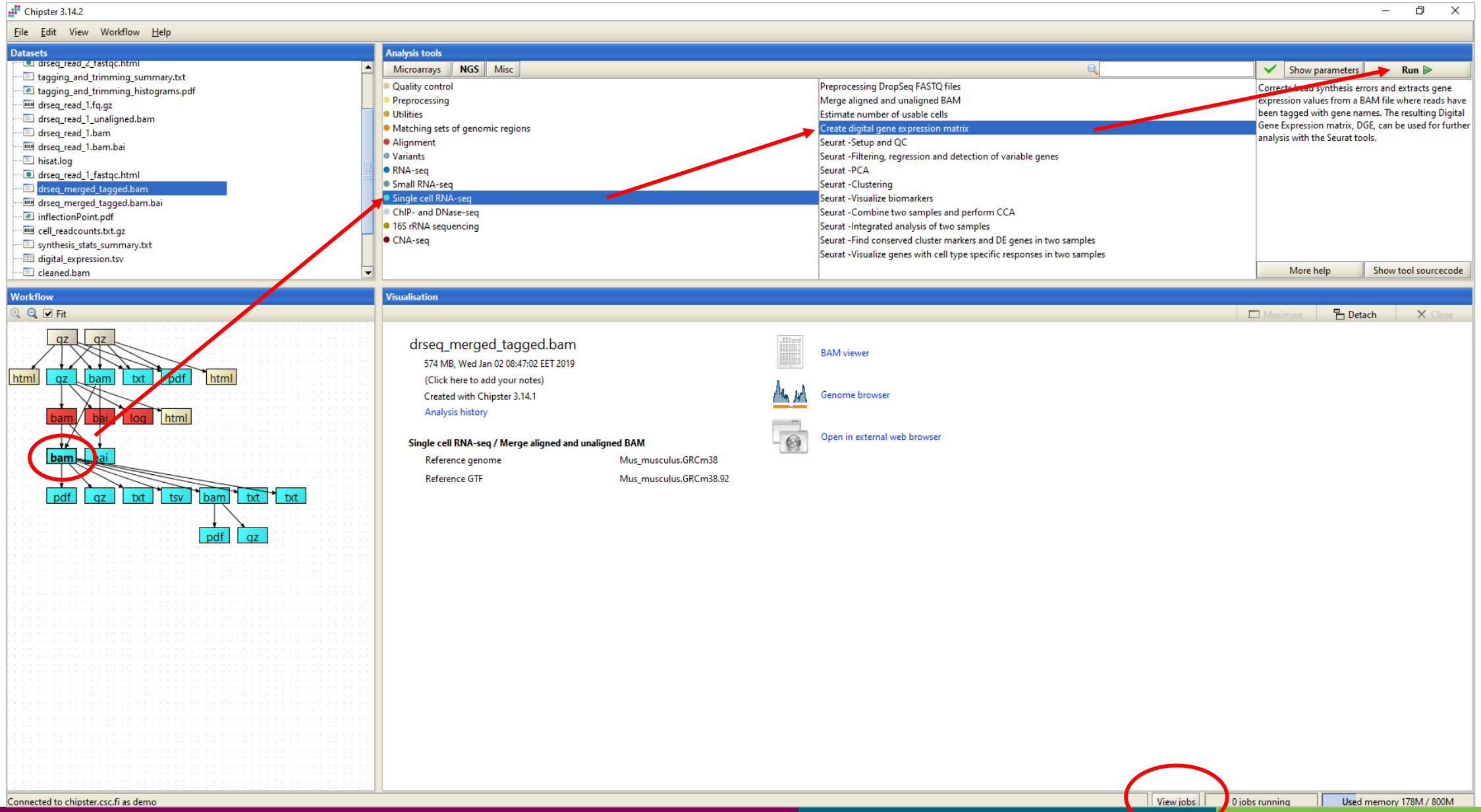
Genome browser

Open in external web browser

Connected to chipster.csc.fi as demo | View jobs | 0 jobs running | Used memory 178M / 800M

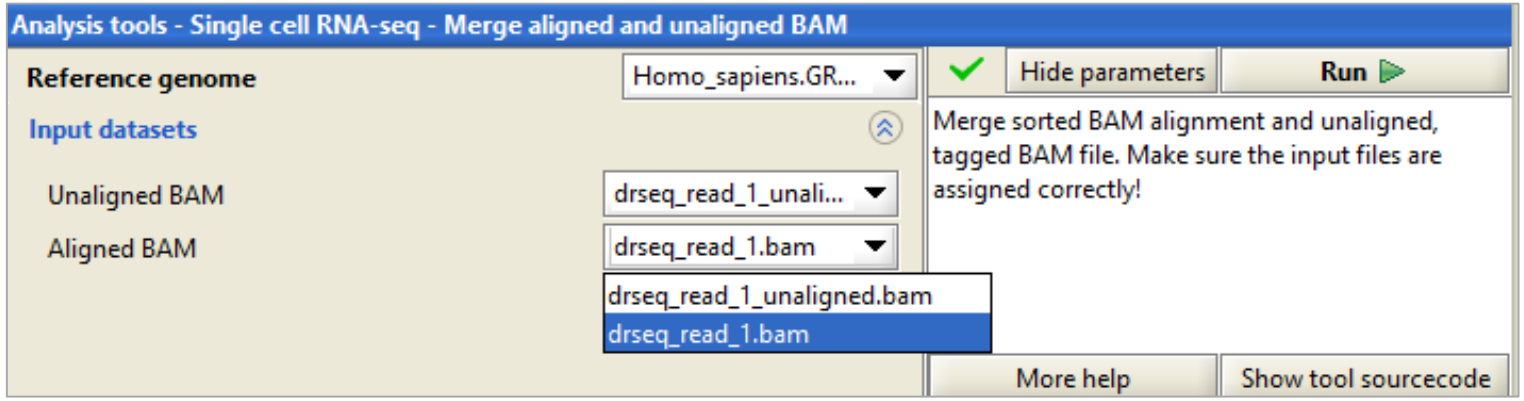
Mode of operation

Select: data → tool category → tool → run → visualize result



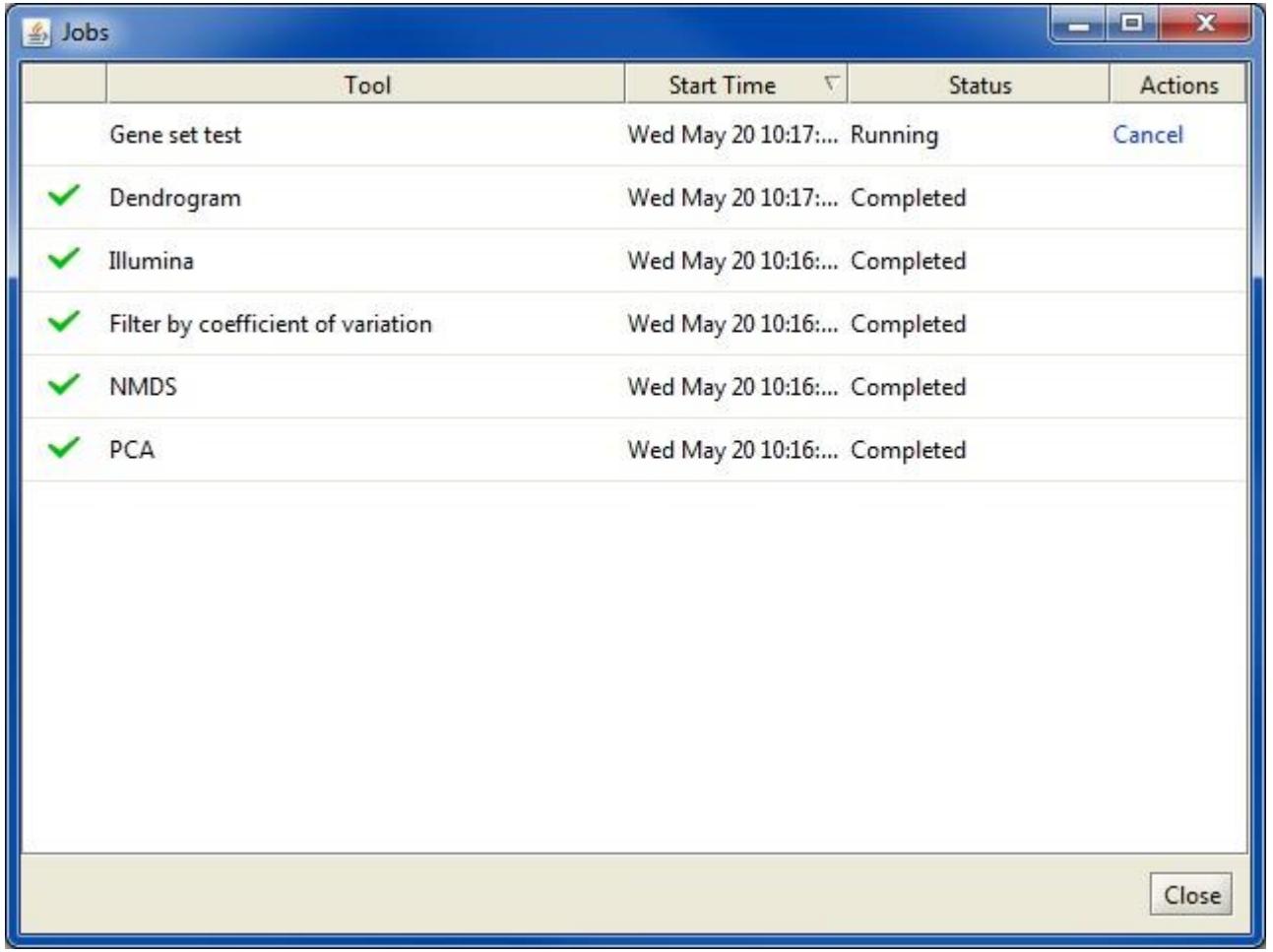
When running analysis tools, pay attention to parameters!

- make sure the input files are correctly assigned if there are multiple files (see below)
- choose the right reference genome
- check especially the **bolded** parameters



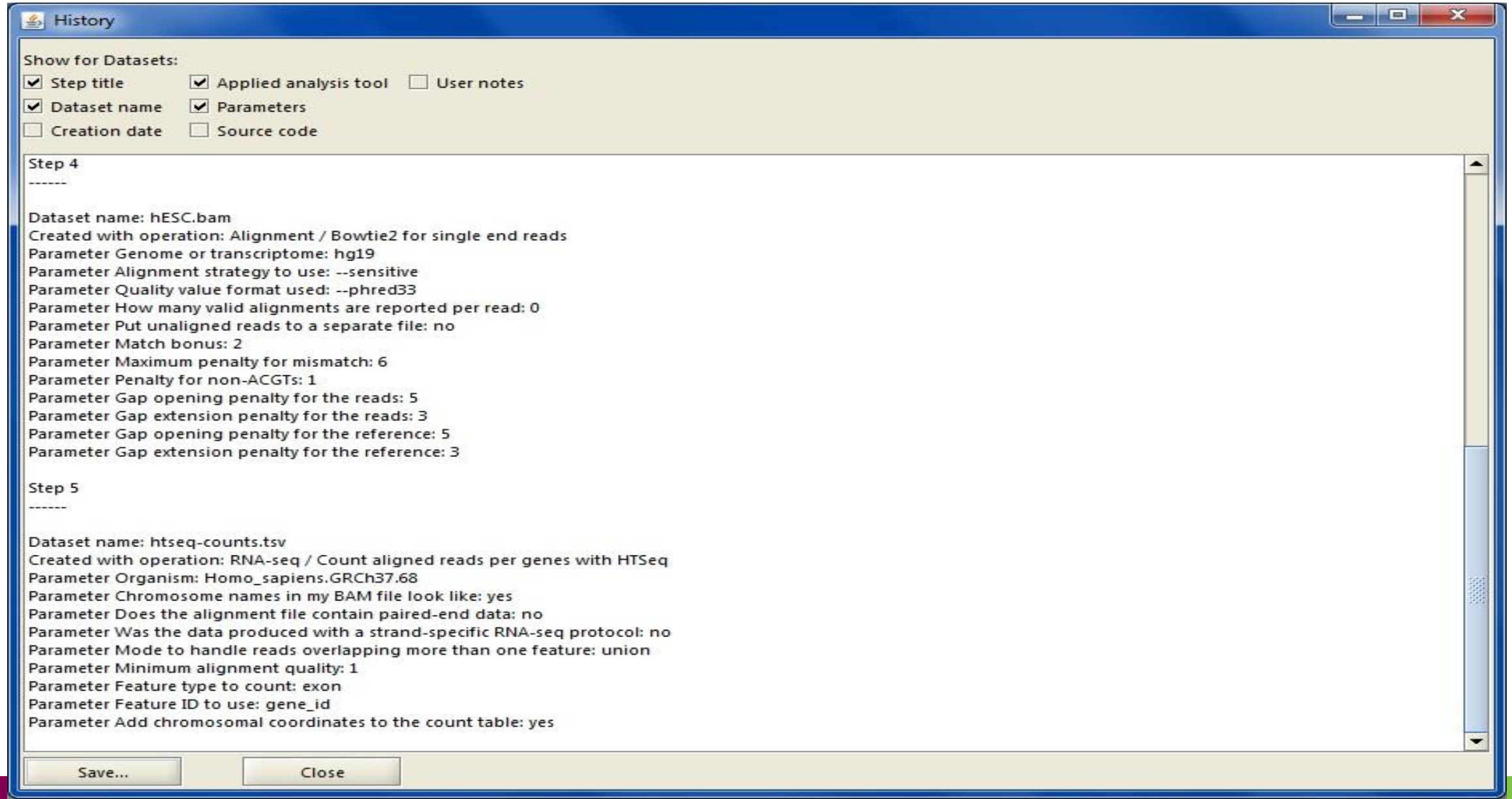
Job manager

- **You can run many analysis jobs at the same time**
- **Use Job manager to:**
  - view status
  - cancel jobs
  - view time
  - view parameters



Analysis history is saved automatically

-you can add tool source code to reports if needed

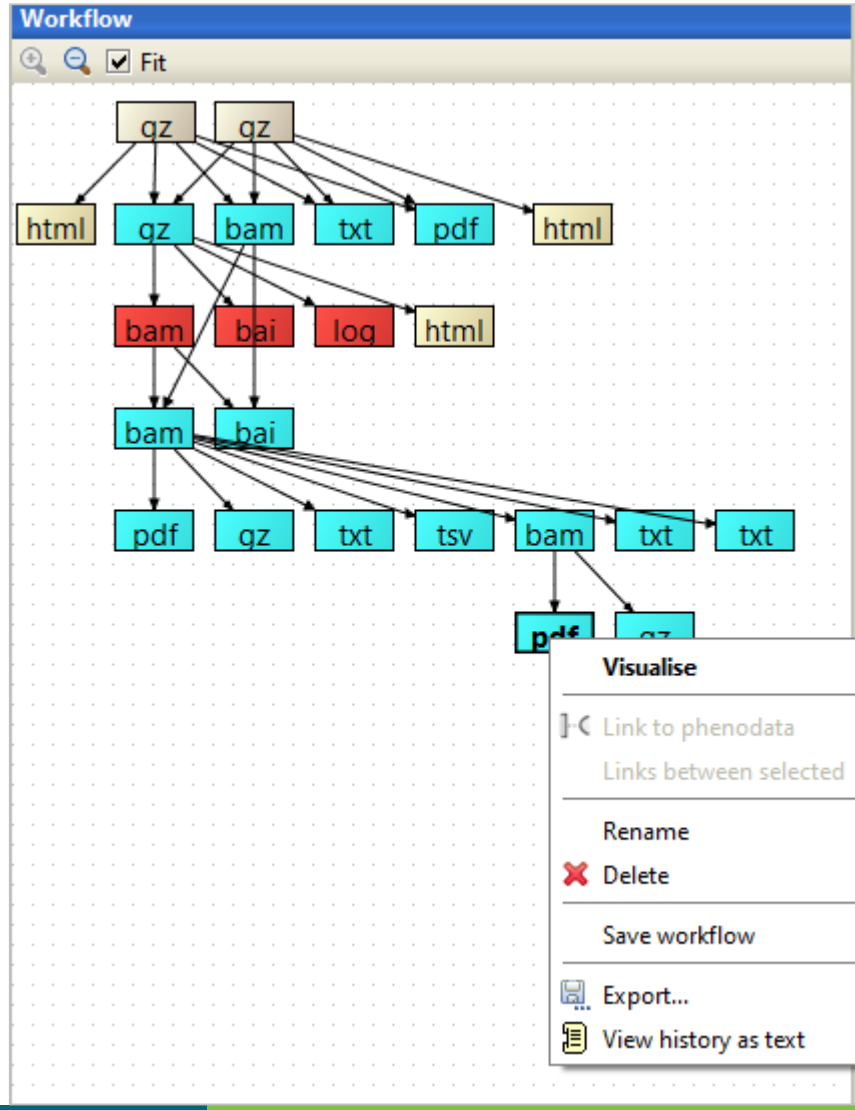


Analysis sessions

- **Remember to save the analysis session within 3 days**
  - Session includes all the files, their relationships and metadata (what tool and parameters were used to produce each file)
  - Session is a single .zip file
  - Note that you can save two sessions of the same data
    - one with raw data (FASTQ files)
    - one smaller, working version where the FASTQ files are deleted after alignment
- **You can save a session locally (= on your computer)**
- **and in the cloud**
  - but note that the cloud sessions are not stored forever!
  - If your analysis job takes a long time, you don't need to keep Chipster open:
    - Wait that the data transfer to the server has completed (job status = running)
    - Save the session in the cloud and close Chipster
    - Open Chipster within 3 days and save the session containing the results

Workflow panel

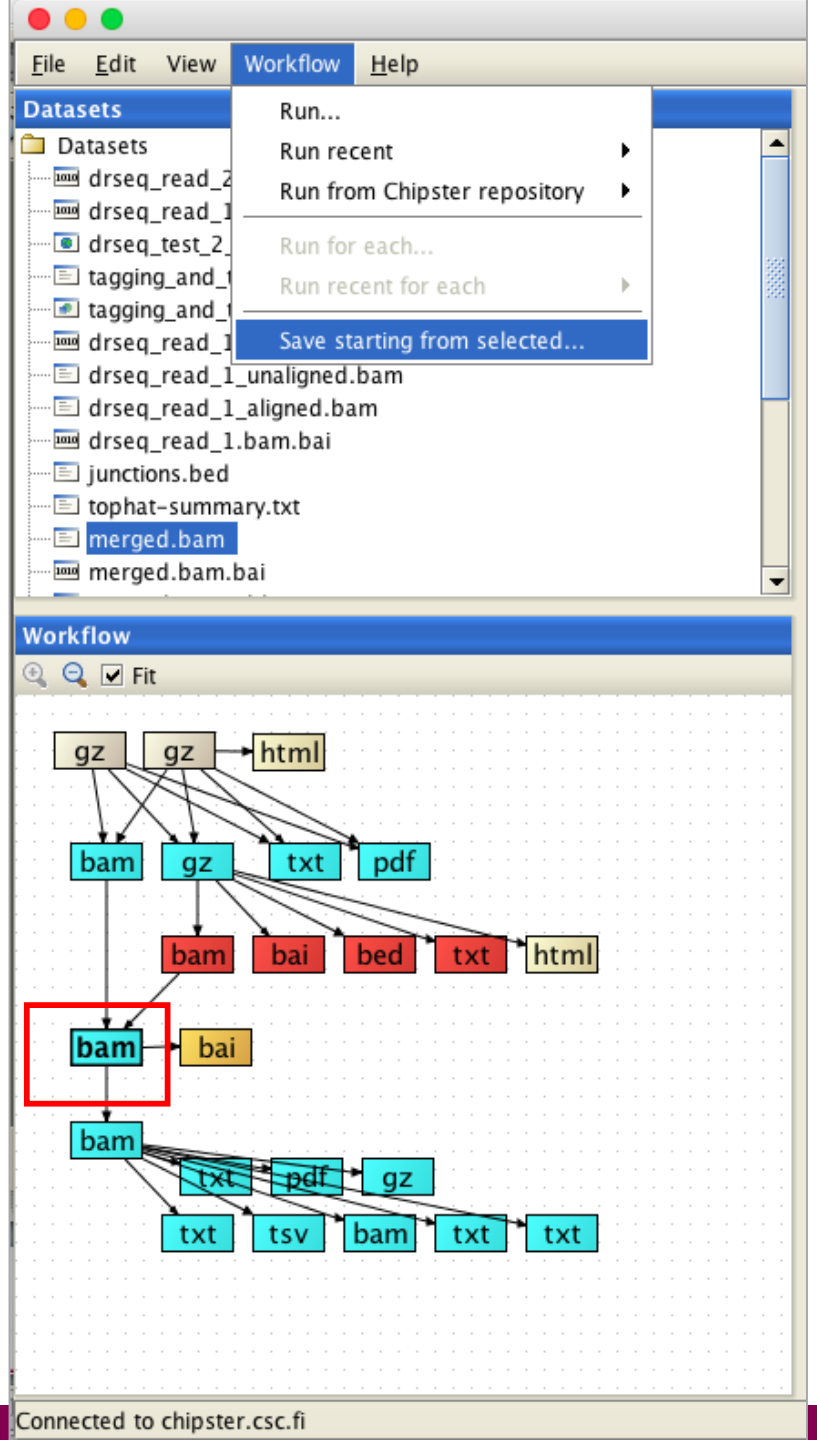
- Shows the relationships of the files
- You can move the boxes around, and zoom in and out.
- Several files can be selected by keeping the Ctrl key down
- Right clicking on the data file allows you to
  - Save an individual result file (”Export”)
  - Delete
  - Link to another data file
  - Save workflow



Workflow – reusing and sharing your analysis pipeline

- You can save your analysis steps as a reusable automatic ”macro”
  - all the analysis steps and their parameters are saved as a script file
  - you can apply workflow to another dataset
  - you can share workflows with other users

Saving and using workflows



- Select the starting point for your workflow
- Select "Workflow / Save starting from selected"
- Save the workflow file on your computer with a meaningful name
  - Don't change the ending (.bsh)!
- To run a workflow, select
  - Workflow -> Open and run
  - Workflow -> Run recent (if you saved the workflow recently).

Analysis tool overview

- **200 NGS tools for**
  - RNA-seq
  - single cell RNA-seq
  - small RNA-seq
  - 16S rRNA amplicon seq
  - exome/genome-seq
  - ChIP-seq
  - FAIRE/DNase-seq
  - CNA-seq

- **140 microarray tools for**
  - gene expression
  - miRNA expression
  - protein expression
  - aCGH
  - SNP
  - integration of different data

- **60 tools for sequence analysis**
  - BLAST, EMBOSS, MAFFT
  - Phylip

Visualizing the data

- **Data visualization panel**
  - Maximize and redraw for better viewing
  - Detach = open in a separate window, allows you to view several images at the same time

- **Two types of visualizations**

  1. **Interactive visualizations produced by the client program**
     - Select the visualization method from the pulldown menu
     - Save by right clicking on the image

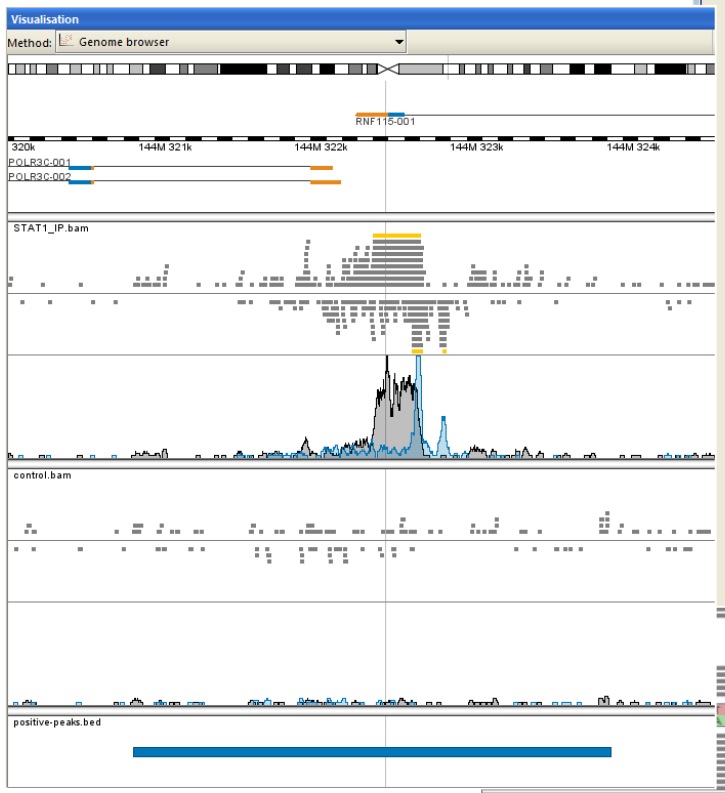
  2. **Static images produced by analysis tools**
     - Select from Analysis tools / Visualisation
     - View by double clicking on the image file
     - Save by right clicking on the file name and choosing "Export"

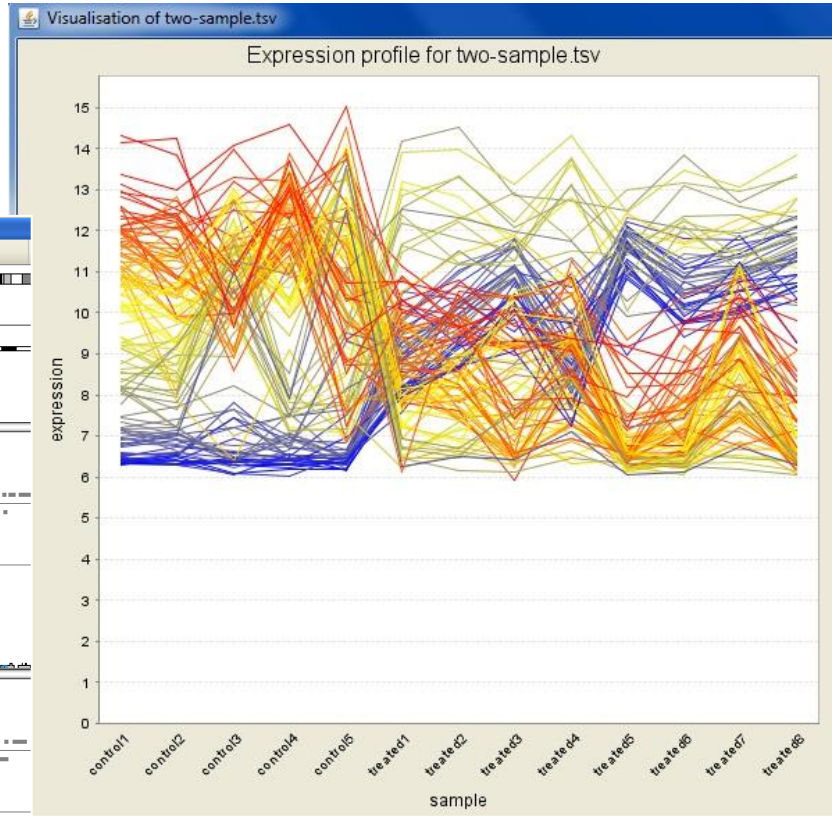
Interactive visualizations by the client

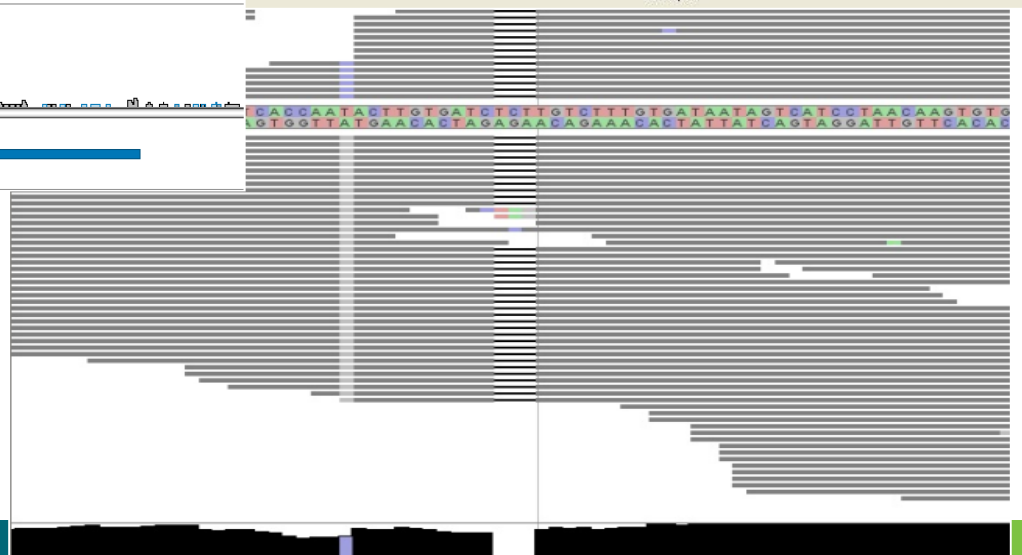
- Genome browser
- Spreadsheet
- Histogram
- Venn diagram
- Scatterplot
- 3D scatterplot
- Volcano plot
- Expression profiles
- Clustered profiles
- Hierarchical clustering
- SOM clustering

Available actions:

- Select genes and create a gene list
- Change titles, colors etc
- Zoom in/out

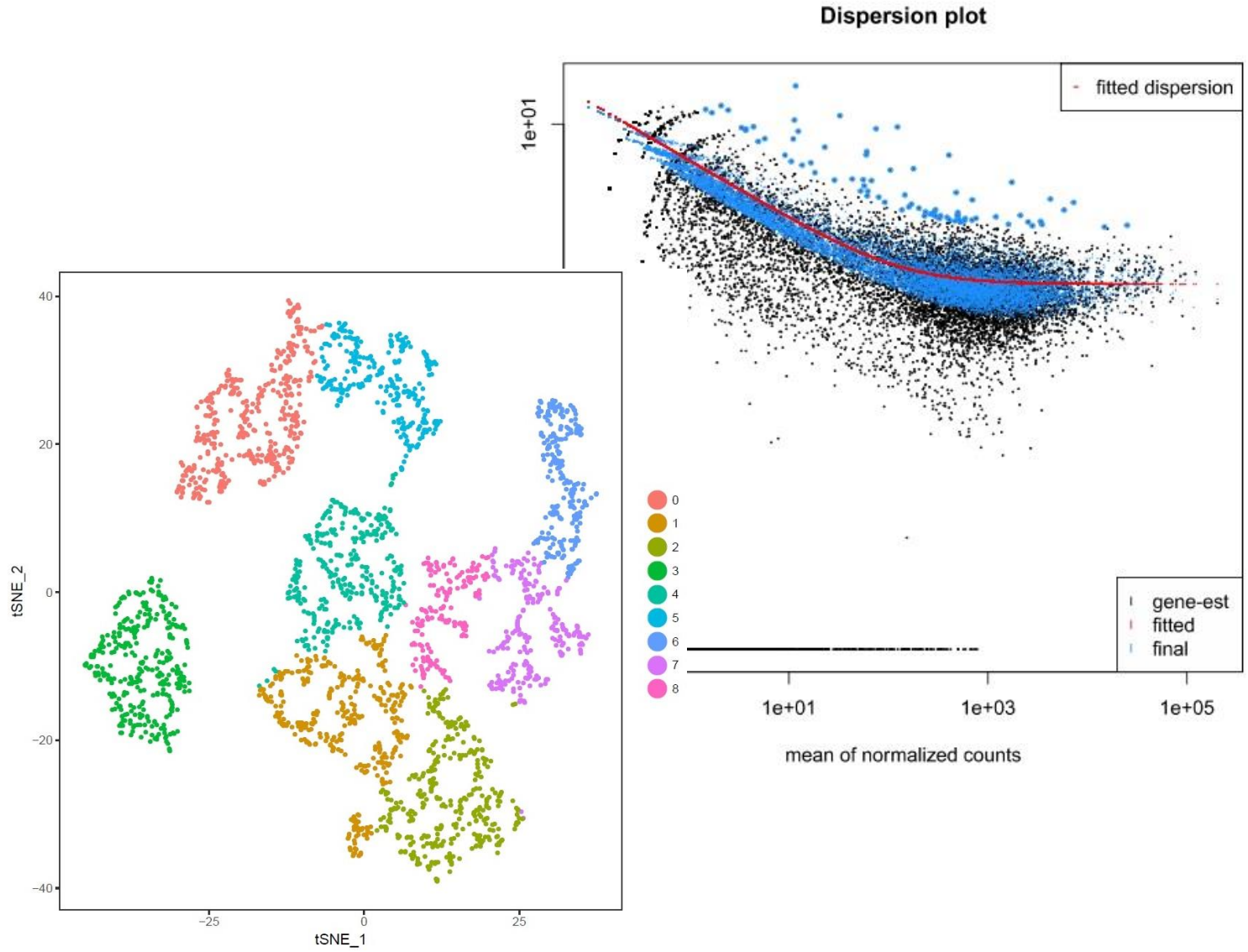






Static images produced by R/Bioconductor

- Dispersion plot
- Heatmap
- tSNE plot
- Violin plot
- PCA plot
- MA plot
- MDS plot
- Box plot
- Histogram
- Dendrogram
- K-means clustering
- Etc…



Options for importing data to Chipster

- **Import files/ Import folder**
- **Import from URL**
  - Utilities / Download file from URL directly to server
- **Import from BaseSpace coming soon**
- **Open an analysis session**
  - Files / Open session
- **Import from SRA database**
  - Utilities / Retrieve FASTQ or BAM files from SRA
- **Import from Ensembl database**
  - Utilities / Retrieve data for a given organism in Ensembl
- **What kind of RNA-seq data files can I use in Chipster?**
  - Compressed files (.gz) and tar packages (.tar) are ok
  - FASTQ, BAM, read count files (.tsv), GTF

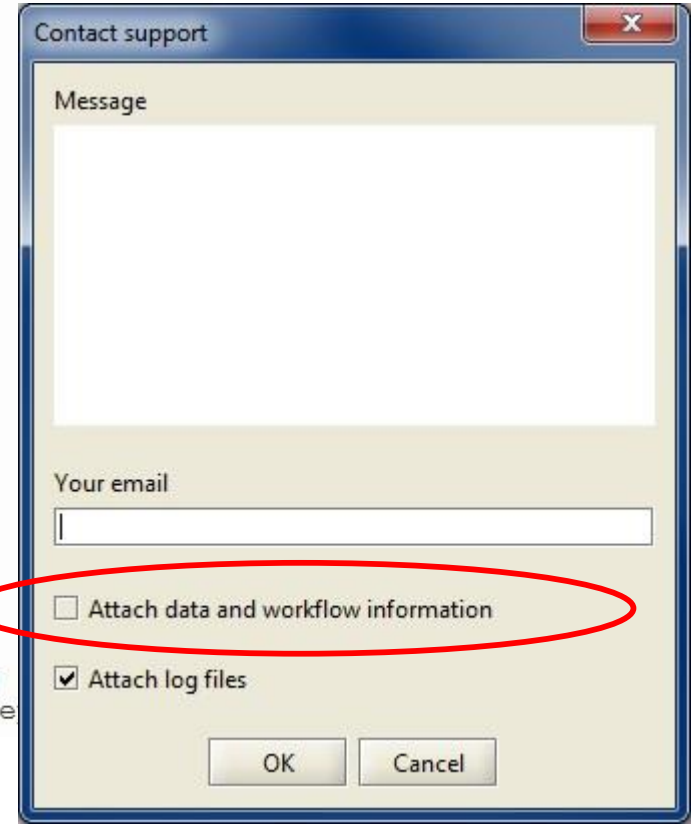
How to import a large tar package of files from a server and use only some of the files?

- Import the tar package by selecting **File / Import from / URL directly to server**
- Check what files it contains using the tool **Utilities / List contents of a tar file**
- Selectively extract the files you want with **Utilities / Extract .tar or .tar.gz file**

Problems? Send us a support request

-request includes the error message and link to analysis session (optional)

```
Hi,
I'm trying to normalise my Illumina microarray data (obtained with the Illumina HT-12 v4.0)
For that purpose I have selected the Normalisation option "Illumina - lumi pipeline"
However, the normalisation did not complete successfully.

Any advice to solve this problem ?

Thank you in advance for your precious help.

Best regards

Error message:
in library(chiptype, character.only = T) :
  there is no package called 'Illumina.db'

------------------------------------------------------------------

> chipster.common.path = '/opt/chipster/comp/modules/common/R-2.12'
> chipster.module.path = '/opt/chipster/comp/modules/microarray'
> setwd("271661a6-946c-450f-bb21-5d5b5a2837aa")
> probe.identifier <- "Probe_ID"
> transformation <- "log2"
> background.correction <- "none"
> normalize.chips <- "quantile"
> chiptype <- "empty"
> # TOOL norm-illumina-lumi.R: "Illumina - lumi pipeline" (Illumina normalization using
BeadSummaryData files, and using lumi methodology. If you have a BeadSummaryData that re
```



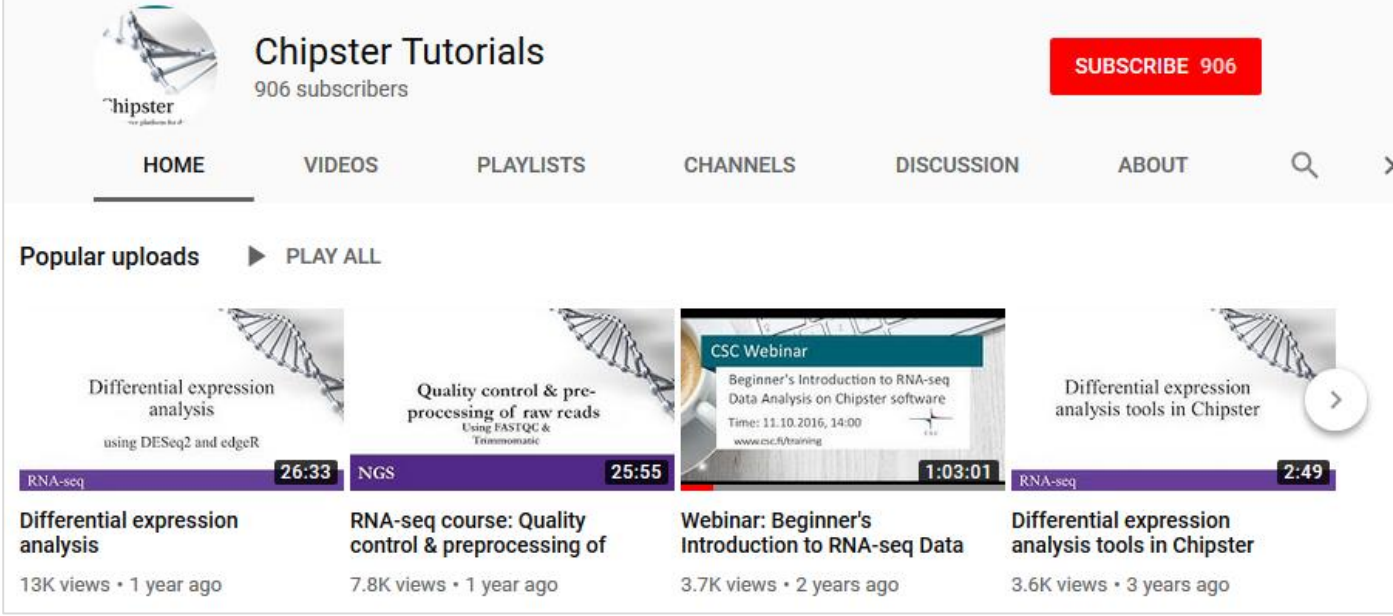
How to get a user account?

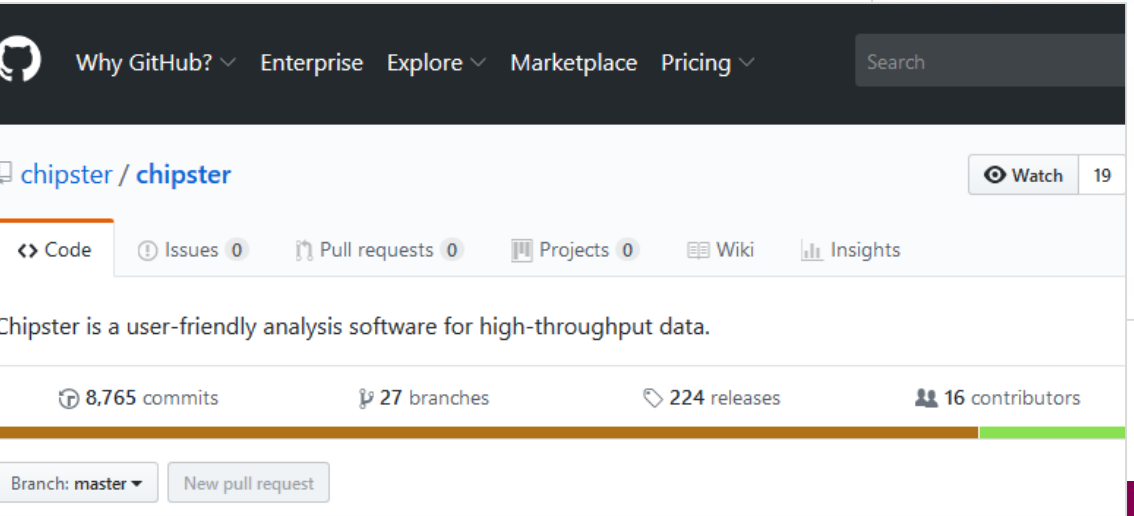
- Finnish academics:
  - use Scientist's User Interface service (https://sui.csc.fi) with HAKA credentials
    - click on the purple HAKA link
    - log in with your HAKA username and password
    - fill in the sign up form as shown in https://research.csc.fi/csc-guide-getting-access-to-csc-services#1.2.1
  - If you don't have HAKA credentials, email servicedesk@csc.fi
  - Users who already have a regular CSC username and password can use those for Chipster

- Others:
  - Set up a local Chipster server at your institute (free of charge)
  - Buy access to CSC's Chipster server (500 euros / person /year)
  - Use EGI's Chipster server (free of charge)
- See https://chipster.csc.fi/access.shtml for details

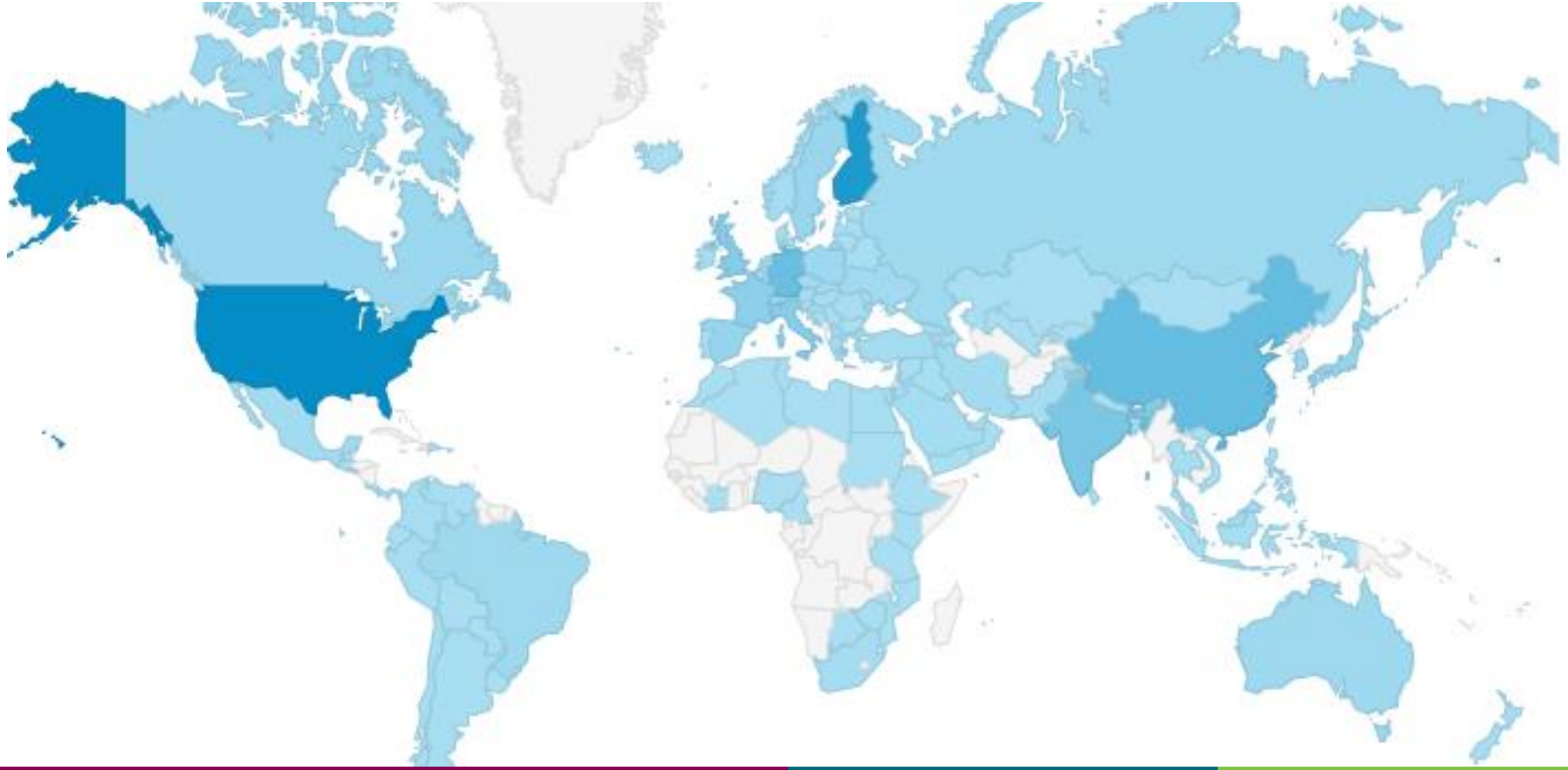
More info

- chipster@csc.fi
- http://chipster.csc.fi
- Chipster tutorials in YouTube
- https://chipster.csc.fi/manual/courses.html





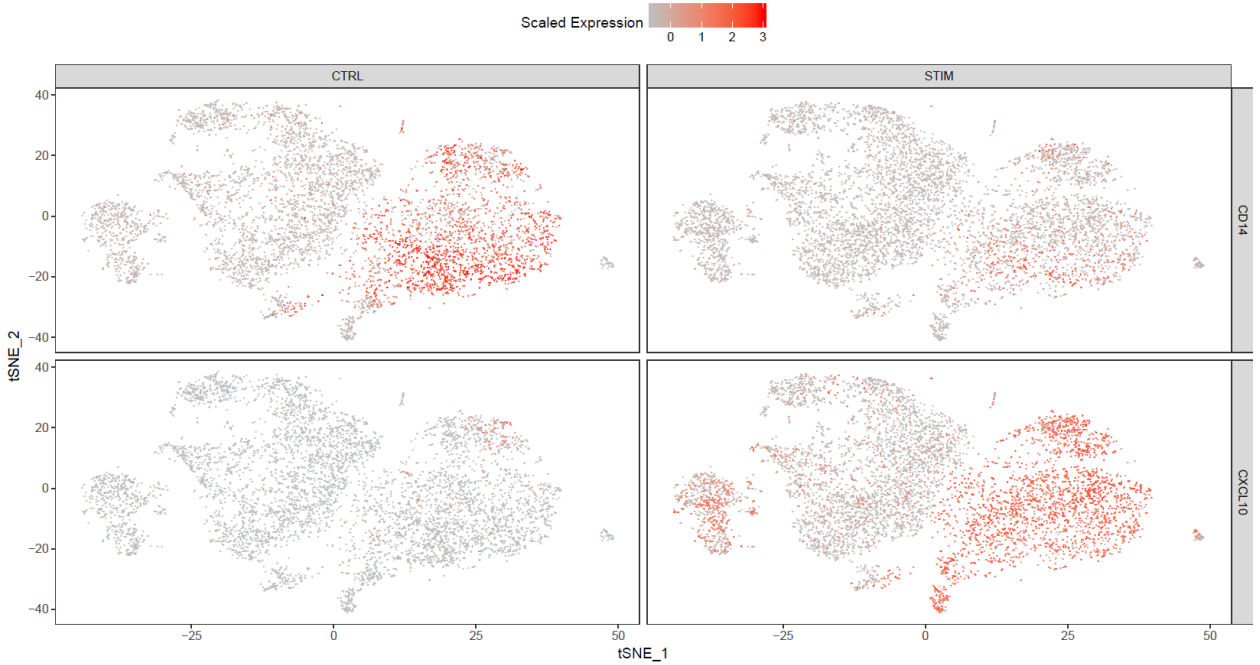
Acknowledgements to Chipster users and contibutors



Introduction to single cell RNA-seq

Single cell RNA-seq

- New technology, data analysis methods are actively developed
- Measures distribution of expression levels for each gene across a population of cells
- Allows to study cell-specific changes in transcriptome
- Applications
  - cell sub-populations within a biological condition
  - cell type specific differential expression
  - transcriptional regulatory networks
  - dynamic processes using pseudotime ordering
    - Switches
    - Branch points



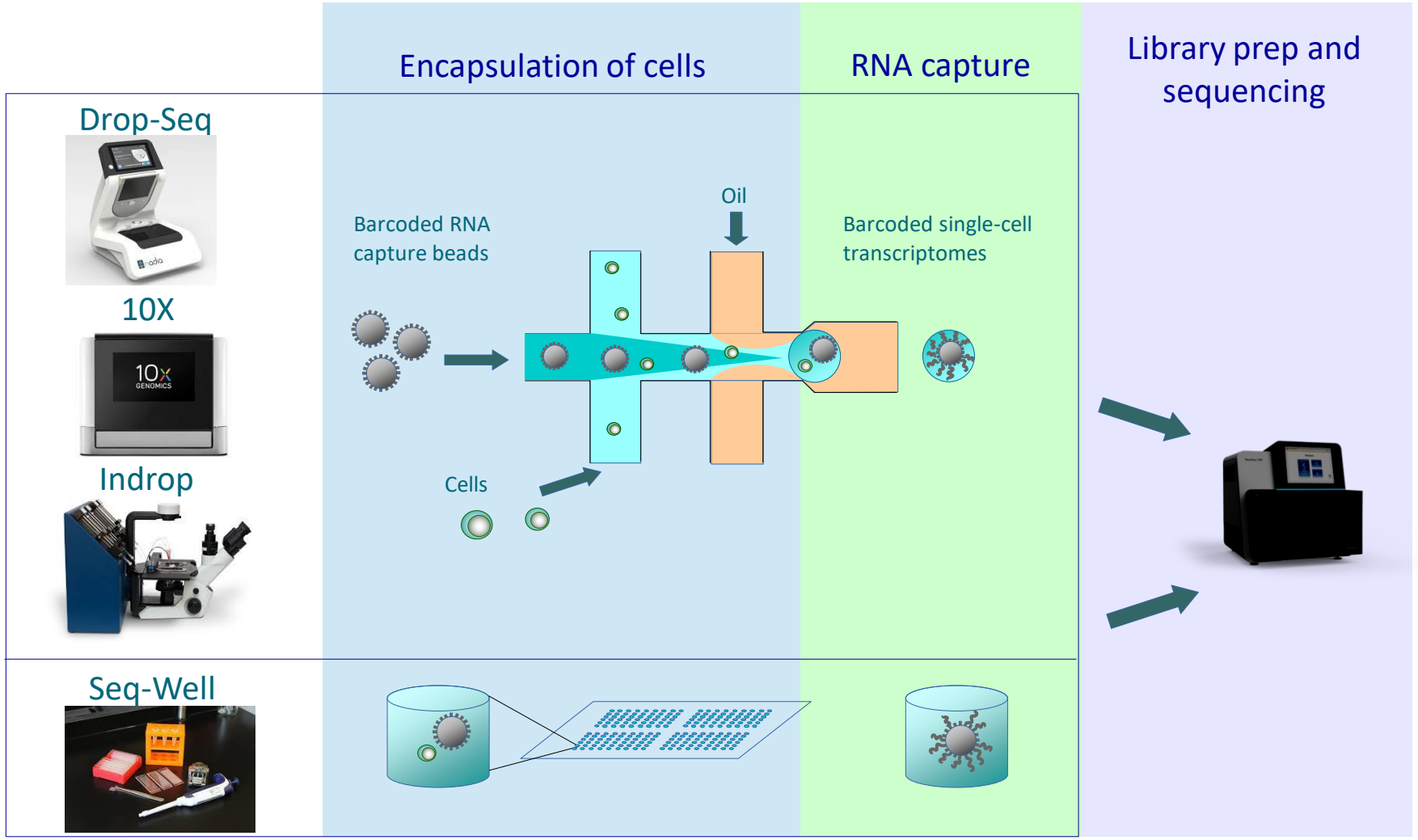
Different technologies for capturing single-cell transcriptomes

- Plate-based: SMART-Seq2, STRT-Seq, CEL-Seq
- Droplet-based: **10X Chromium**, **Drop-Seq**, Indrop, Fluidigm C1
- Microwell-based: ICell8, Seq-Well

- Libraries are usually 3’ tagged: only a short sequence at the 3’ end of the mRNA is sequenced

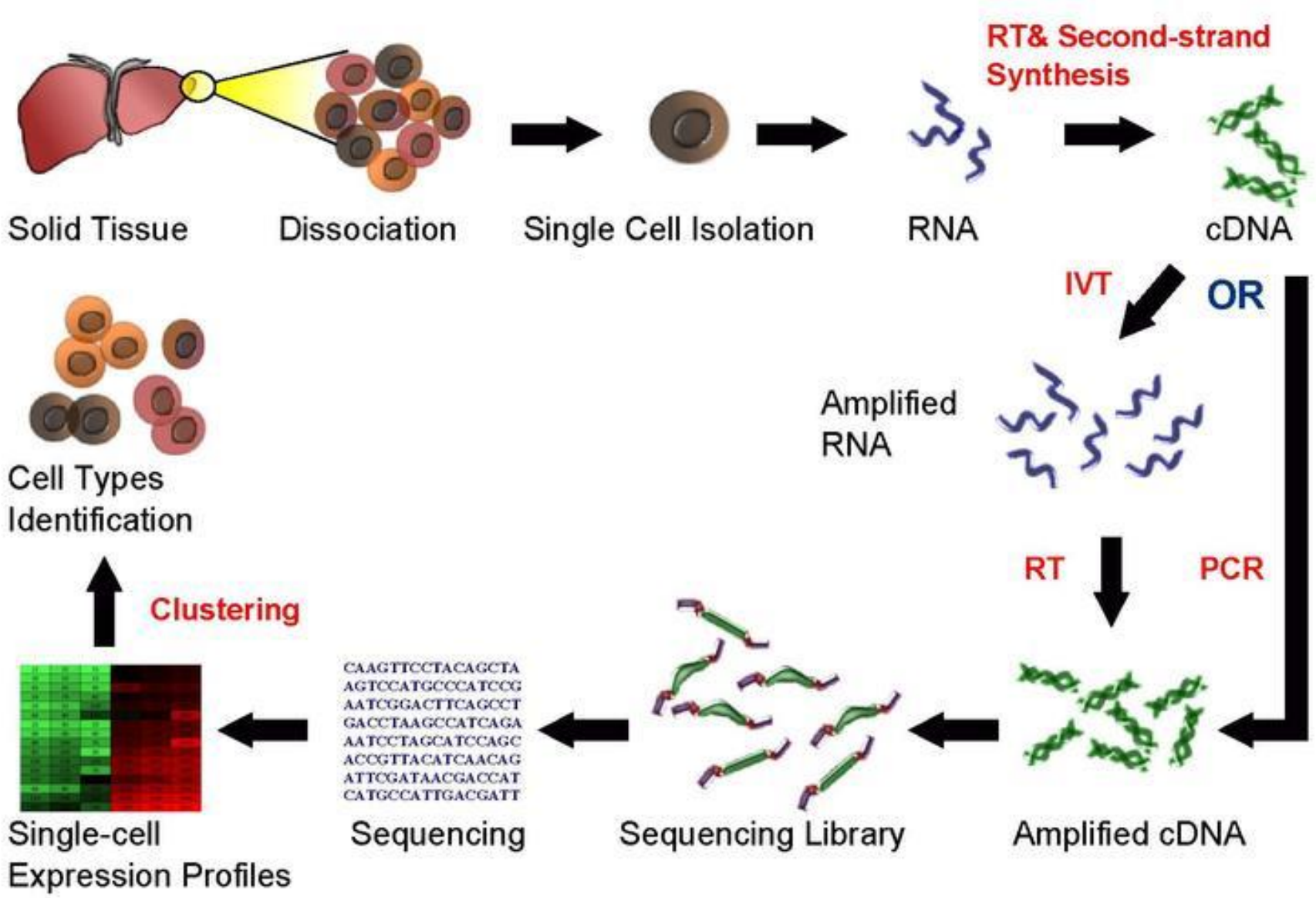
Slide by Heli Pessa

Different technologies for capturing single-cell transcriptomes

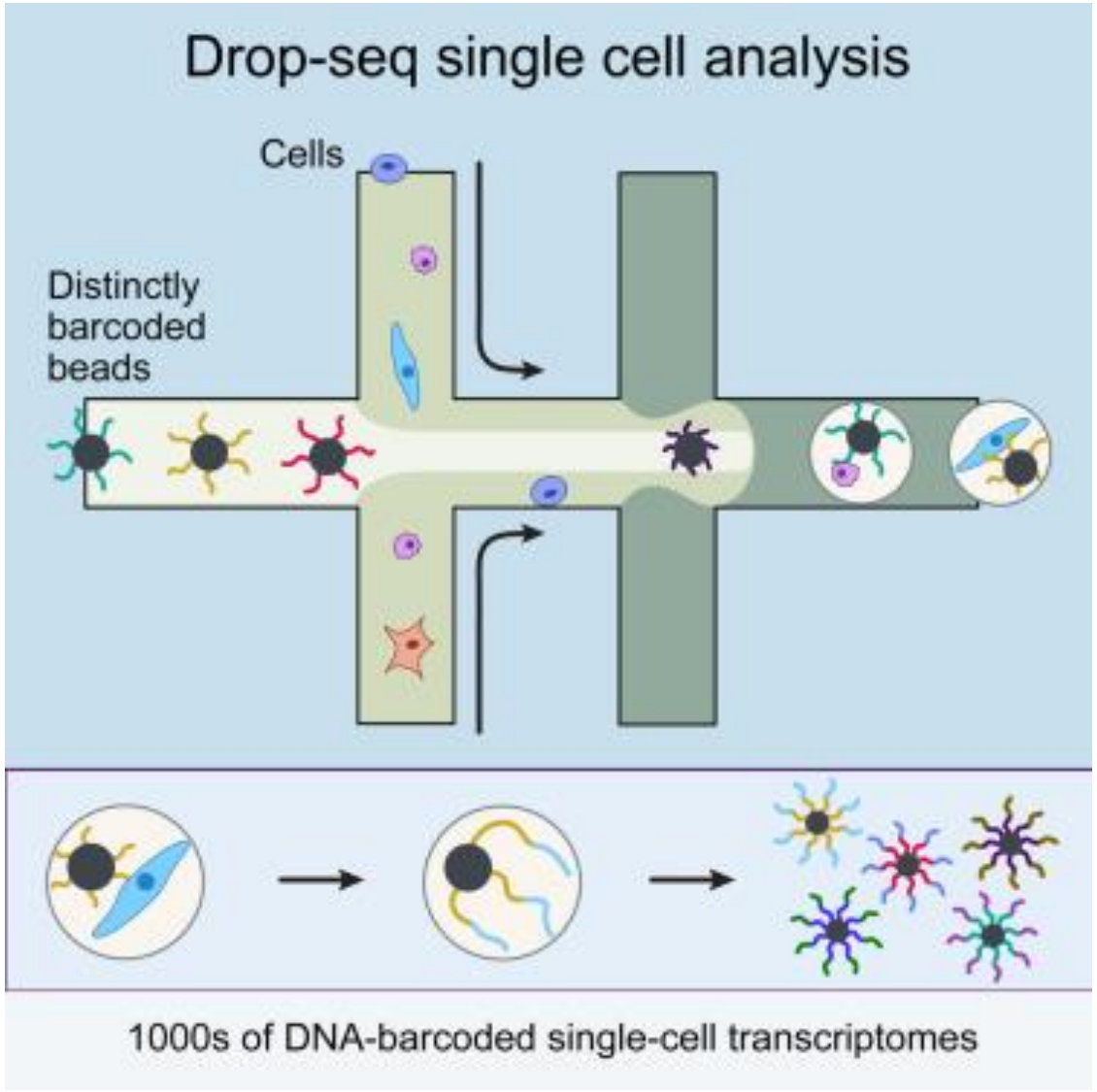


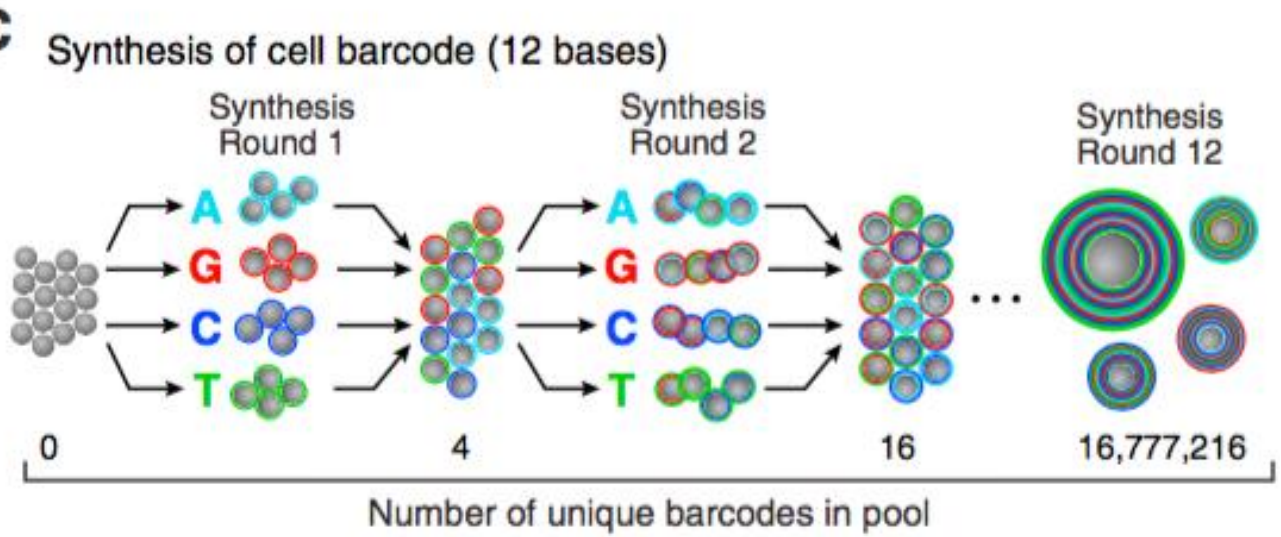
Slide by Heli Pessa

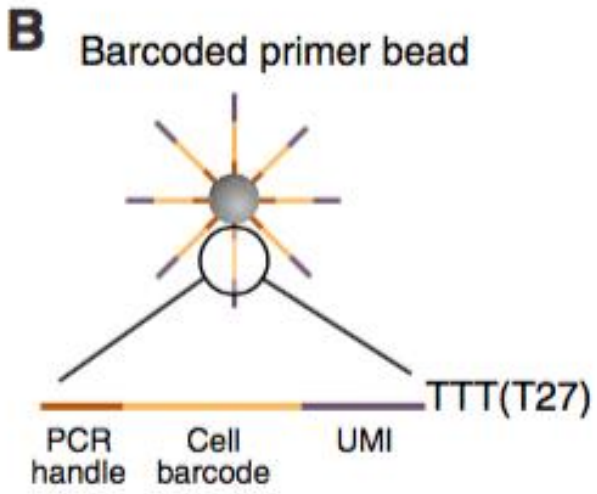
Single Cell RNA Sequencing Workflow

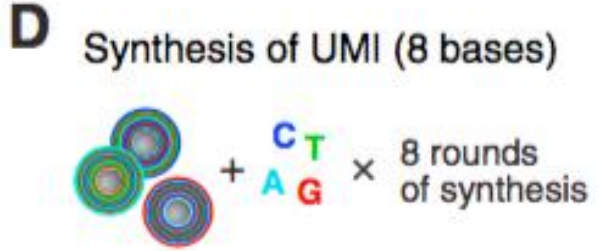


https://hemberg-lab.github.io/scRNA.seq.course/introduction-to-single-cell-rna-seq.html



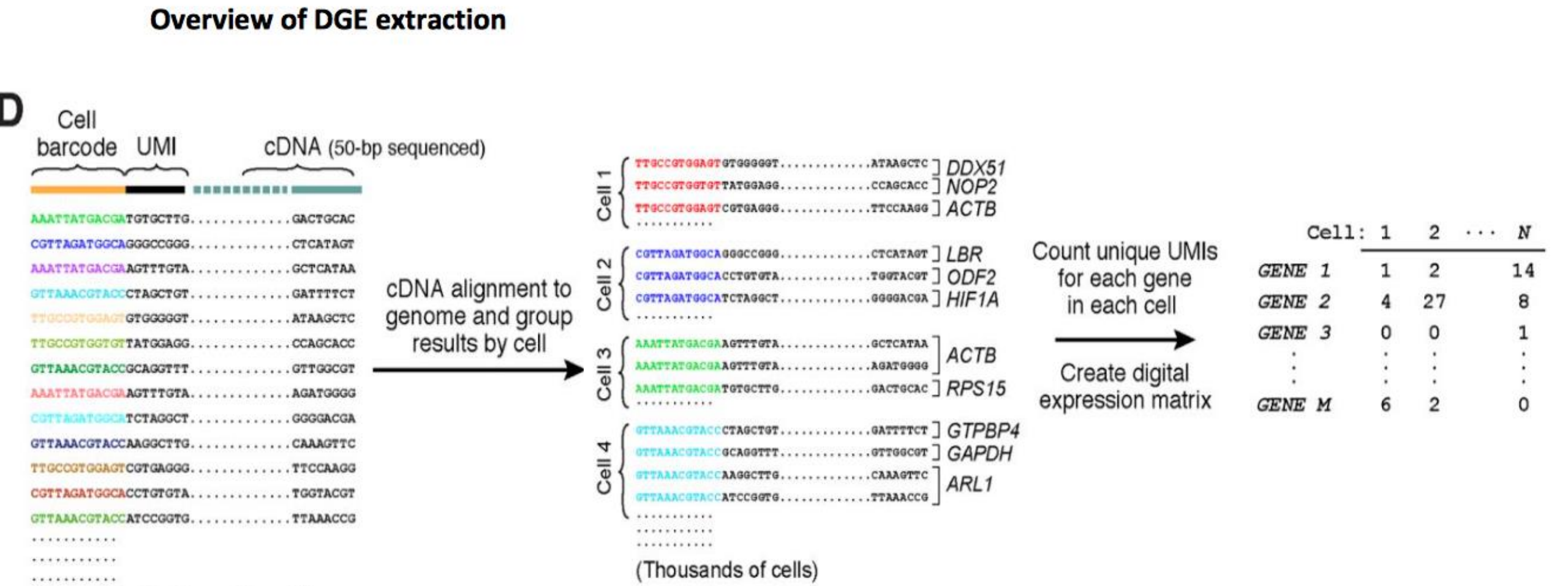






- Millions of the same **cell barcode** per bead
- 4^8 different **molecular barcodes** (UMIs) per bead

DropSeq data preprocessing overview



What can go wrong in Drop-seq?

1. Ideally there is one healthy cell in the droplet so that we get a "STAMP" (single-cell transcriptome attached to a microparticle). However, sometimes
   - There is no cell in the droplet, just ambient RNA

     → Detect "empties" based on the small number of genes expressed and remove
   - There are two (or more) cells in a droplet

     → Detect duplets (and multiplets) based on the large number of genes expressed and remove
   - The cell in the droplet is broken/dead

     → Detect based on high proportion of reads mapping to mitochondrial genome and remove
2. Sometimes barcodes have synthesis errors in them, e.g. one base is missing

   → Detect by checking the distribution of bases at each position and fix the code or remove the cell

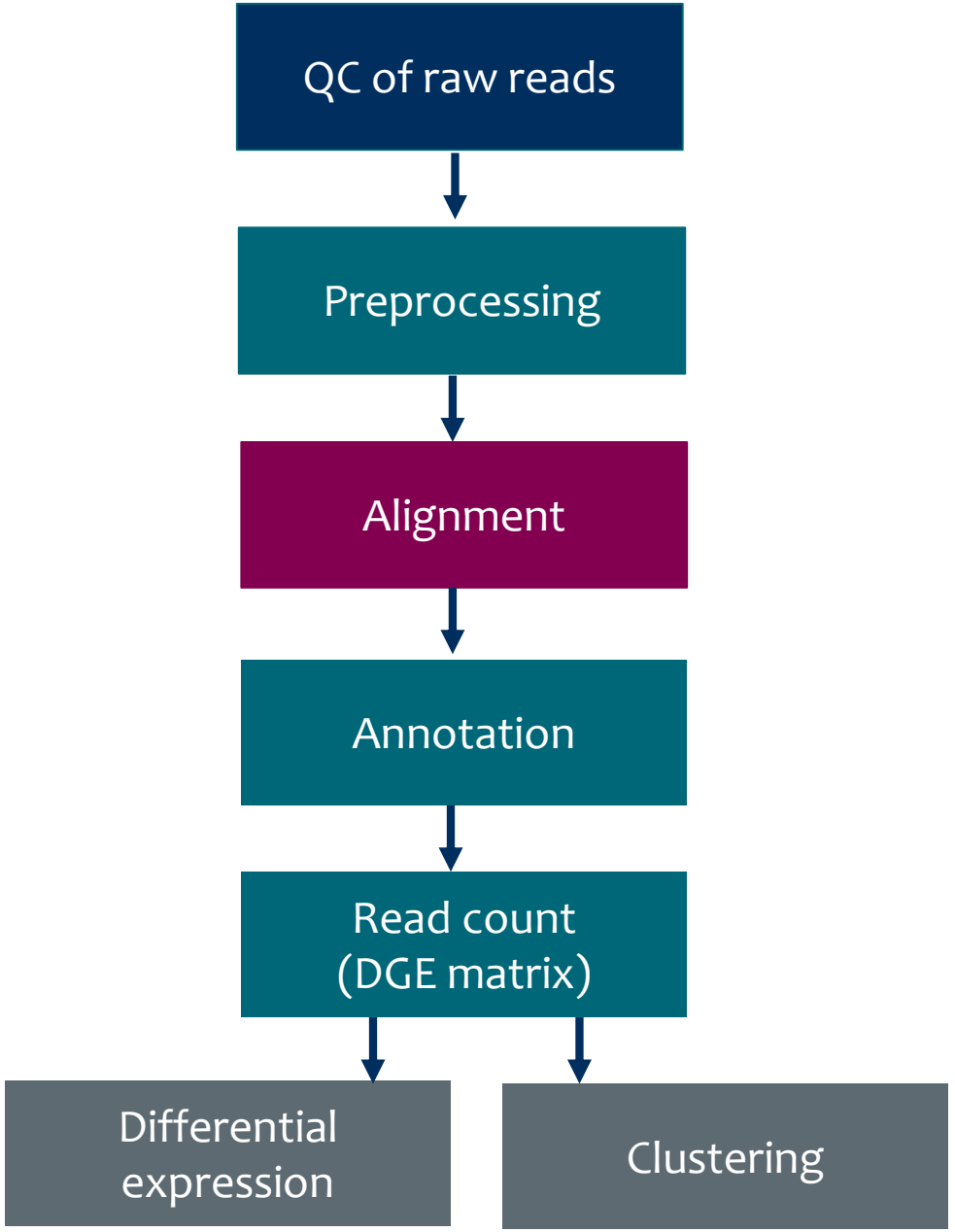
Single cell RNA-seq data analysis

Single cell RNA-seq data is challenging

- The detected expression level for many genes is zero
- Data is noisy. High level of variation due to
  - Capture efficiency (percentage of mRNAs captured)
  - Amplification bias (non-uniform amplification of transcripts)
  - Cells differ in terms of cell-cycle stage and size
- Complex distribution of expression values
  - Cell heterogeneity and the abundance of zeros give rise to multimodal distributions

→ Many methods used for bulk RNA-seq data won't work

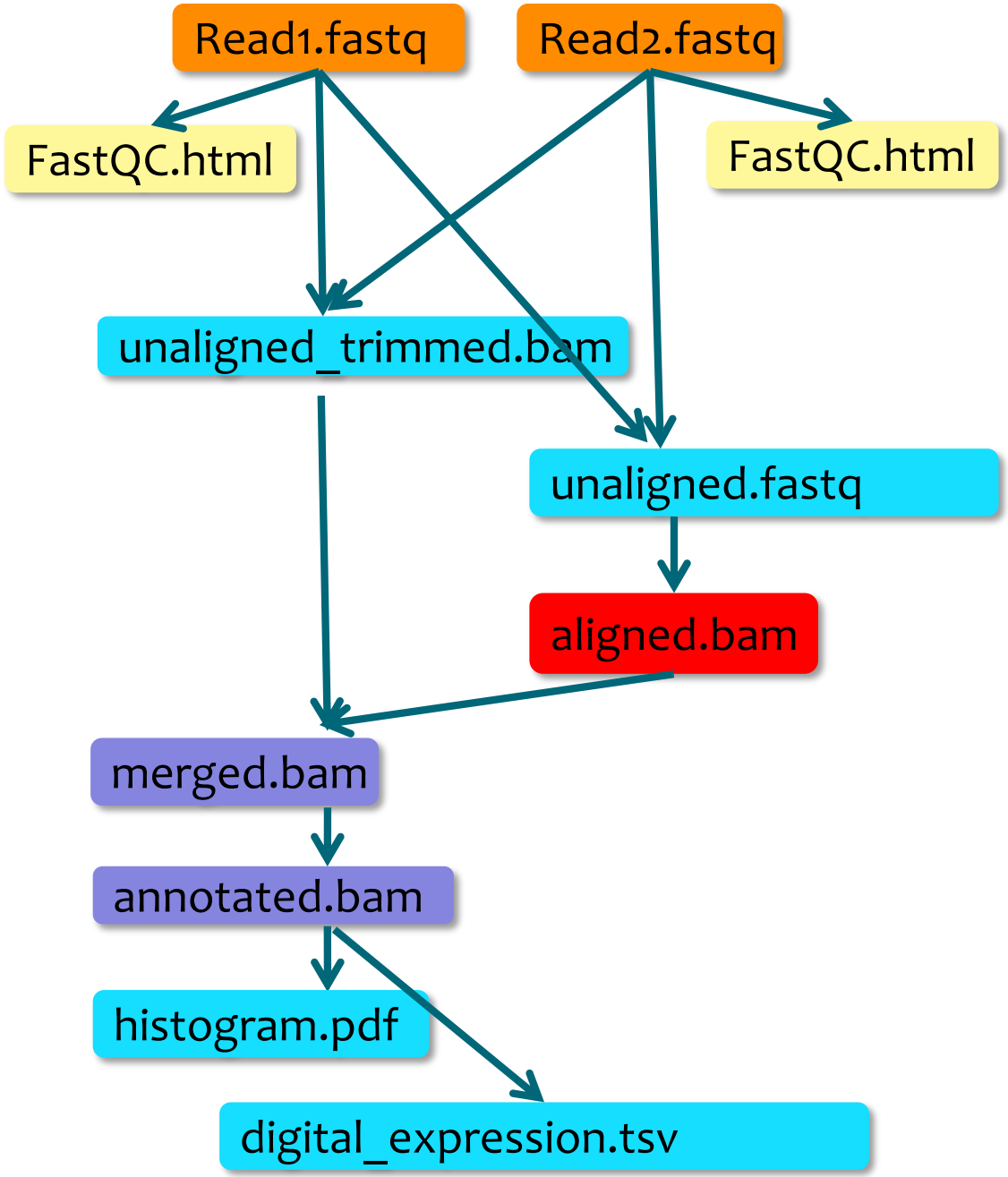
scRNA-seq data analysis steps



DropSeq data preprocessing

From FASTQ to expression matrix –preprocessing of DropSeq data

- Quality control of raw reads
- Preprocessing: tagging with barcodes & filtering
- Alignment to reference genome
- Merge BAM files
- Annotate with gene names
- Estimate the number of usable cells
- Detect bead synthesis errors
- Generate digital expression matrix



Import files

Quality control / FASTQC

Preprocess:

- FASTQ to BAM
- Tag BAM
- Filter and trim BAM
- BAM to FASTQ
- Filter too short reads

-filter bad quality barcodes
-trim adapter sequences
-trim polyA tails

Aligner (STAR, HISAT)

Merge BAM

Combines the barcode tags and the alignment info

Tag reads with gene names

Estimate number of usable cells

Create digital expression matrix

+ Detect bead synthesis errors

From FASTQ to expression matrix –preprocessing of DropSeq data

- Quality control of raw reads
- Preprocessing: tagging with barcodes & filtering
- Alignment to reference genome
- Merge BAM files
- Annotate with gene names
- Estimate the number of usable cells
- Detect bead synthesis errors
- Generate digital expression matrix

What and why?

Potential problems

- low confidence bases, Ns
- sequence specific bias, GC bias
- adapters
- sequence contamination
- unexpected length of reads
- …

Knowing about potential problems in your data allows you to

- correct for them before you spend a lot of time on analysis
- take them into account when interpreting results

Raw reads: FASTQ file format

- Four lines per read:

  @read name
  GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT
  + read name
  !''*((((***+))%%%++)(%%%%).1***-+*''))**55CCF>>>>>>CCCCCCC65

- http://en.wikipedia.org/wiki/FASTQ_format

- Attention: **Do not unzip FASTQ files!**

  - Chipster's analysis tools can cope with zipped files (.gz)

Base qualities

- If the quality of a base is 20, the probability that it is wrong is 0.01.
  - **Phred quality score** Q = -10 * $\log_{10}$ (probability that the base is wrong)

```
T   C   A   G   T   A   C   T   C   G

40  40  40  40  40  40  40  40  37  35
```

Probability:	0.0001
Phred score:	40
ASCII coding:	I (capital i)

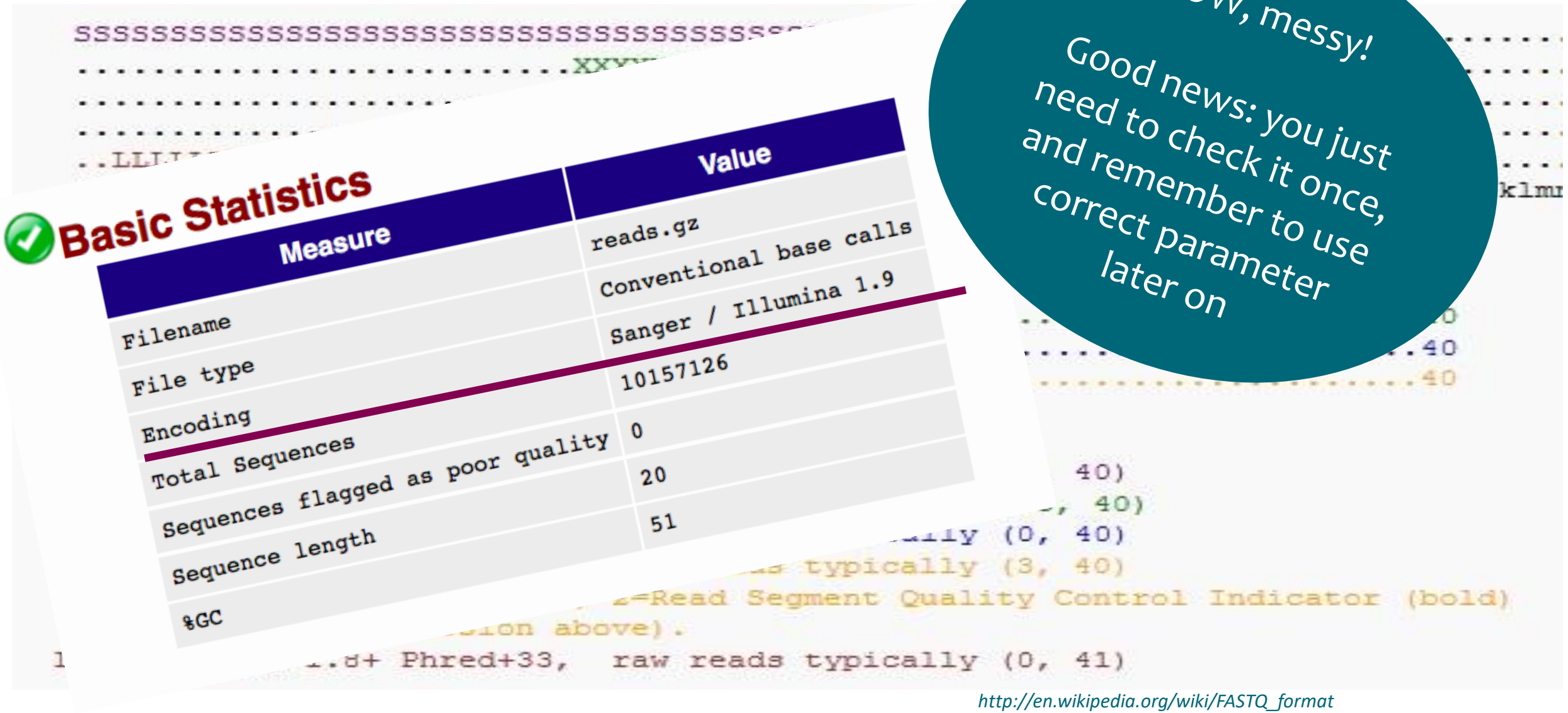
- "Sanger" encoding: numbers are shown as ASCII characters so that 33 is added to the Phred score
  - E.g. 39 is encoded as "H", the 72nd ASCII character (39+33 = 72)
  - Note that older Illumina data uses different encoding
    - Illumina1.3: add 64 to Phred
    - Illumina 1.5-1.7: add 64 to Phred, ASCII 66 "B" means that the whole read segment has low quality

Base quality encoding systems

```
  SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS.....................................
  !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmr
  |                          |    |        |                              |
 33                         59   64       73                            104
  0........................26...31.......40

S - Sanger        Phred+33,  raw reads typically (0, 40)
```

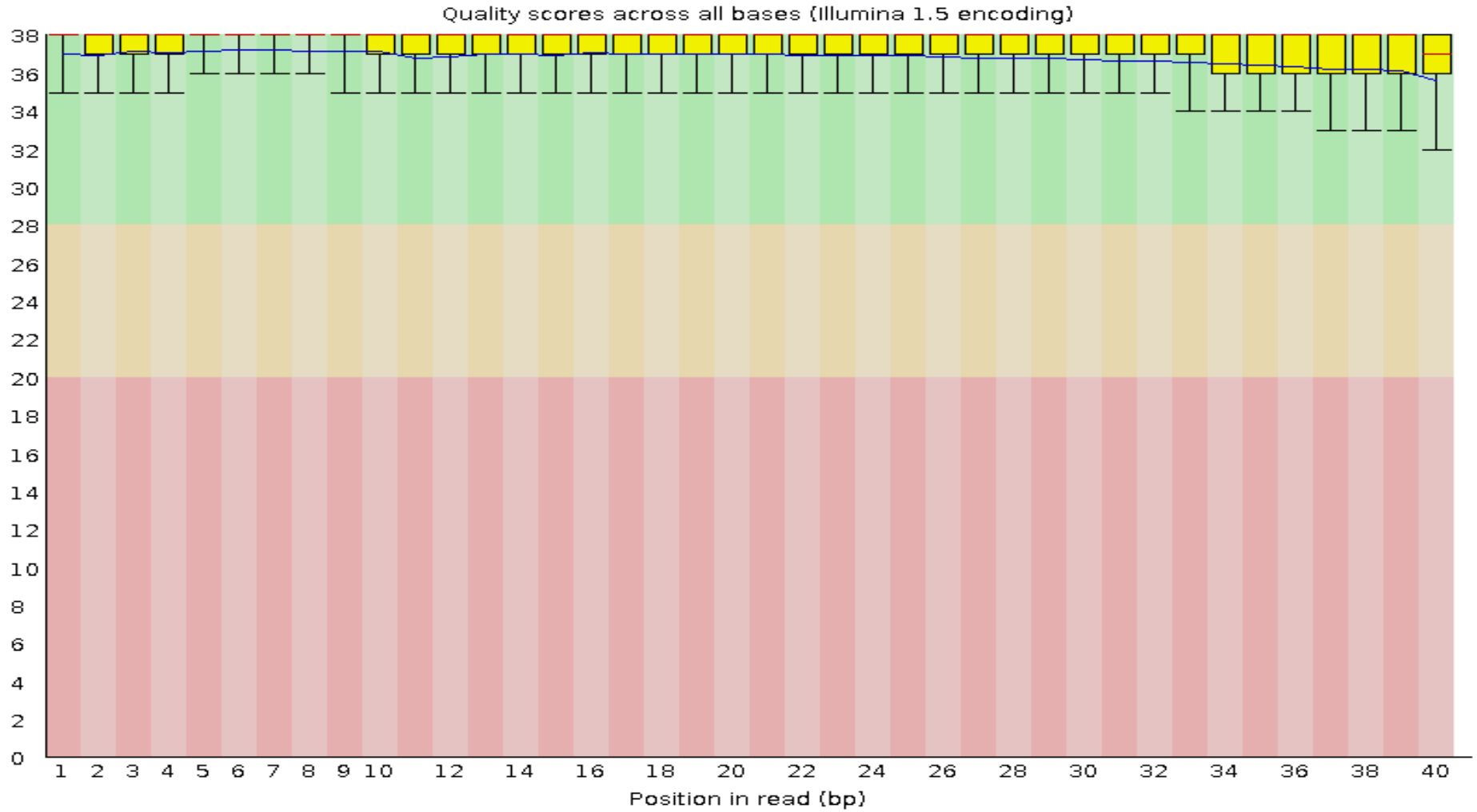
http://en.wikipedia.org/wiki/FASTQ_format

Base quality encoding systems

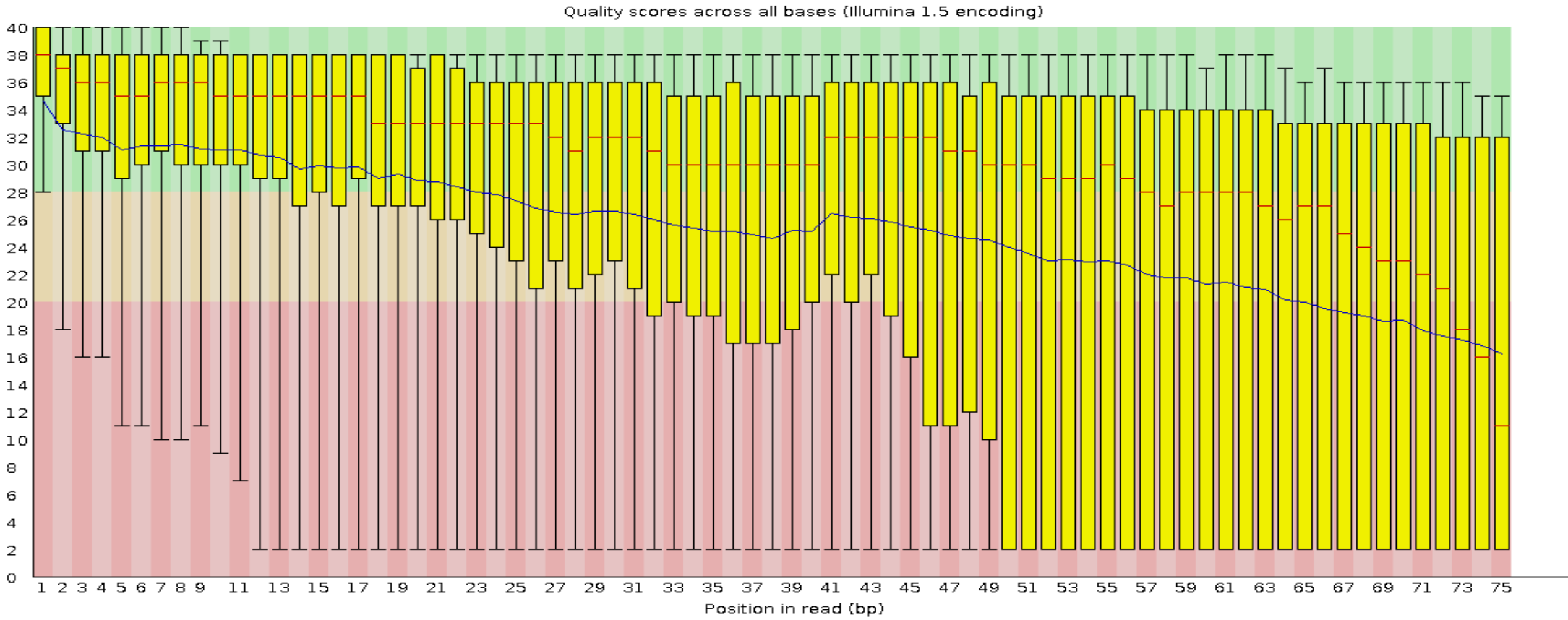


http://en.wikipedia.org/wiki/FASTQ_format

Per position base quality (FastQC)

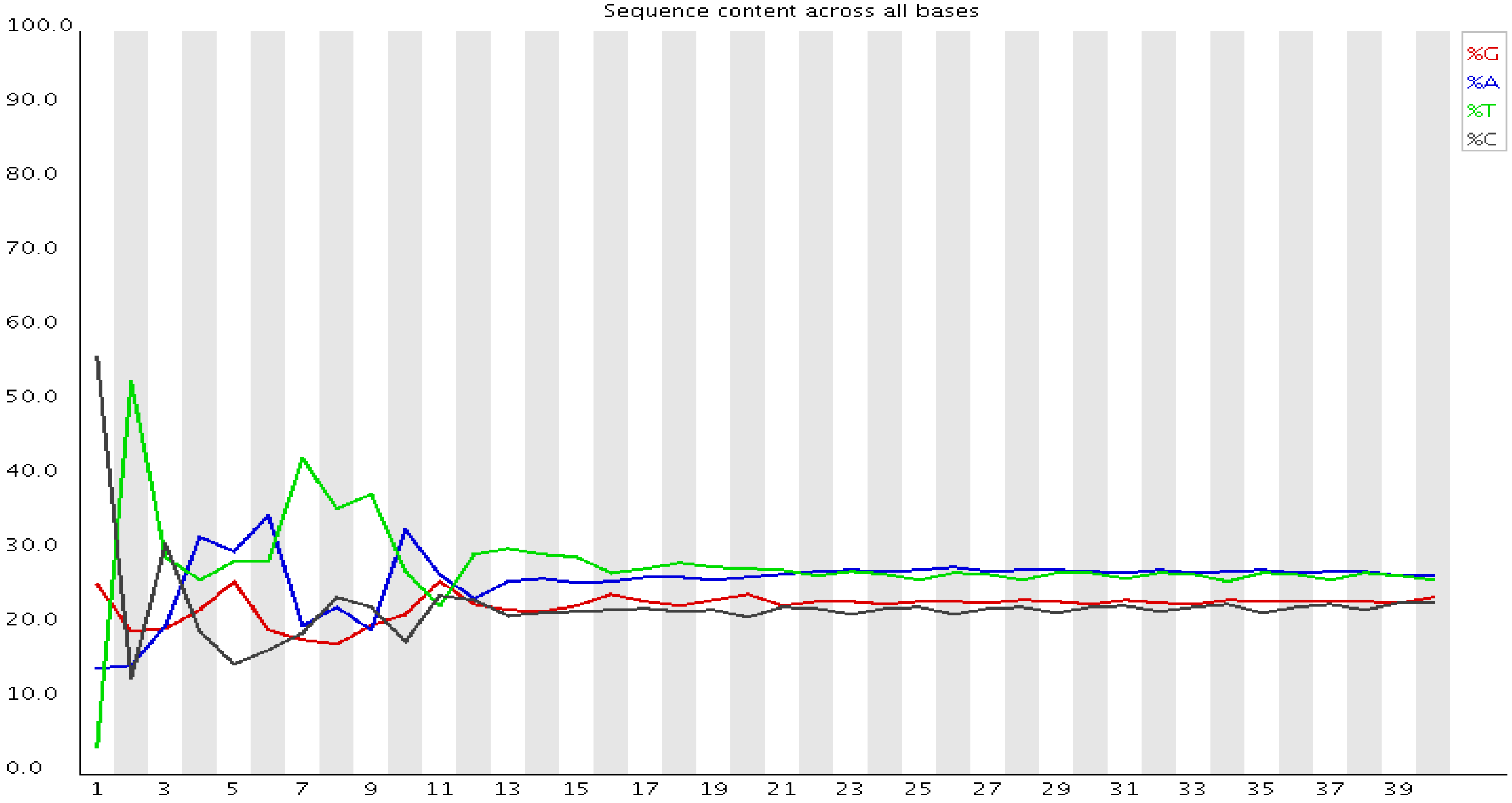


Per position base quality (FastQC)



Per position sequence content (FastQC)

Per base sequence content



Overrepresented sequences

Sequence	Count	Percentage	Possible Source
GGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGG	32852	0.49846796813762156	No Hit

- Enrichment of k-mers at the 5′ end due to use of random hexamers or transposases in the library preparation
  - Typical for RNA-seq data
  - Can't be corrected, doesn't usually effect the analysis
- More A's : polyA tails
  - These we remove
- Over-represented sequences with lots of G's: NextSeq sequencer does this

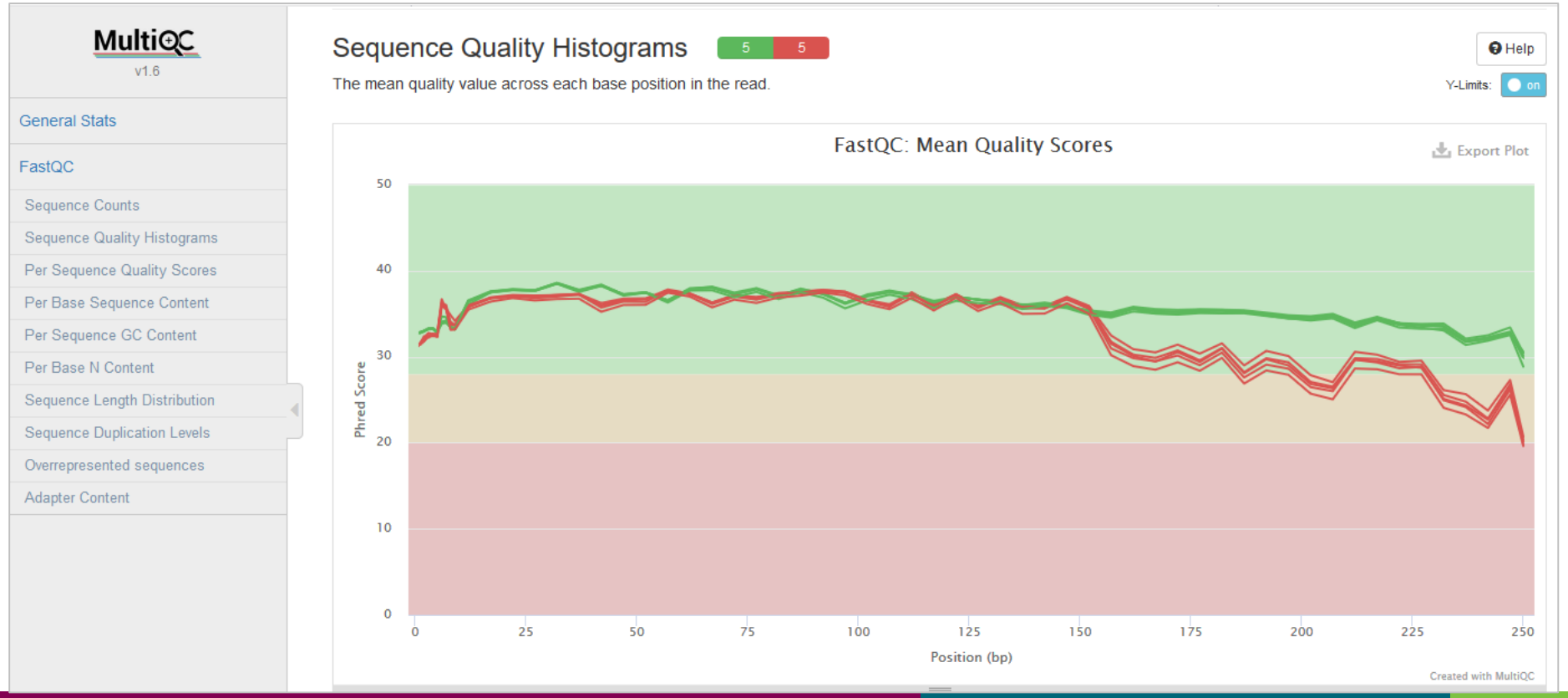
Check also the length of the reads

Measure	Value
Filename	reads.gz
File type	Conventional base calls
Encoding	Sanger / Illumina 1.9
Total Sequences	10157126
Sequences flagged as poor quality	0
Sequence length	20
%GC	51

- Barcode read: should be 20 bp
- Other length → BAM tagging won't work
- To **filter** shorter barcode reads:
  - select **BOTH FASTQ files**
  - tool **Preprocessing / Trim reads with Trimmomatic** with parameter **Minimum length of reads to keep = 20**
  - Continue analysis with the files that are named "--trimmed.fq.gz" (instead of the "--unpaired_trimmed.fq.gz" ones).
- To **trim** too long barcode reads:
  - after the previous step: select **read1** file, run **Preprocessing / Trim reads with Trimmomatic** with parameter **Number of bases to keep from the start = 20**

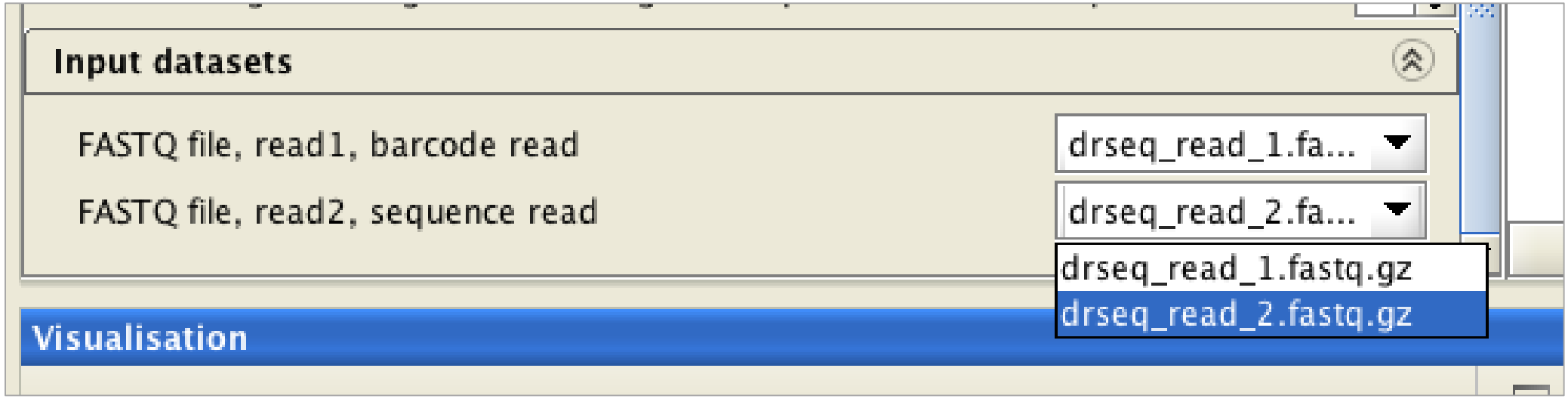
I have many FASTQ files – how can I quickly check them all?

- Make a tar package of all the FASTQ files using the tool Utilities / Make a tar package
- Select the tar package and run the tool Quality control / Read quality with MultiQC for many FASTQ files



Multiple input files: make sure they are correctly assigned!

- When running tools, remember to…
  - Make sure the files are correctly assigned
  - Choose the right reference
  - Check especially **bolded** parameters

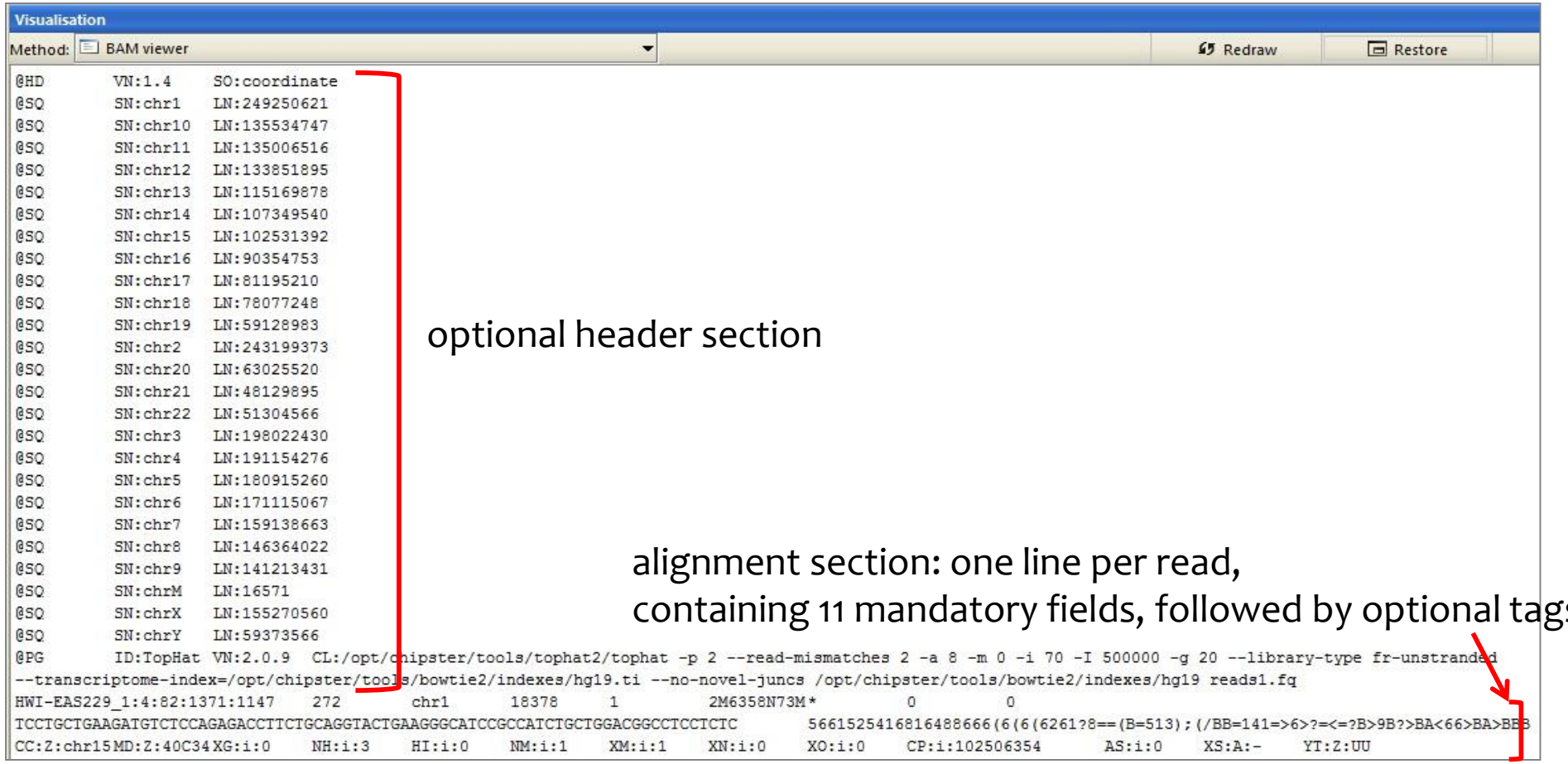


From FASTQ to expression matrix –preprocessing of DropSeq data

- Quality control of raw reads
- Preprocessing: tagging with barcodes & filtering
- Alignment to reference genome
- Merge BAM files
- Annotate with gene names
- Estimate the number of usable cells
- Detect bead synthesis errors
- Generate digital expression matrix

BAM file format

- BAM was developed for storing genomic positions of aligned reads, but it can be used for storing unaligned reads too
- BAM is a binary file, SAM (Sequence Alignment/Map) is a tab-delimited text file.
- Different BAM from different tools (different use of tags, different quality coding…)



Fields in BAM file

- read name HWI-EAS229_1:2:40:1280:283
- flag 272
- reference name 1
- position 18506
- mapping quality 0
- CIGAR 17M6183N26M
- mate name *
- mate position 0
- insert size 0
- sequence AGGGCCGATCTTGGTGCCATCCAGGGGGCCTCTACAAGGATAA
- base qualities ECC@EEF@EB:EECFEECCCBEEEE;>5;2FBB@FBFEEFCF@
- tags MD:Z:75 NH:i:7 AS:i:-8 XS:A:-

```
SRR1853178.393696883 16        1         3020298   1         60M       *         0         0
CCTTTTCAGGTGTGTTAGGGTGCCCTGGACTGGGCGAAGTGGGTGTTCTGGGTTCTGATG  FFAAFFFFFFFFAF..FF.F.FAFFAFFFAFFFFFFFFF.FFF.F.FFFFFF)<FAAAAA  MD:Z:60
XG:i:0    NH:i:5    NM:i:0    XM:i:0    XN:i:0    XO:i:0    AS:i:0    ZS:i:0    YT:Z:UU
```

Preprocessing single cell DropSeq FASTQ files

- This tool is a combination of several tools:
  - DropSeq (TagBamWithReadSequenceExtended, FilterBam)
  - Picard (FASTQ to BAM, BAM to FASTQ)
  - Trimmomatic (MINLEN)
- The steps are:
  1. Convert the FASTQ files into an unaligned BAM file
  2. Tag the reads in the BAM file with the cell and molecular barcodes
  3. Filter and trim the reads in the BAM file
  4. Convert the tagged and trimmed BAM file back into a FASTQ file for the alignment
  5. Filter out too short reads of the FASTQ file

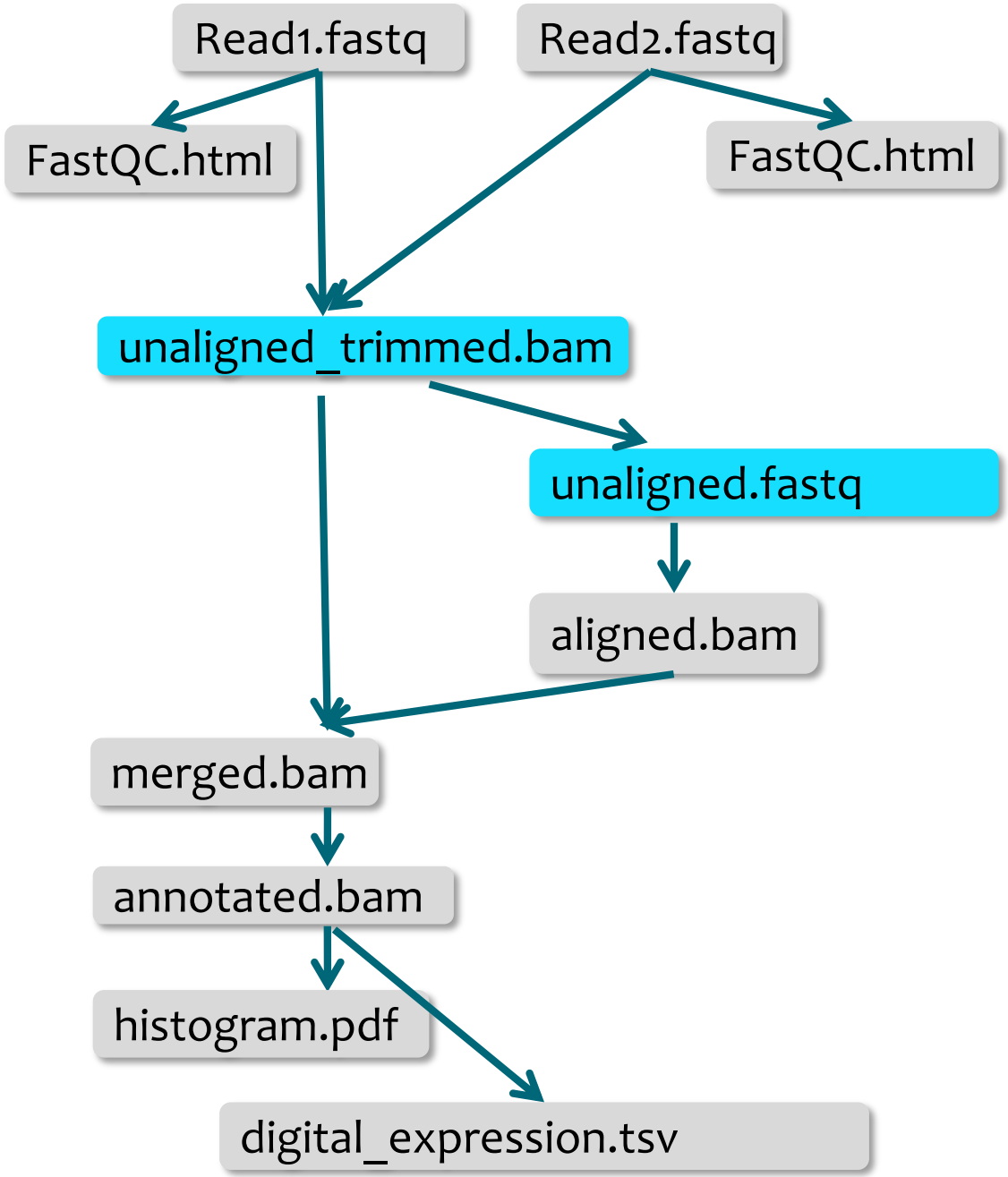
Why FASTQ-BAM-FASTQ-BAM?

- BAM format has tag fields where we can store cell and molecular barcodes.

- We trim and filter the reads in BAM format (would be possible in FASTQ too).

- However, the aligners take as input only FASTQ format, so we need to transform the trimmed & filtered BAM back to FASTQ.

- Alignment produces a BAM file, which we then merge with the unaligned BAM.



Import files

Quality control / FASTQC

Preprocess:

- FASTQ to BAM
- Tag BAM
- Filter and trim BAM
- BAM to FASTQ
- Filter too short reads

-filter bad quality barcodes
-trim adapter sequences
-trim polyA tails

Aligner (STAR, HISAT)

Merge BAM

Combines the barcode tags and the alignment info

Tag read with gene names

Estimate number of usable cells

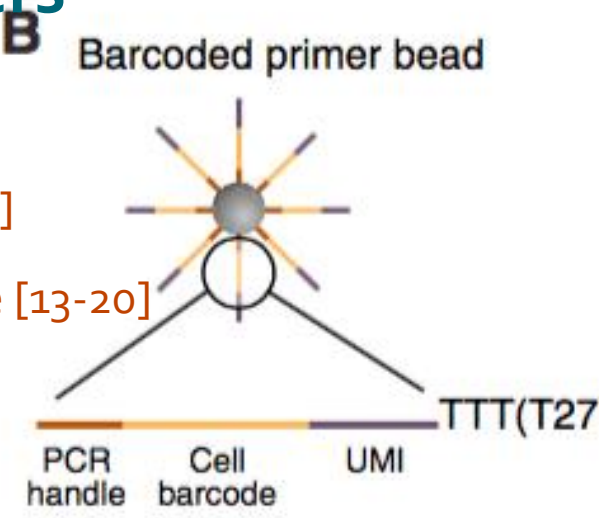
Create digital expression matrix

+ Detect bead synthesis errors

Preprocessing single cell DropSeq FASTQ files, parameters

- Convert the FASTQ files into an unaligned BAM file

- Tag the reads in the BAM file with the cellular and molecular barcodes
  - Base range for cell barcode [1-12]
  - Base range for molecule barcode [13-20]
  - Base quality [10]



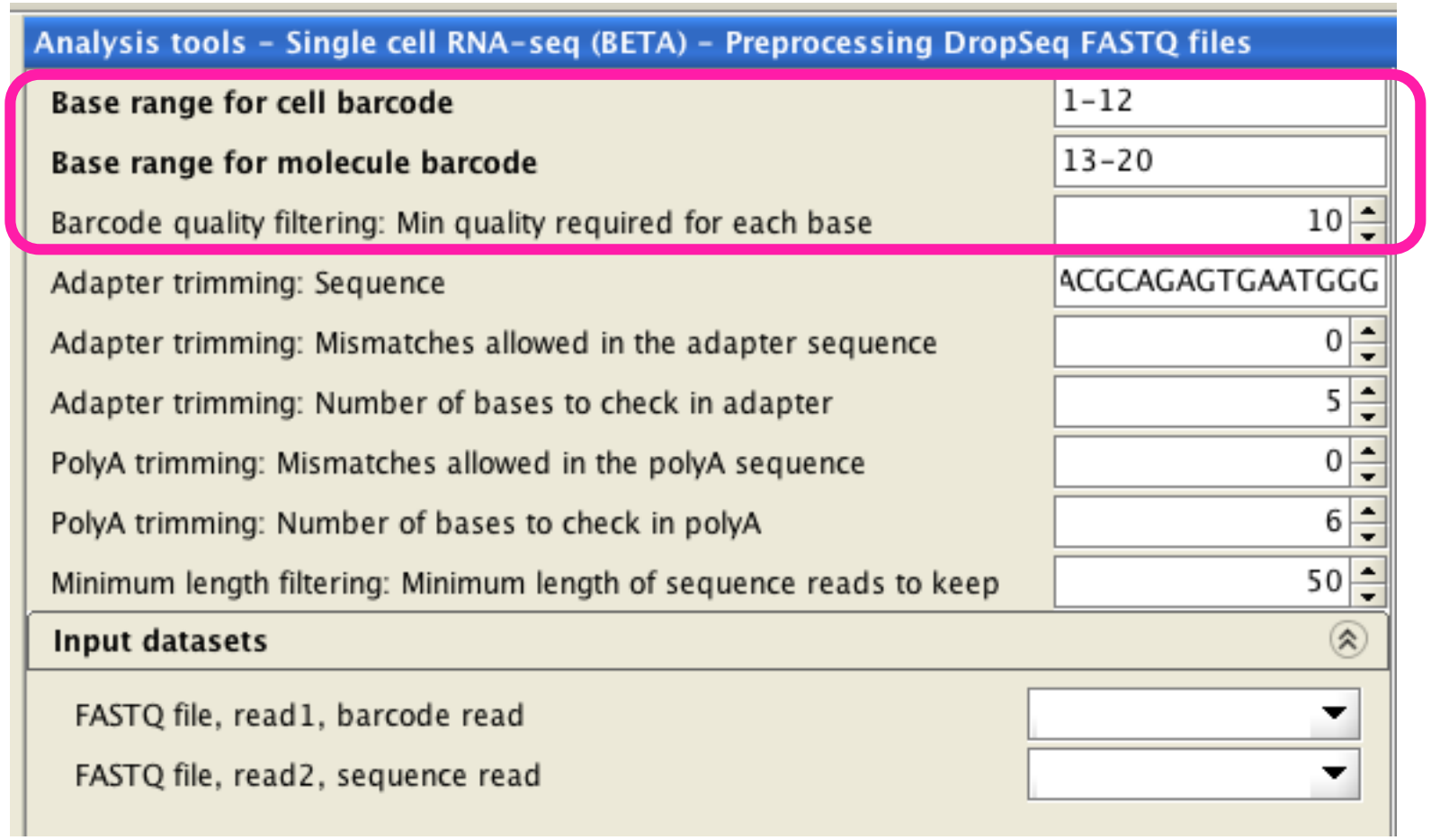
- Filter and trim the reads in the BAM file
  - Adapter sequence [AAGCAGTGGTATCAACGCAGAGTGAATGGG]
  - Mismatches in adapter [0]
  - Number of bases to check in adapter [5]
  - Mismatches in polyA [0]
  - Number of bases to check in polyA [6]

- Convert the tagged and trimmed BAM file back into a FASTQ file for the alignment

- Filter out too short reads of the FASTQ file
  - Minimum length of reads to keep [50]

Preprocessing DropSeq FASTQ files – tagging reads

- You need to tell the tool which bases correspond to which barcode
- We store the barcodes in BAM tags
  - XM = molecular barcode
  - XC = cellular barcode
  - XQ = number of bases that fall below quality threshold
- After this, we can forget the barcode read



Unaligned BAM containing barcode information

Visualisation

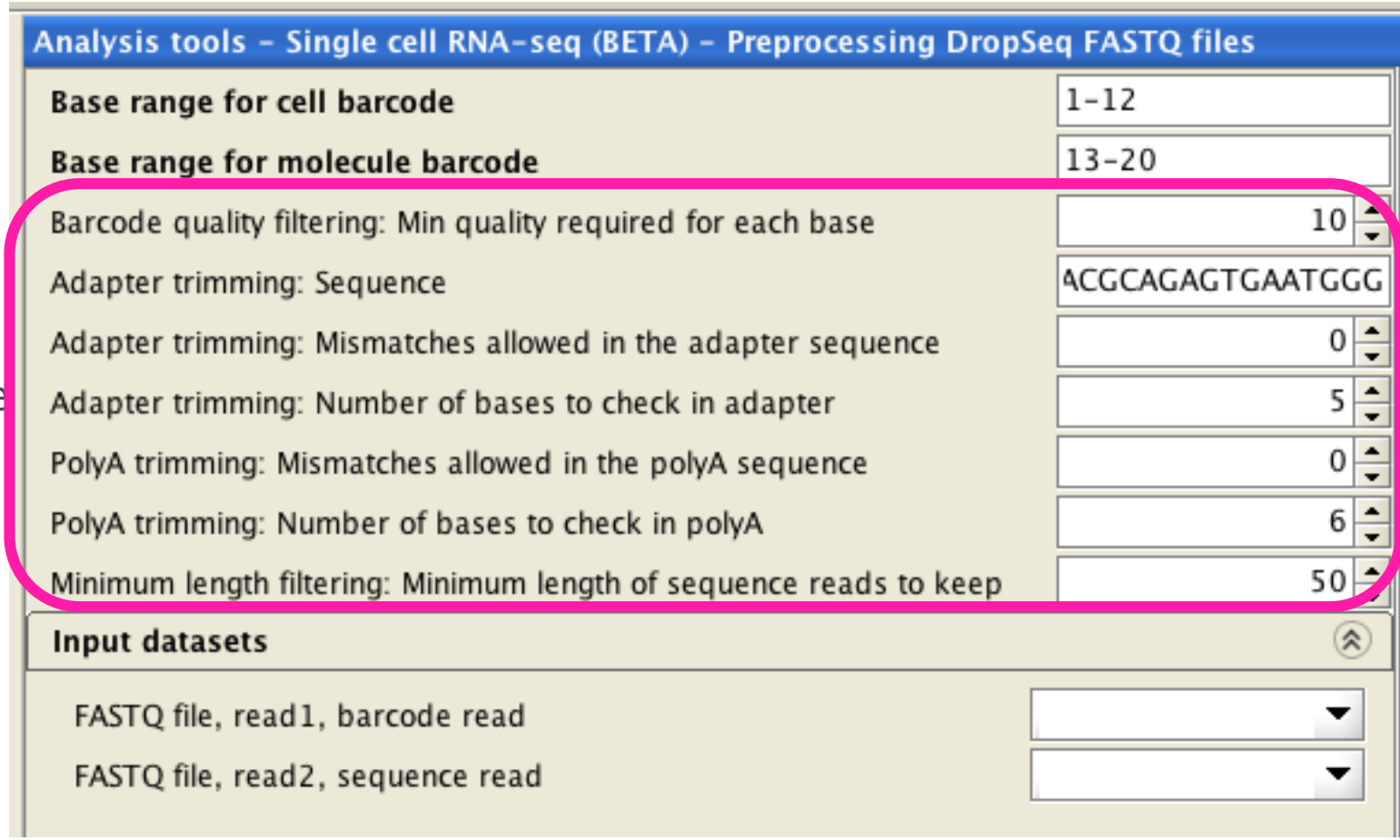
BAM viewer

```
@HD       VN:1.5     SO:queryname
@RG       ID:A       SM:something
SRR1853178.100000036 4        *        0        0        *        *        0        0        GTTTAGGAGATTGTAAAGGGAGGTTTTGTGAAGTTCTAAAAGGTTCTAGTTTGAAGGTCG
A<<AAFFFF.FFFFFAFFFFFFFFFFFFFFFFFFF.FFFFFFFFAAAFAFFAFFAFFF.  XC:Z:TCGGATAGTGAT   RG:Z:A    XM:Z:TAGGGTTT
SRR1853178.100000092 4        *        0        0        *        *        0        0        CTGGGAGAGTCAGCAGGCTCATCTAGAGTGGCTTCGGGCCAATGGAGATTTGCCTCAGCC
<AAAAFF.FAF7FA.FF<AF..FA<F<FFFFA.A<FFFFAA7FFF.F77FFFA<.A.FAF  XC:Z:TTATAGTAAAGA   RG:Z:A    XM:Z:AGGGGGTG
SRR1853178.100000104 4        *        0        0        *        *        0        0        GTTACAAAGACAGCAGGATGTTTGTAAATTATCTTACAATAAACAACTTACAATGAGAAA
AAAA<AFFFAFFAF7FFAFFAF.7F.<.FA.FAF77F.F..AFF7FA...A.<<<.F...  XC:Z:CTTAAAGACGAC   RG:Z:A    XM:Z:GGGGGTAG     XQ:i:1
```

- The 20 base long barcode read is split into the 12 base cell barcode and 8 base UMI (molecular barcode), and both are stored in BAM tags (XC and XM,respectively)
- XQ tag is added, if the barcode read contained low quality bases
  - Reports the number of low quality bases

Preprocessing DropSeq FASTQ files – trimming and filtering

- **Quality**: filter out reads where more than 1 base have poor quality (<10)
- **Adapters**: Trim away any user determined sequences
  - SMART adapter as default
  - How many mismatches allowed (0)
  - How long stretch of the sequence there has to be at least (5)
- **polyA**: hard-clip polyA tails
  - How many A's need to be there before clipping happens (6)
  - mismatches allowed (0)
- **Minimum length**: filter out short (<50bp) reads from the FASTQ file



What if I have several FASTQ files per sample?

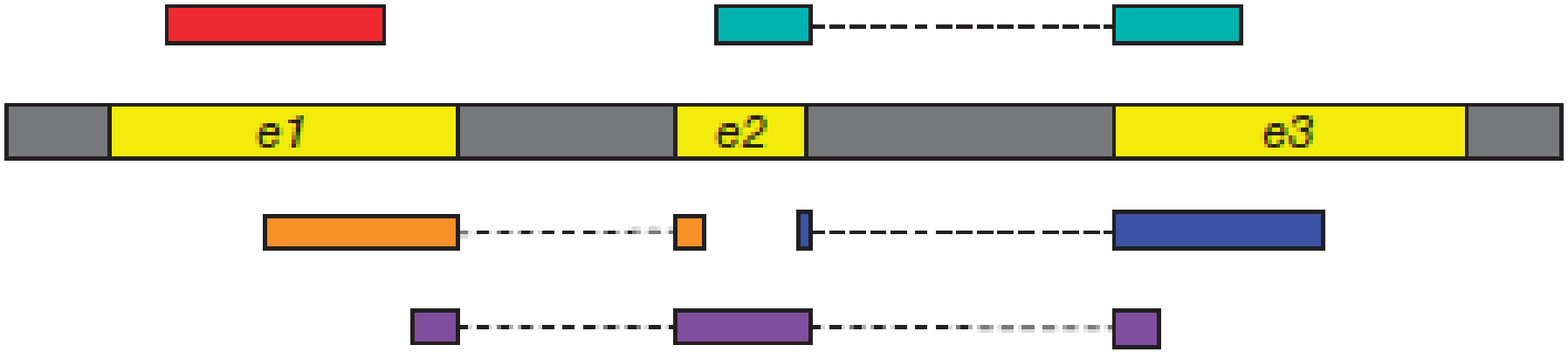
- From Illumina NextSeq device you will get 4 FASTQs for read1 and 4 for read2
- You can merge them together with the tool **Utilities / Merge FASTQ**.
  - You need to run the tool twice: once for read1 files and once for read2 files.
  - Make sure your paired files are named similarly, so that the alphabetical ordering will work and the pairing information is maintained.

From FASTQ to expression matrix –preprocessing of DropSeq data

- Quality control of raw reads
- Preprocessing: tagging with barcodes & filtering
- Alignment to reference genome
- Merge BAM files
- Annotate with gene names
- Estimate the number of usable cells
- Detect bead synthesis errors
- Generate digital expression matrix

Aligning reads to reference genome

- The goal is to find the location where a read originated from
- Challenges
  - Reads contain genomic variants and sequencing errors
  - Genomes contain non-unique sequence and introns
- RNA-seq aligner needs to be able to map splice junction spanning reads to genome non-contiguously
  - Spliced alignments are difficult because sequence signals at splice sites are



Modified from Kim et al (2015) Nature methods 12:358

Alignment programs

- Many aligners have been developed over the years
  - Convert genome fasta file to a data structure which faster to search (e.g. BWT index or suffix array)
  - Differ in speed, memory requirements, accuracy and ability to deal with spliced alignments
- Use splice-aware aligner for mapping RNA-seq reads
  - Examples:
    - STAR (fast and accurate, needs a lot of memory)
    - HISAT2 (fast and accurate, creating the genomic index needs a LOT of memory)
    - TopHat2 (slower, needs less memory)

Splice-aware aligners in Chipster

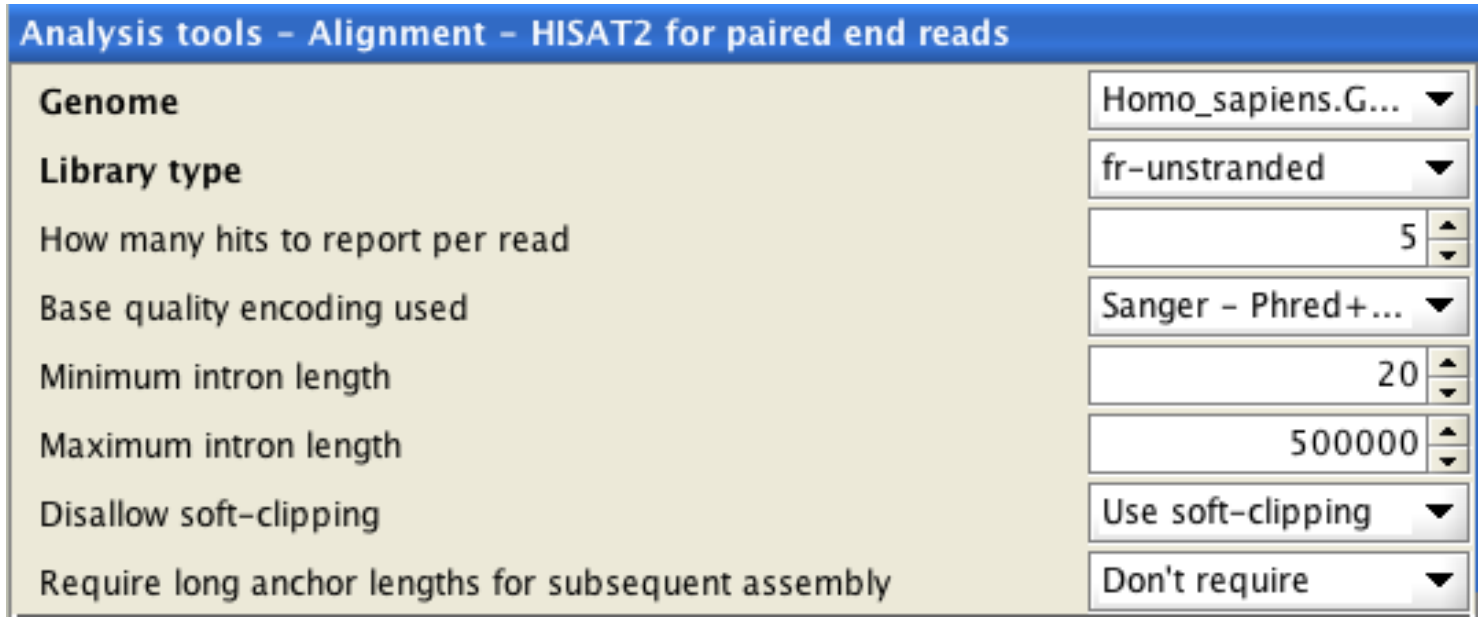
- STAR
  - Human genome available
- HISAT2
  - Human and mouse genome available
  - You can also supply own genome if it is small
- TopHat2
  - Many genomes available
  - You can also supply own genome

- Output files
  - BAM = contains the alignments
  - bai = index file for BAM, required by genome browsers etc
  - log = useful information about the alignment run

HISAT2

- HISAT = Hierarchical Indexing for Spliced Alignment of Transcripts
- Fast spliced aligner with low memory requirement
- Reference genome is (BWT FM) indexed for fast searching
  - Currently Chipster offers human and mouse reference genome
  - Let us know if you need others!
  - You can provide own (small) reference genome in fasta format
- Uses two types of indexes
  - A global index: used to anchor a read in genome (28 bp is enough)
  - Thousands of small local indexes, each covering a genomic region of 56 Kbp: used for rapid extension of alignments (good for spliced reads with short anchors)
- Uses splice site information found during the alignment of earlier reads in the same run

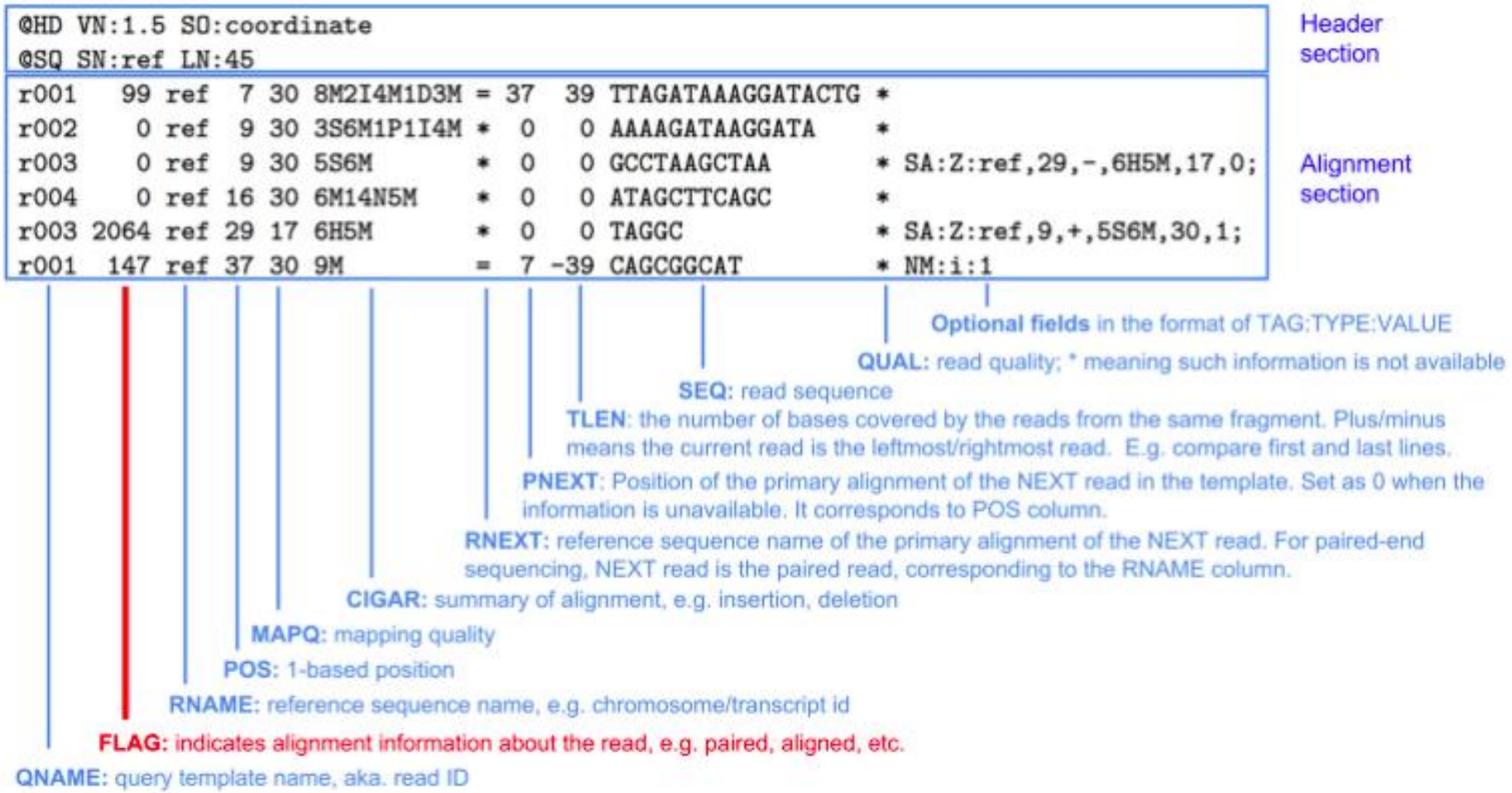
HISAT2 parameters



- Remember to set the strandedness (library type) correctly!
- Note that there can be alignments that are better than the 5 reported ones
- Soft-clipping = read ends don't need to align to the genome, if this maximizes the alignment score

STAR

- STAR = Spliced Transcripts Alignment to a Reference
- Reference genome fasta is converted to a suffix array for fast searching
- 2-pass mapping process
  - splice junctions found during the 1st pass are inserted into the genome index, and all reads are re-mapped in the 2nd mapping pass
  - this doesn't increase the number of detected novel junctions, but it allows more spliced reads mapping to novel junctions.
- Maximum alignments per read -parameter sets the maximum number of loci the read is allowed to map to
  - Alignments (all of them) will be output only if the read maps to no more loci than this. Otherwise no alignments will be output.
- Chipster offers an Ensembl GTF file to detect annotated splice junctions
  - you can also give your own, e.g. GENCODE GTF



- Really nice pages for SAM/BAM interpretation: http://www.samformat.info

Mapping quality

- Confidence in read's point of origin
- Depends on many things, including
  - uniqueness of the aligned region in the genome
  - length of alignment
  - number of mismatches and gaps
- Expressed in Phred scores, like base qualities
  - Q = -10 * log10 (probability that mapping location is wrong)
- Values differ in different aligners. E. g. unique mapping is
  - 60 in HISAT2
  - 255 in STAR
  - 50 in TopHat
  - https://sequencing.qcfail.com/articles/mapq-values-are-really-useful-but-their-implementation-is-a-mess/

CIGAR string

- Letters tell what happened
  - M = match or mismatch
  - I = insertion
  - D = deletion
  - N = intron (in RNA-seq read alignments)
  - S = soft clip (ignore these bases)
  - H = hard clip (ignore and remove these bases)
- Example:

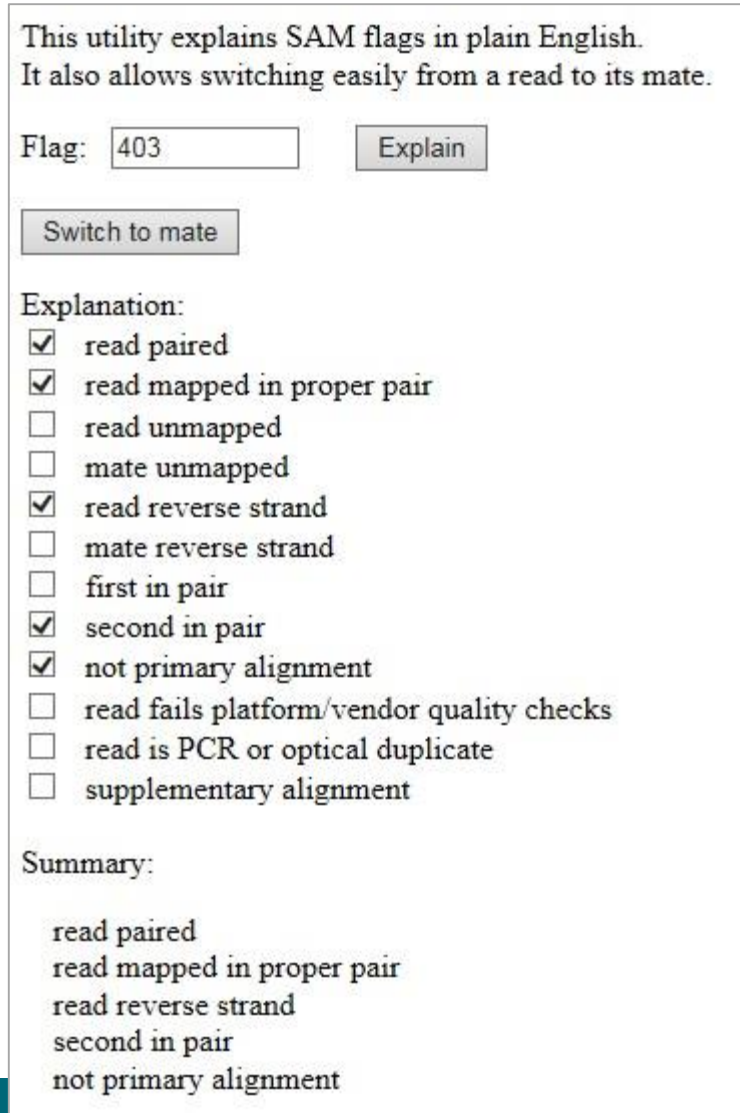
```
@HD VN:1.3 SO:coordinate
@SQ SN:ref LN:45
r001 163 ref 7 30 8M2I4M1D3M = 37 39 TTAGATAAAGGATACTG *
```

  - The corresponding alignment

```
Ref   AGCATGTTAGATAA**GATAGCTGTGCTAGTAGGCAGTCAGCGCCAT
r001       TTAGATAAAGGATA*CTG
```

Flag field in BAM

- Read's flag number is a sum of values
  - E.g. 4 = unmapped, 1024 = duplicate
  - Explained in detail at http://samtools.github.io/hts-specs/SAMv1.pdf
  - You can interpret them at http://broadinstitute.github.io/picard/explain-flags.html


This utility explains SAM flags in plain English.
It also allows switching easily from a read to its mate.

Flag: 403 Explain

Switch to mate

Explanation:

- [x] read paired
- [x] read mapped in proper pair
- [ ] read unmapped
- [ ] mate unmapped
- [x] read reverse strand
- [ ] mate reverse strand
- [ ] first in pair
- [x] second in pair
- [x] not primary alignment
- [ ] read fails platform/vendor quality checks
- [ ] read is PCR or optical duplicate
- [ ] supplementary alignment

Summary:

read paired
read mapped in proper pair
read reverse strand
second in pair
not primary alignment

How did the alignment go? Check the log file

- How many reads mapped to the reference?
  - How many of them mapped uniquely?
- How many pairs mapped?
  - How many pairs mapped concordantly?
- What was the overall alignment rate?

Visualisation

View text

```
25354832 reads; of these:
  25354832 (100.00%) were paired; of these:
    6098272 (24.05%) aligned concordantly 0 times
    18567284 (73.23%) aligned concordantly exactly 1 time
    689276 (2.72%) aligned concordantly >1 times
    ----
    6098272 pairs aligned concordantly 0 times; of these:
      724806 (11.89%) aligned discordantly 1 time
    ----
    5373466 pairs aligned 0 times concordantly or discordantly; of these:
      10746932 mates make up the pairs; of these:
        8812069 (82.00%) aligned 0 times
        1800817 (16.76%) aligned exactly 1 time
        134046 (1.25%) aligned >1 times
82.62% overall alignment rate
```

Log file by STAR

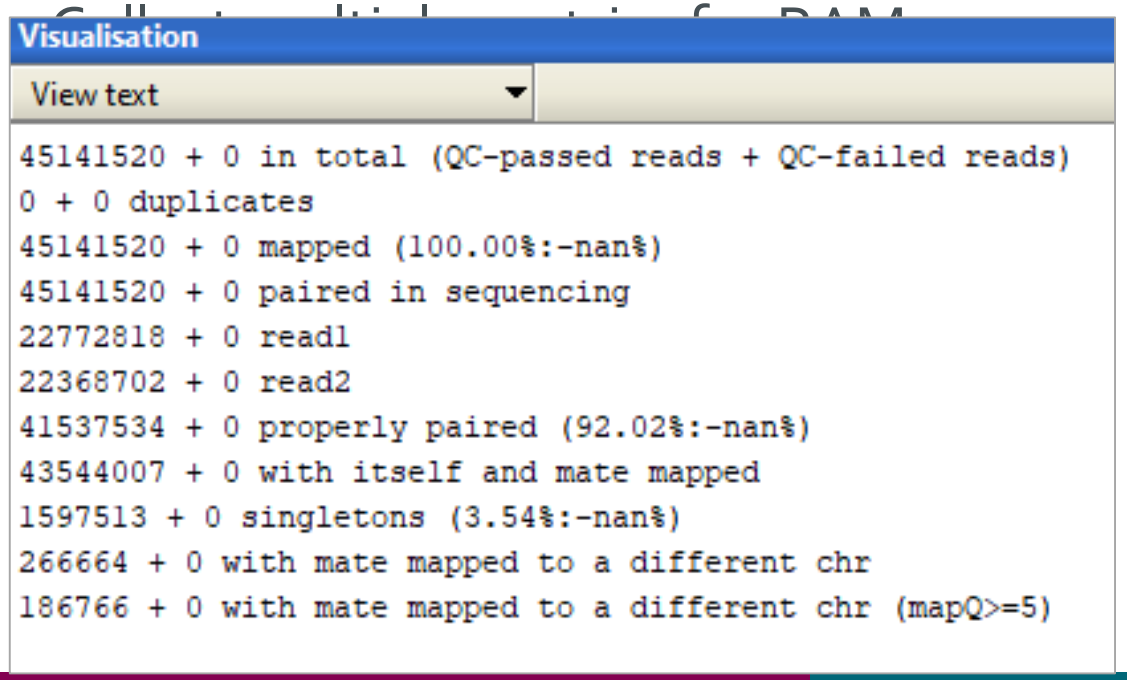
Visualisation

View text

```
                                 Started job on |   Feb 17 12:38:11
                             Started mapping on |   Feb 17 12:47:47
                                    Finished on |   Feb 17 12:52:32
       Mapping speed, Million of reads per hour |   320.27

                           Number of input reads |   25354832
                       Average input read length |   202
                                    UNIQUE READS:
                    Uniquely mapped reads number |   20409554
                         Uniquely mapped reads % |   80.50%
                           Average mapped length |   197.39
                        Number of splices: Total |   12378576
             Number of splices: Annotated (sjdb) |   12378175
                        Number of splices: GT/AG |   12272618
                        Number of splices: GC/AG |   89423
                        Number of splices: AT/AC |   9589
                Number of splices: Non-canonical |   6946
                       Mismatch rate per base, % |   0.39%
                          Deletion rate per base |   0.01%
                         Deletion average length |   1.75
                         Insertion rate per base |   0.01%
                        Insertion average length |   1.36
                             MULTI-MAPPING READS:
        Number of reads mapped to multiple loci |   970016
             % of reads mapped to multiple loci |   3.83%
        Number of reads mapped to too many loci |   11610
             % of reads mapped to too many loci |   0.05%
                               UNMAPPED READS:
       % of reads unmapped: too many mismatches |   0.00%
                 % of reads unmapped: too short |   15.55%
                     % of reads unmapped: other |   0.08%
                               CHIMERIC READS:
                        Number of chimeric reads |   0
                             % of chimeric reads |   0.00%
```

Other tools for checking BAM files

- Count alignments in BAM
  - How many alignments does the BAM contain.
  - Includes an optional mapping quality filter.
- Count alignments per chromosome in BAM
- Count alignment statistics for BAM
- Collect multiple metrics for BAM

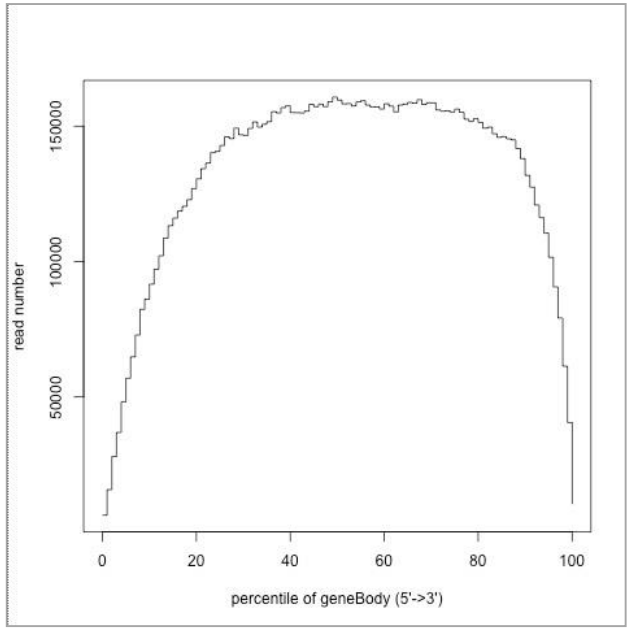


Tools for manipulating BAM files

- Make a subset of BAM
  - Retrieve alignments for a given chromosome/region, e.g. chr1:100-1000
  - Can filter based on mapping quality
- Index BAM
- Convert SAM to BAM, sort and index BAM
  - "Preprocessing" when importing SAM/BAM, runs on your computer.
  - The tool available in the "Utilities" category runs on the server

Quality check after alignment with RseQC

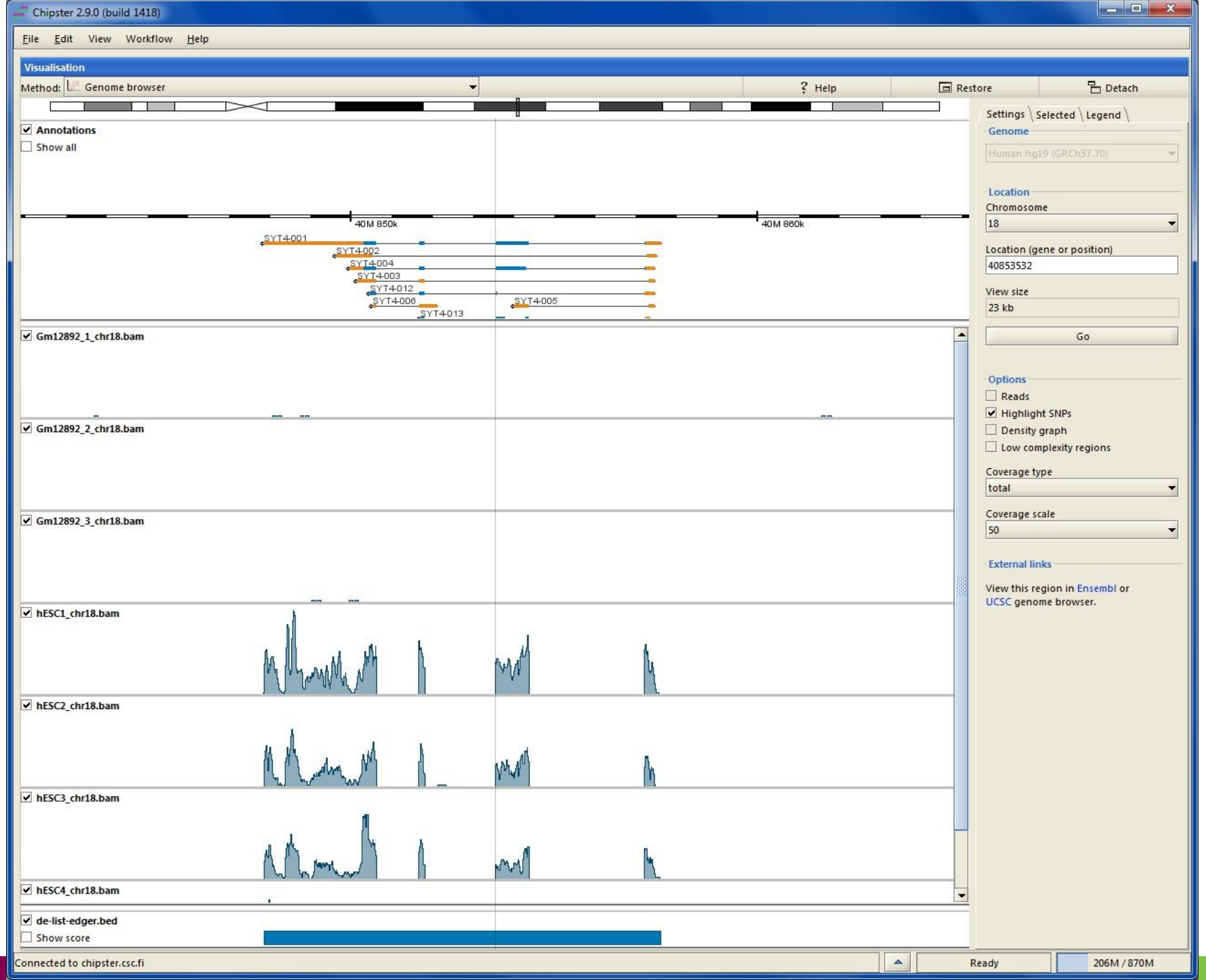
- Checks coverage uniformity, read distribution between different genomic regions, novelty of splice junctions, etc.
- Takes a BAM file and a BED file
  - Chipster has BED files available for several organisms
  - You can also use your own BED if you prefer

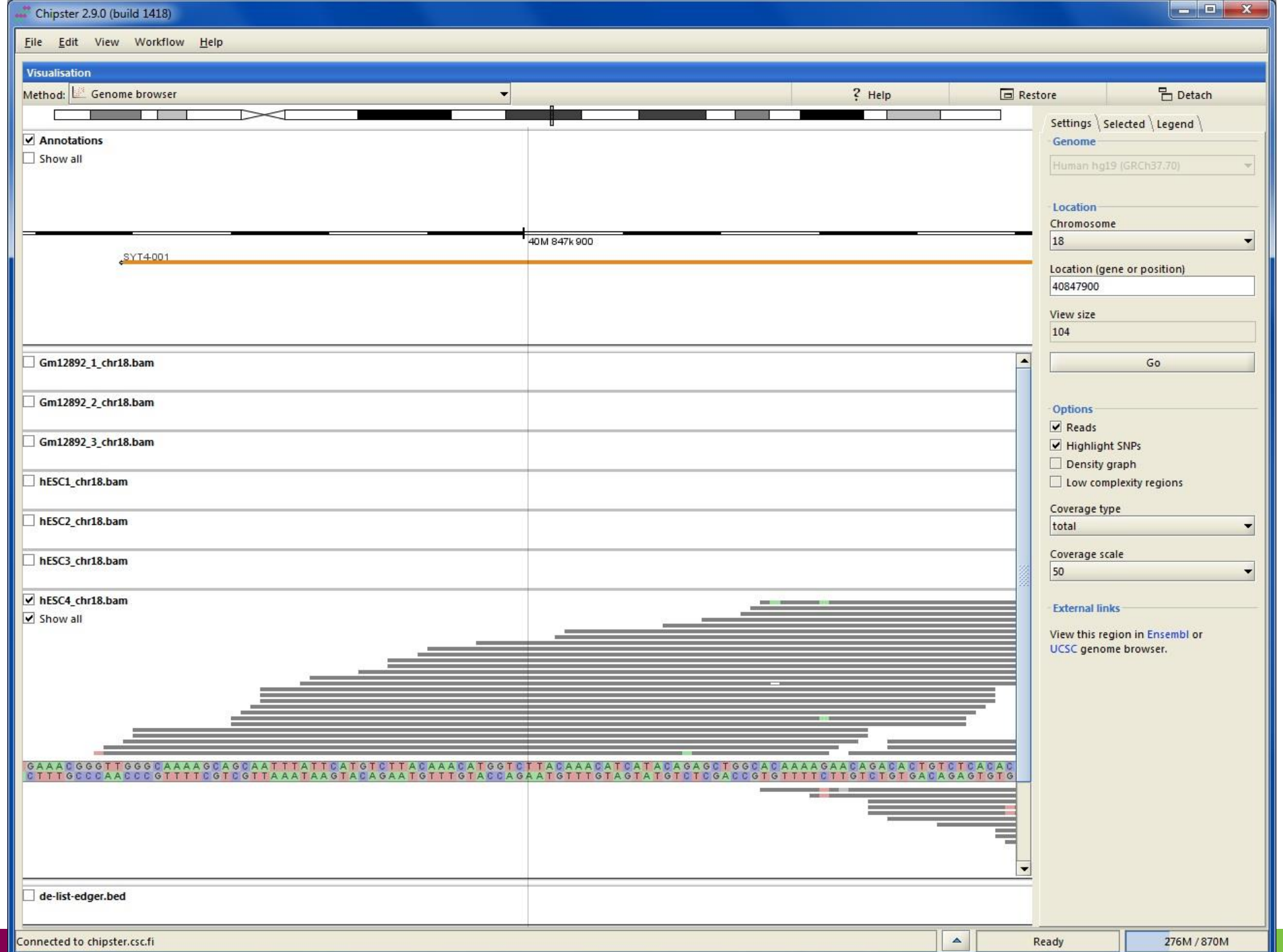



```
read_distribution:

Total Reads                   84808
Total Tags                    116738
Total Assigned Tags           111352
=================================================================
Group               Total_bases         Tag_count           Tags/Kb
CDS_Exons           2211343             90961               41.13
5'UTR_Exons         529860              1662                3.14
3'UTR_Exons         1415234             12423               8.78
Introns             25801210            5349                0.21
TSS_up_1kb          1295771             31                  0.02
TSS_up_5kb          5332522             321                 0.06
TSS_up_10kb         8804879             584                 0.07
TES_down_1kb        1292506             217                 0.17
TES_down_5kb        5108821             344                 0.07
TES_down_10kb       8282641             373                 0.05
=================================================================
```

Visualize alignments in genomic context: Chipster Genome Browser

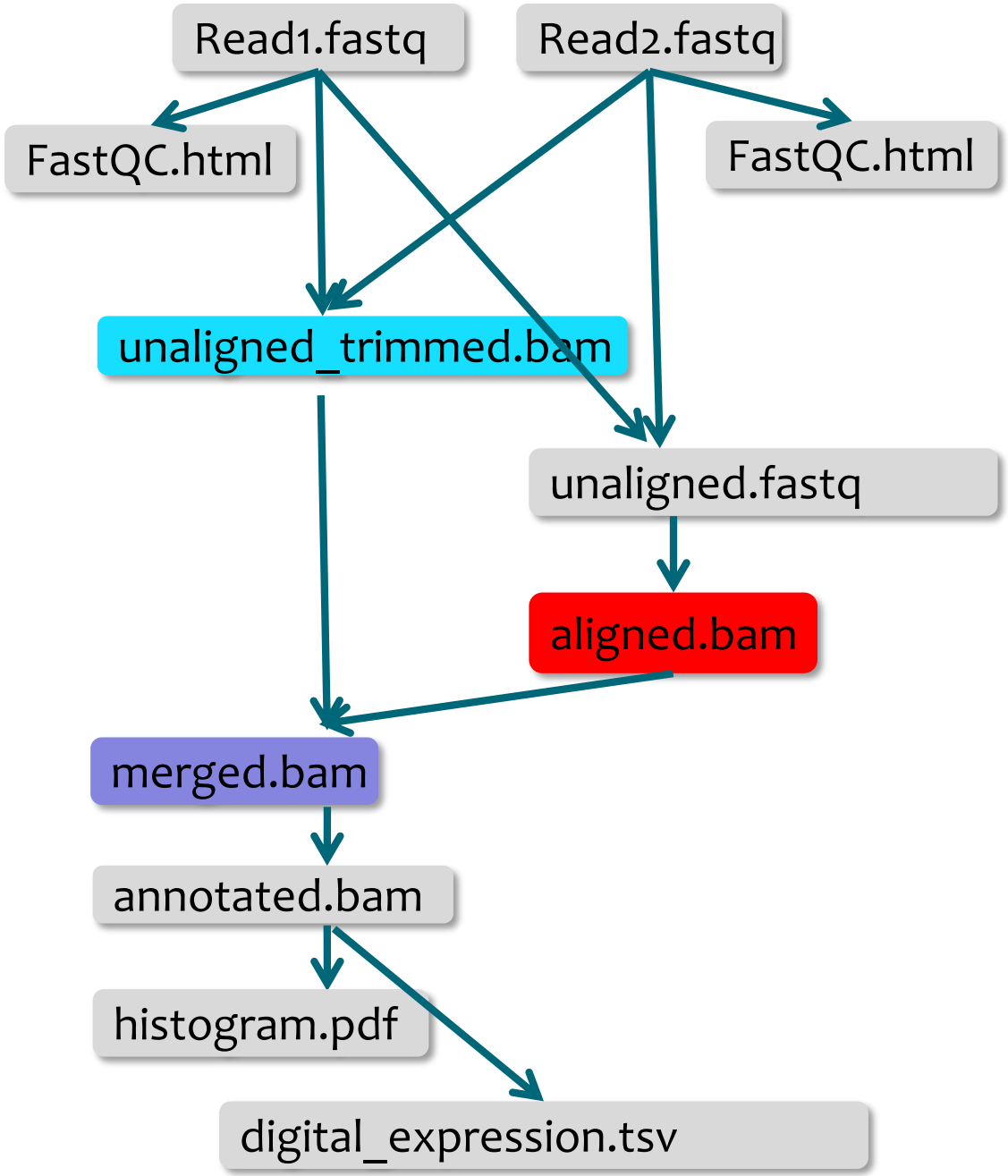
- Integrated with Chipster analysis environment
- Automatic coverage calculation (total and strand-specific)
- Zoom in to nucleotide level
- Highlight variants
- Jump to locations using BED, GTF, VCF and tsv files
- View details of selected BED, GTF and VCF features
- Several views (reads, coverage profile, density graph)


Chipster 2.9.0 (build 1418)
File Edit View Workflow Help
Visualisation
Method: Genome browser
Help
Restore
Detach
Annotations
Show all
40M 850k
40M 860k
SYT4-001
SYT4-002
SYT4-004
SYT4-003
SYT4-012
SYT4-006
SYT4-005
SYT4-013
Gm12892_1_chr18.bam
Gm12892_2_chr18.bam
Gm12892_3_chr18.bam
hESC1_chr18.bam
hESC2_chr18.bam
hESC3_chr18.bam
hESC4_chr18.bam
de-list-edger.bed
Show score
Settings
Selected
Legend
Genome
Human hg19 (GRCh37.70)
Location
Chromosome
18
Location (gene or position)
40853532
View size
23 kb
Go
Options
Reads
Highlight SNPs
Density graph
Low complexity regions
Coverage type
total
Coverage scale
50
External links
View this region in Ensembl or
UCSC genome browser.
Connected to chipster.csc.fi
Ready
206M / 870M


Chipster 2.9.0 (build 1418)
File Edit View Workflow Help
Visualisation
Method: Genome browser
Help
Restore
Detach
Annotations
Show all
40M 847k 900
SYT4-001
Gm12892_1_chr18.bam
Gm12892_2_chr18.bam
Gm12892_3_chr18.bam
hESC1_chr18.bam
hESC2_chr18.bam
hESC3_chr18.bam
hESC4_chr18.bam
Show all
de-list-edger.bed
Settings
Selected
Legend
Genome
Human hg19 (GRCh37.70)
Location
Chromosome
18
Location (gene or position)
40847900
View size
104
Go
Options
Reads
Highlight SNPs
Density graph
Low complexity regions
Coverage type
total
Coverage scale
50
External links
View this region in Ensembl or
UCSC genome browser.
Connected to chipster.csc.fi
Ready
276M / 870M

From FASTQ to expression matrix –preprocessing of DropSeq data

- Quality control of raw reads
- Preprocessing: tagging with barcodes & filtering
- Alignment to reference genome
- Merge BAM files
- Annotate with gene names
- Estimate the number of usable cells
- Detect bead synthesis errors
- Generate digital expression matrix



Import files

Quality control / FASTQC

Preprocess:

- FASTQ to BAM
- Tag BAM
- Filter and trim BAM
- BAM to FASTQ
- Filter too short reads

-filter bad quality barcodes
-trim adapter sequences
-trim polyA tails

Aligner (STAR, HISAT)

Merge BAM

Combines the barcode tags and the alignment info

Tag reads with gene names

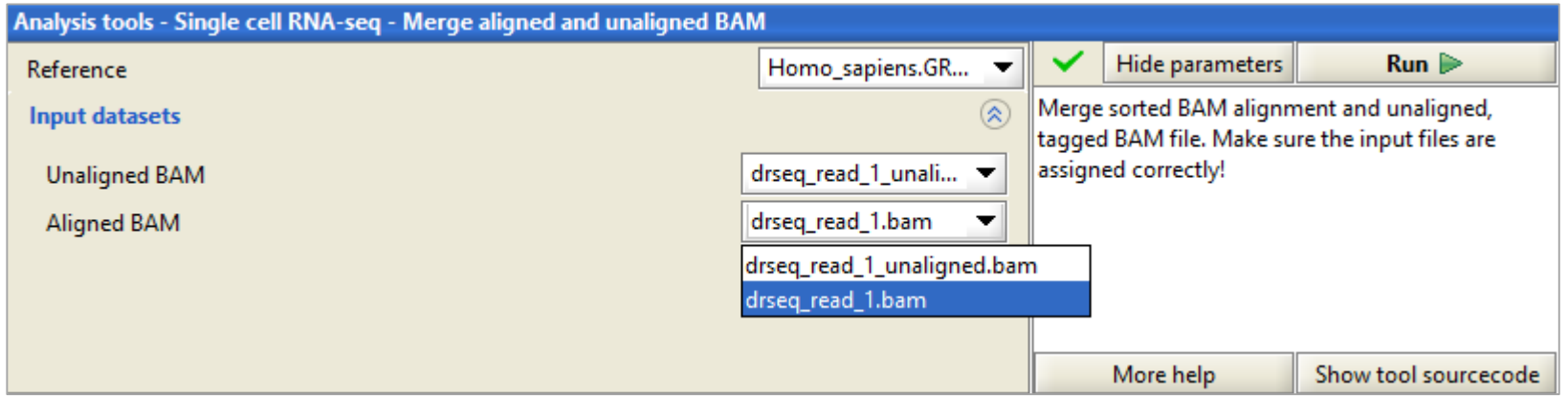
Estimate number of usable cells

Create digital expression matrix

+ Detect bead synthesis errors

Merge BAM files

- We need read's cell and molecular barcode and alignment location in one file
  - Unaligned BAM contains the barcode information
  - Aligned BAM file contains genomic positions but not the barcodes (because aligners use FASTQ)
- Before merging, the tool sorts the files in queryname (read name) order
- Secondary alignments are ignored!
- Remember to…
  - Make sure the files are correctly assigned!
  - Choose the reference



Merged BAM contains alignment information and barcodes in tags

Visualisation

BAM viewer | Maximise | Detach | Close

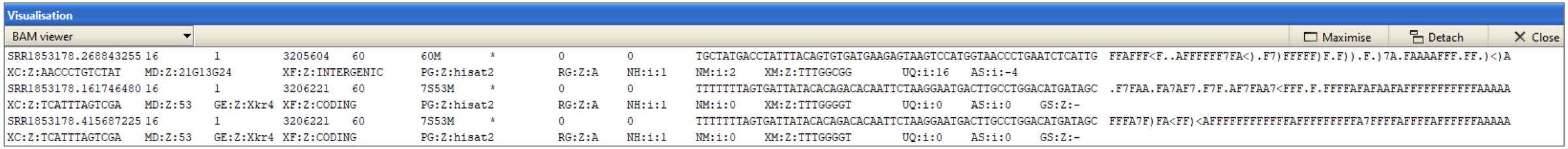
```
@HD	VN:1.5	SO:coordinate
@SQ	SN:1	LN:195471971	M5:c4ec915e7348d42648eefc1534b71c99	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:2	LN:182113224	M5:fe020a692e23f8468b376e14e304a10f	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:3	LN:160039680	M5:50f9385167e70825931231ddf1181b80	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:4	LN:156508116	M5:e7bdfb3ce7f54d2720b0718ed2ea063c	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:5	LN:151834684	M5:095f3d4ebe1f0bafff057cc9b130186d	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:6	LN:149736546	M5:62628d042ea5e01adff5b481d23def67	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:7	LN:145441459	M5:65da9ab01a76dcbcaef6f32a753585c1	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:8	LN:129401213	M5:dd2d079a37c02e8a3f95abff9e37ac69	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:9	LN:124595110	M5:ef8a85e56b750c1056865636lfac7990	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:10	LN:130694993	M5:7831ecda5dd6bcf838e2452ea0139eac	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:11	LN:122082543	M5:e168c7a3194813f597181f26bb1bd13f	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:12	LN:120129022	M5:671f85bb54a6e097d631e2e2dd813ad4	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:13	LN:120421639	M5:7f9b9fa3fbd9a38634107dfdc7fd8dc8	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:14	LN:124902244	M5:bf4e1efa25a8fd23b41c91f9bcb86388	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:15	LN:104043685	M5:106358dace00e5825ae337c1f1821b5e	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:16	LN:98207768	M5:5482110a6cedd3558272325eee4c5a17	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:17	LN:94987271	M5:0d21e8edbfcd8410523b2b94e6dae892	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:18	LN:90702639	M5:46fda2f7e6dbf91bff91d6703e004afb	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:19	LN:61431566	M5:7d363594531514ce41dfacfd97bc995d	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:X	LN:171031299	M5:b3db6d6da78d5268688ee395c2c8cb4a	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:Y	LN:91744698	M5:837a35bcca18643d030d4eec5e5b9c64	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@SQ	SN:MT	LN:16299	M5:11c8af2a2528b25f2c080ab7da42edda	UR:file:/mnt/data/jobs-data/0c5e9e78-7165-4603-9322-8a39b1bdfc2e/815303f5-e588-4fe8-8301-5bf0687fb114/data/Mus_musculus.GRCm38.fa
@RG	ID:A	SM:something
@PG	ID:hisat2	PN:hisat2	VN:2.1.0	CL:"/opt/chipster/tools/hisat2/hisat2-align-s --wrapper basic-0 --phred33 --min-intronlen 20 --max-intronlen 500000 -x Mus_musculus.GRCm38.90 -k 5 -p 16 --passthrough -U drseq_read_1.fq.gz"
SRR1853178.393696883	16	1	3020298	1	60M	*	0	0	CCTTTTCAGGTGTGTTAGGGTGCCCTGGACTGGGCGAAGTGGGTGTTCTGGGTTCTGATG	AAAAAF<)FFFFFF.F.FFF.FFFFFFFFFAFFFAFFAF.F.FF..FAFFFFFFFFAAFF
XC:Z:CAGTCTGGTTGA	MD:Z:60	PG:Z:hisat2	RG:Z:A	NH:i:5	NM:i:0	XM:Z:GGGTGGTC	UQ:i:0	AS:i:0
SRR1853178.210234973	0	1	3023534	0	4S56M	*	0	0	CAATAGATCCTGAAGAAATCCTAAACACCATCAGATCCTTCTACAAAAGGCTATACTCAA	A)AAAF)FF<F.<<FAF.FFF))))F<FF7FAFA.FFAAFFF.<<A..FFFA.AAF.F<.
XC:Z:TGTATCTCTGTG	MD:Z:56	PG:Z:hisat2	RG:Z:A	NH:i:5	NM:i:0	XM:Z:AAGATAGC	UQ:i:0	AS:i:-4
SRR1853178.185505387	0	1	3055764	1	60M	*	0	0	CATGTATGTTTTCTGTTTCGTTGTCTAAATGATTGTTTCATTGTCTGAATGACTGGCTGT	AAAAAFFAFFFFFAAFFFFFFFFFAFAFFFFFFF<FFAFF.FAFFAFFFFF7FFFFF<F
XC:Z:CATGTTCTGAAG	MD:Z:60	PG:Z:hisat2	RG:Z:A	NH:i:5	NM:i:0	XM:Z:GCGAGATT	UQ:i:0	AS:i:0
```

From FASTQ to expression matrix –preprocessing of DropSeq data

- Quality control of raw reads
- Preprocessing: tagging with barcodes & filtering
- Alignment to reference genome
- Merge BAM files
- Annotate with gene names
- Estimate the number of usable cells
- Detect bead synthesis errors
- Generate digital expression matrix

Annotate

- GE tag = gene name name, if the read overlaps an exon
- XF tag = location (intron, exon…)
- Choose the annotation file (GTF) from Chipster server or use your own



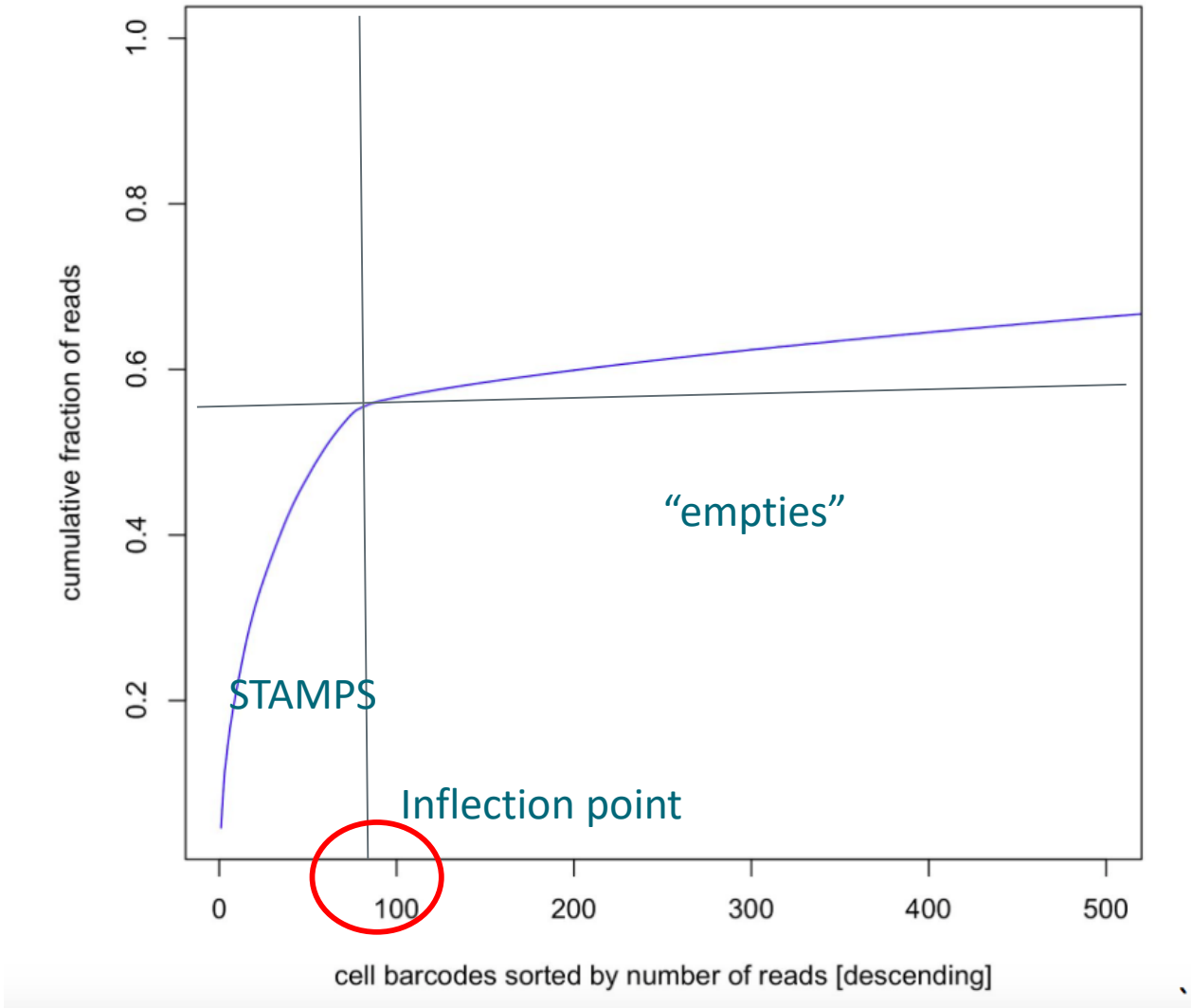
From FASTQ to expression matrix –preprocessing of DropSeq data

- Quality control of raw reads
- Preprocessing: tagging with barcodes & filtering
- Alignment to reference genome
- Merge BAM files
- Annotate with gene names
- Estimate the number of usable cells
- Detect bead synthesis errors
- Generate digital expression matrix

Estimate the number of usable cells

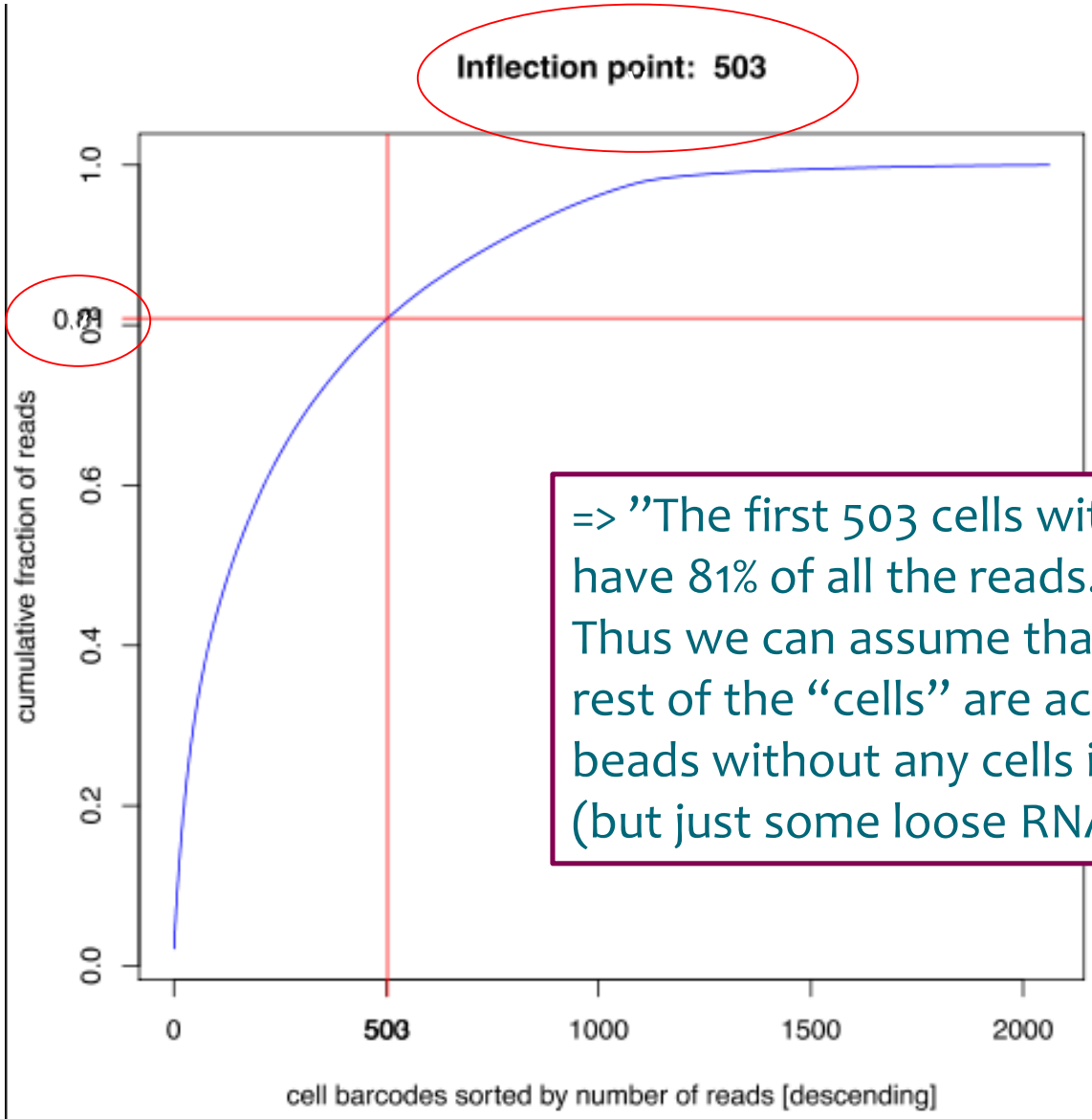
- How many cells do you want to keep?
- To estimate this: the inflection point
  1. extract the number of reads per cell (barcode)
  2. plot the cumulative distribution of reads
  3. select the "knee" of the distribution (knee = inflection point)

  - The number of **STAMPs** (= beads exposed to a cell in droplets): cell barcodes to the left of the inflection point
  - **Empties** (= beads only exposed to ambient RNA in droplets): to the right of the inflection point



Inflection point also computed using the inflection R -package

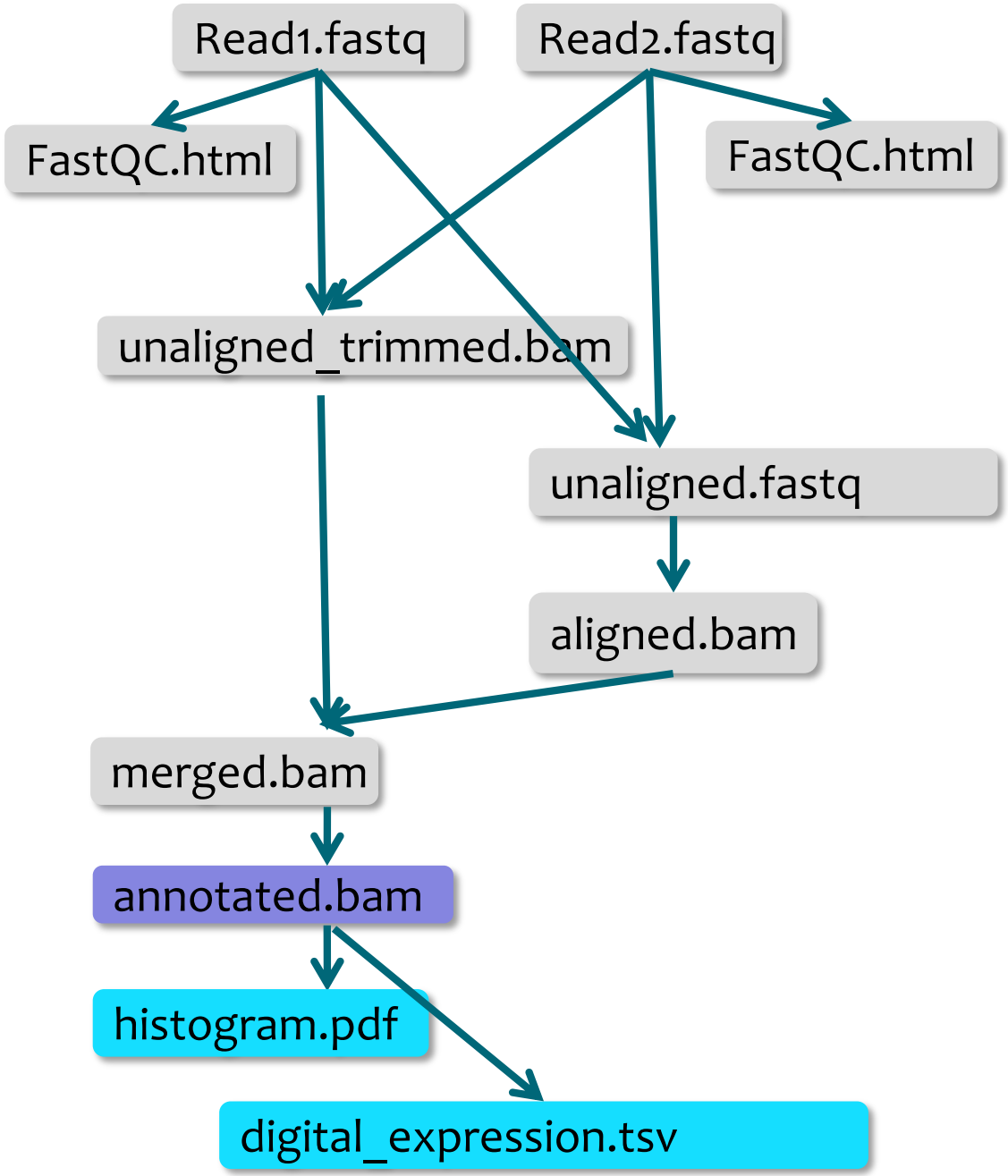
- Sometimes visual estimation can be tricky, so the tool also gives a numerical estimate
- Finds the inflection point using extreme distance estimator (ede) from *inflection* R-package
- Thank you **Dawit Yohannes**!



=> ”The first 503 cells with most reads have 81% of all the reads.”
Thus we can assume that the rest of the “cells” are actually beads without any cells in them (but just some loose RNA)

From FASTQ to expression matrix –preprocessing of DropSeq data

- Quality control of raw reads
- Preprocessing: tagging with barcodes & filtering
- Alignment to reference genome
- Merge BAM files
- Annotate with gene names
- Estimate the number of usable cells
- Detect bead synthesis errors
- Generate digital expression matrix



Import files

Quality control / FASTQC

Preprocess:

- FASTQ to BAM
- Tag BAM
- Filter and trim BAM
- BAM to FASTQ
- Filter too short reads

-filter bad quality barcodes
-trim adapter sequences
-trim polyA tails

Aligner (STAR, HISAT)

Merge BAM

Combines the barcode tags and the alignment info

Tag read with gene names

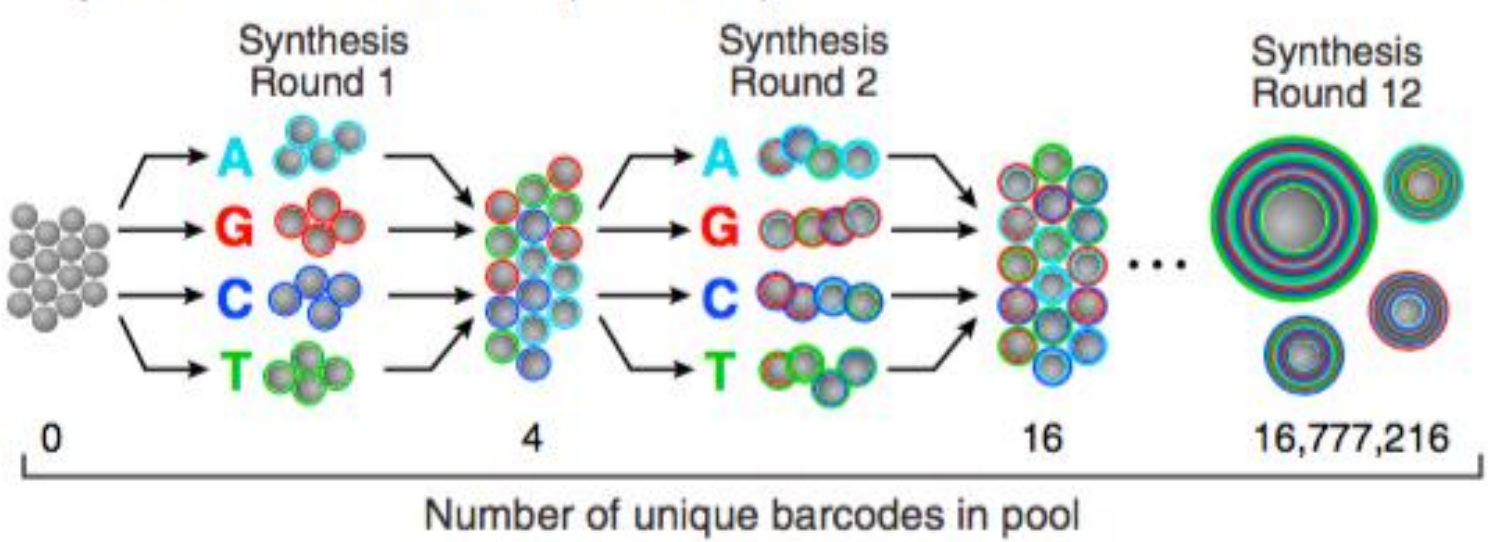
Estimate number of usable cells

Create digital expression matrix

+ Detect bead synthesis errors

Detect bead synthesis errors

ATTC GAGT TAT? CAGC GTAAT TTTT
Cell (12) UMI (8)



- Known problem with ChemGenes beads: a percentage of beads did not undergo all twelve split-and-pool bases.
  - A **mixed base** at base 12 (=actually 1st base of UMI) and a **fixed T** base at base 20 (=actually the 1st base of the polyT segment)
- DropSeq-tool *DetectBeadSynthesisErrors* fixes this:
  - The last base of cell barcode is trimmed off and all cell barcodes with identical sequence at the first 11 bases are merged
  - If any other UMI base is fixed, the reads with that cell barcode are discarded
- Parameter: number of barcodes on which to perform the correction
  - roughly 2 times the anticipated number cells (empirically found that this allows to recover nearly every defective cell barcode).

Error types

- SYNTHESIS_MISSING_BASE
  - 1 or more bases missing from cell barcode → T's at the end of UMIs
  - Fix: insert an "N" (reading frame fixed) and merge. If more than 1 missing, discard these reads

- SINGLE_UMI_ERROR
  - At each position of the UMIs, single base appears in >80% of the UMIs for that cell.
  - Fix: cell barcodes with this property are dropped
- PRIMER_MATCH
  - Same as with SINGLE_UMI_ERROR, + the UMI matches one of the PCR primers
  - Fix: these barcodes are dropped
- OTHER
  - UMIs are extremely skewed towards at least one base, but not at all 8 positions
  - Fix: these barcodes are dropped

Synthesis statistics

- **synthesis_stats.txt** contains a bunch of useful information:
  1. CELL_BARCODE the 12 base cell barcode
  2. NUM_UMI the number of total UMIs observed
  3. FIRST_BIASED_BASE the first base position where any bias is observed (**-1** for no detected bias)
  4. SYNTH_MISSING_BASE as 3 but specific to runs of T's at the end of the UMI
  5. ERROR_TYPE
  6. For bases 1-8 of the UMI, the observed base counts across all UMIs. This is a "|" delimited field, with counts of the A,C,G,T,N bases.

- **synthesis_stats.summary.txt** contains a histogram of the SYNTHESIS_MISSING_BASE errors*, as well as the counts of all other errors, the number of total barcodes evaluated, and the number of barcodes ignored.

  *1 or more bases missing from cell barcode => T's at the end of UMIs

CELL_BARCODE	NUM_UMI	FIRST_BIASED_BASE	SYNTH_MISSING_BASE	ERROR_TYPE	BASE_1	BASE_2
TCATTTAGTCGA	11832	-1	-1	NO ERROR	1564|1783|5254|3231|0	1620|18
					A | C | G | T | N	

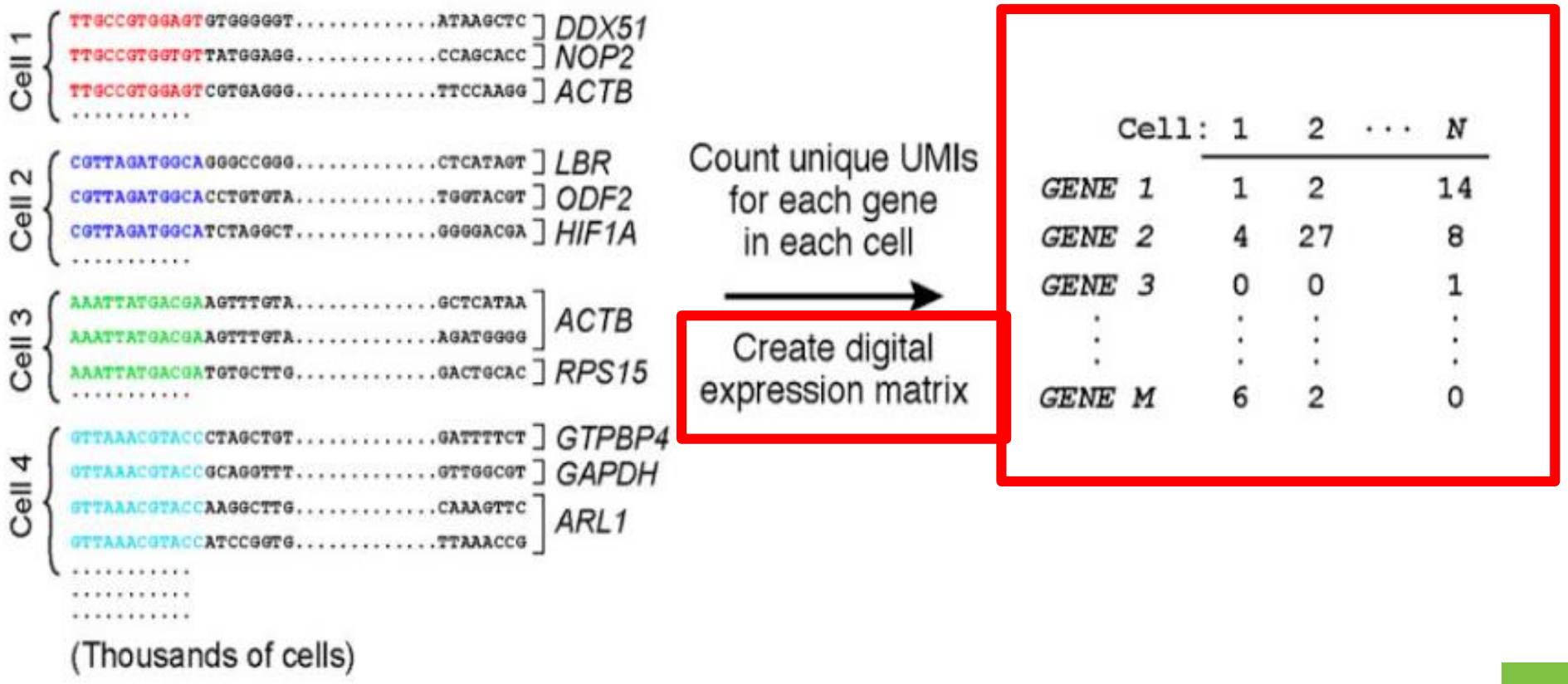
From FASTQ to expression matrix –preprocessing of DropSeq data

- Quality control of raw reads
- Preprocessing: tagging with barcodes & filtering
- Alignment to reference genome
- Merge BAM files
- Annotate with gene names
- Estimate the number of usable cells
- Detect bead synthesis errors
- Generate digital expression matrix

Digital expression matrix

- To digitally count gene transcripts:
  1. a list of high quality UMIs in each gene, within each cell, is assembled
  2. UMIs within edit distance = 1 are merged together
  3. The total number of unique UMI sequences is counted

     → this number is reported as the number of transcripts of that gene for a given cell



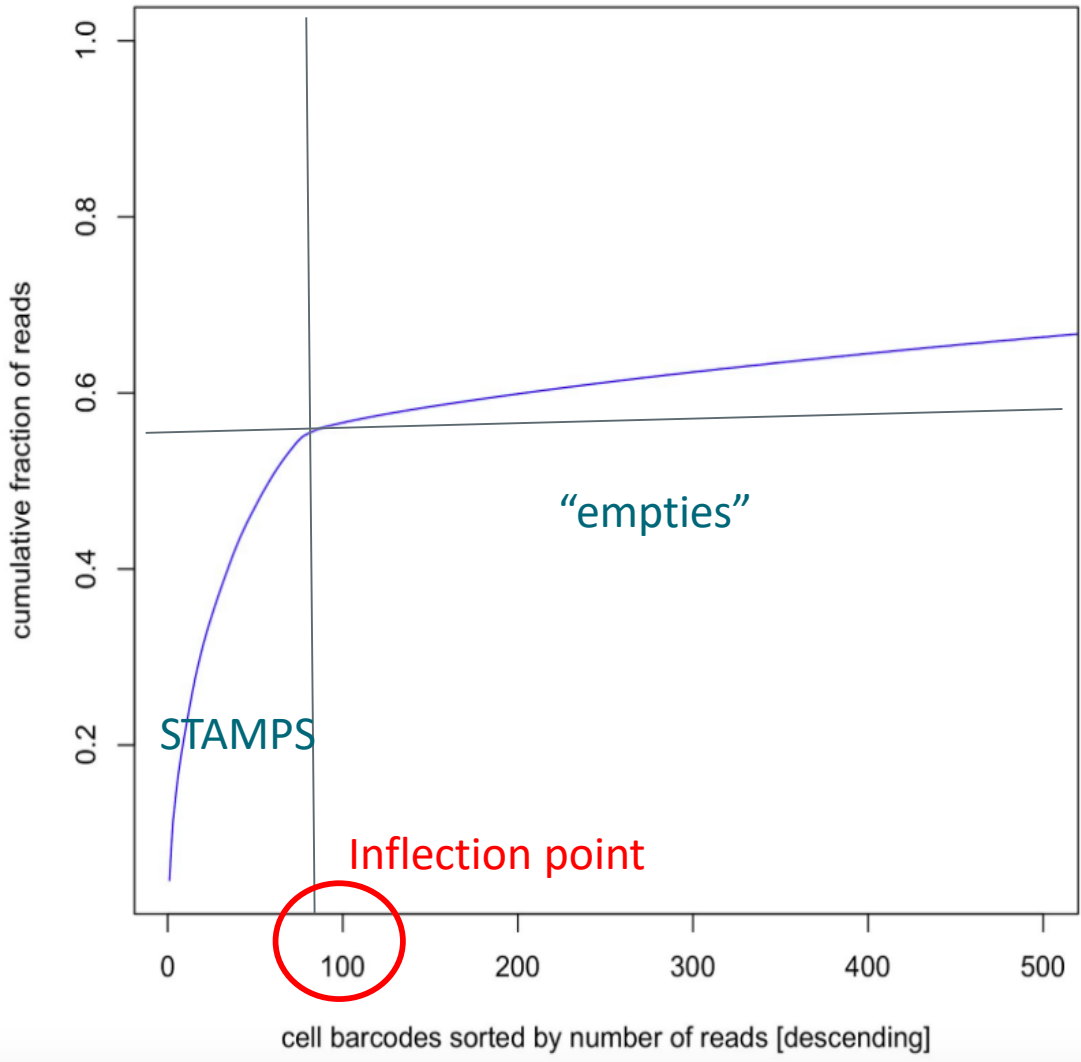
Filtering the data for DGE

- Why don't we just take all the cells?
  - the aligned BAM can contain hundreds of thousands of cell barcodes
  - Some of them are "empties"
  - Some cell (barcode)s contain just handful of reads
  - It is painful to deal with a huge matrix.

- Filtering based on:
  - Number of core barcodes (**top X cells** with most reads)
  - Minimum number of expressed **genes** per cell

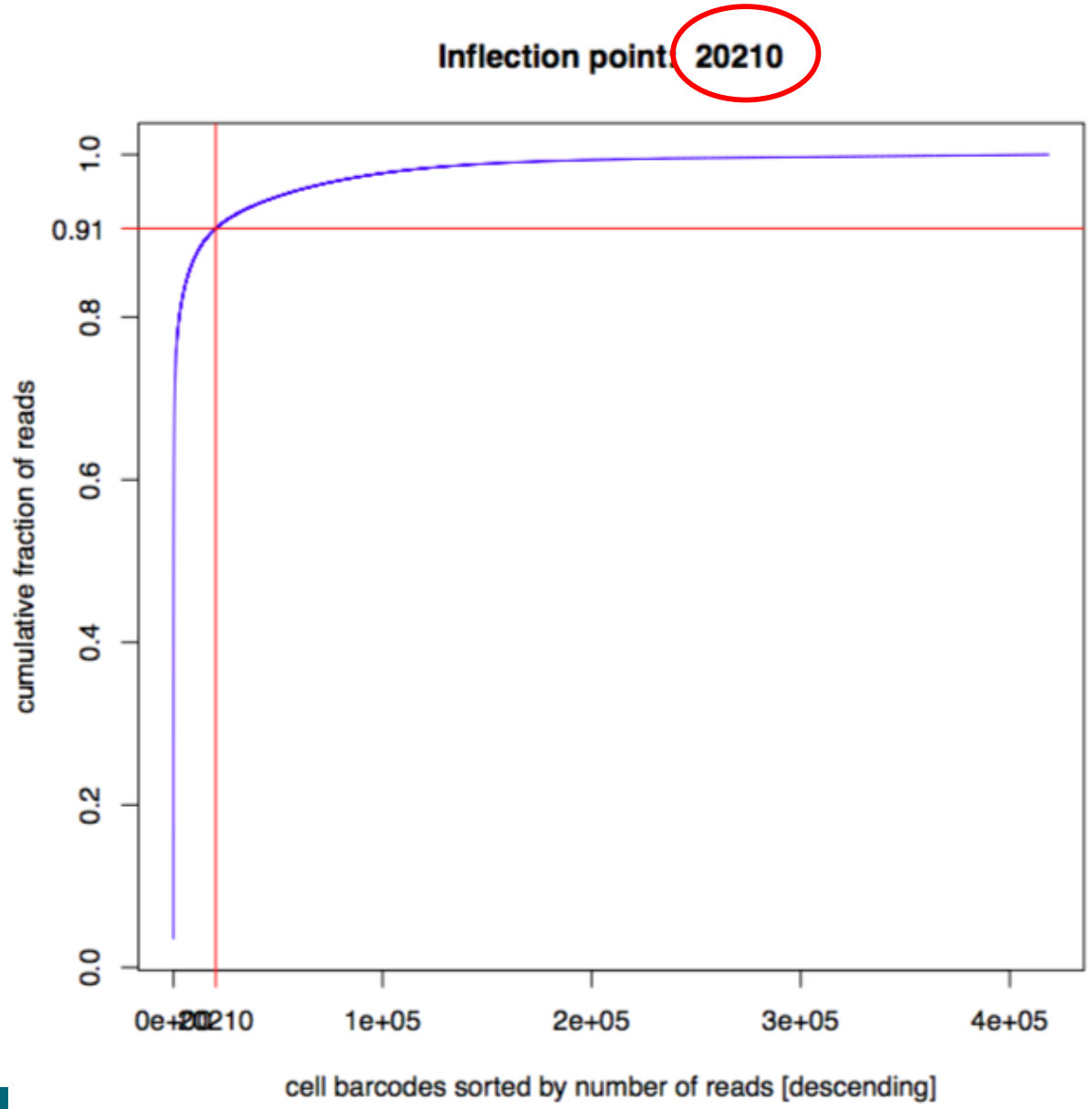
NOTE: You can also choose to have a bigger number of cells and use that for further analysis (Seurat has its own filtering tools)

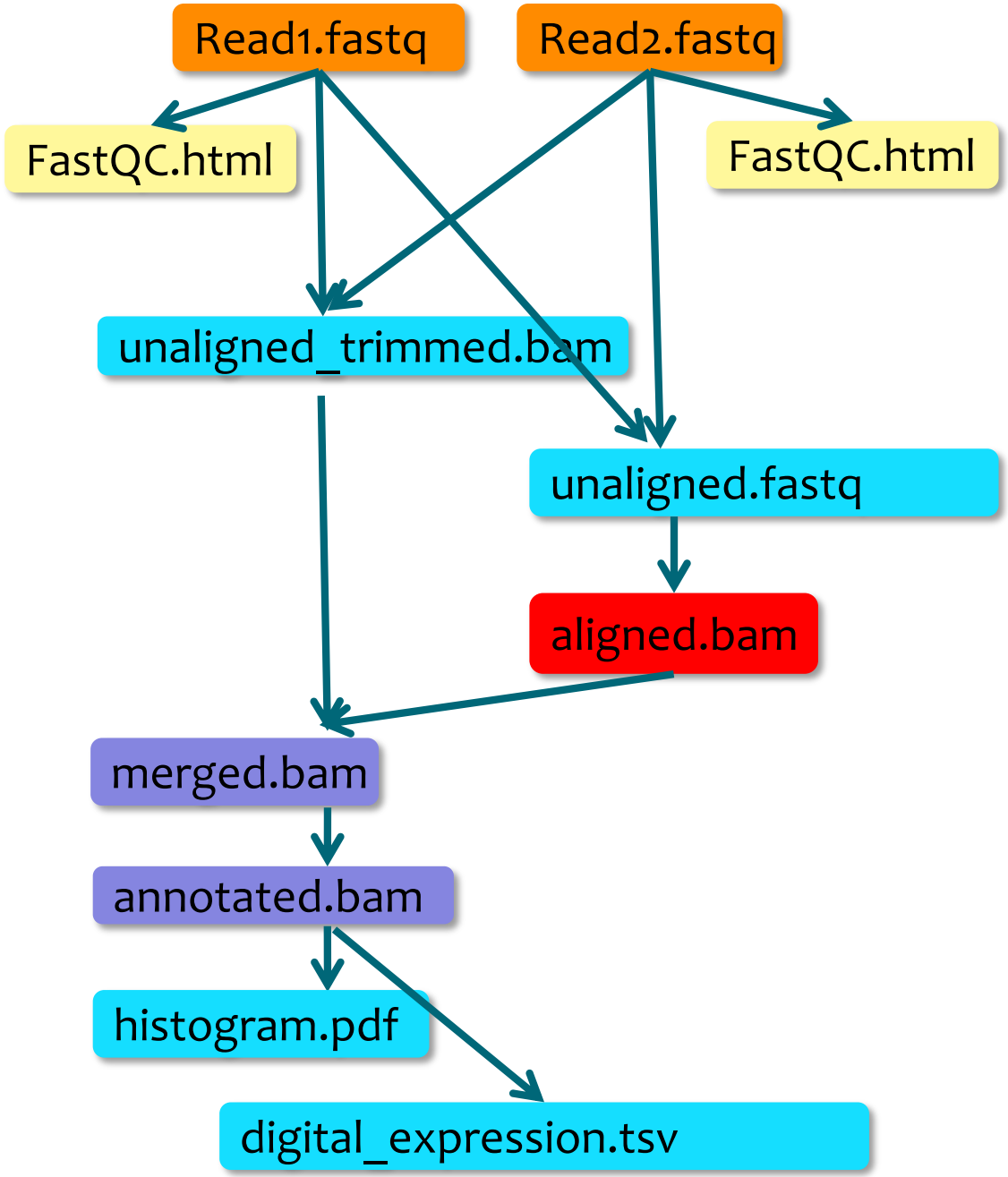
NOTE: "read" is a molecule here, which may or may not have the same/almost same UMI as another molecule



Filtering the data for DGE –huge number of reads

- What do you do if this happens?
  - Use the "minimum number of expressed genes per cell" as a requirement
  - Do you know how many cells to expect? Use that number!
  - Take a large number of cells and filter in the Seurat tools





Import files

Quality control / FASTQC

Preprocess:

- FASTQ to BAM
- Tag BAM
- Filter and trim BAM
- BAM to FASTQ
- Filter too short reads

-filter bad quality barcodes
-trim adapter sequences
-trim polyA tails

Aligner (STAR, HISAT)

Merge BAM

Combines the barcode tags and the alignment info

Tag read with gene names

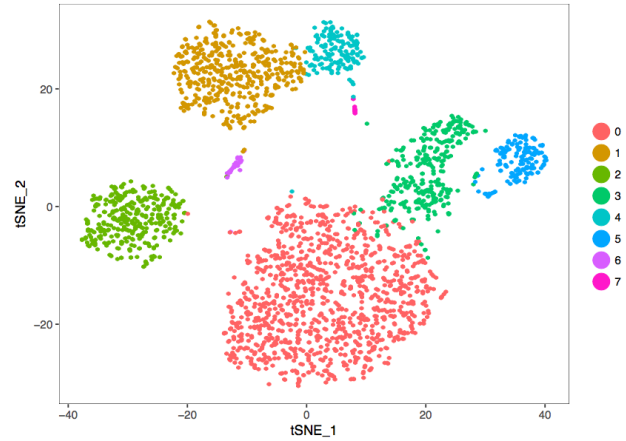
Estimate number of usable cells

Create digital expression matrix

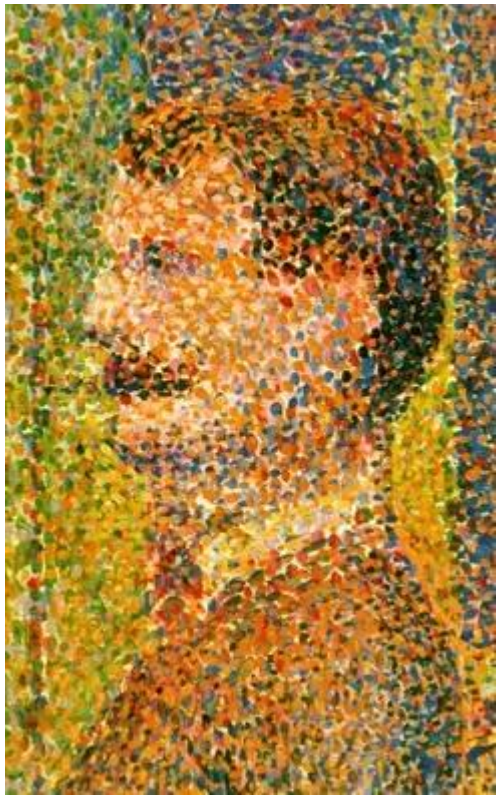
+ Detect bead synthesis errors

Clustering analysis with Seurat tools

Seurat http://satijalab.org/seurat



- Seurat combines dimensionality reduction and graph-based partitioning algorithms for unsupervised clustering of single cells.
- Summary of the steps:
  1. Identification of highly variable genes
  2. Linear dimensionality reduction (PCA, principal component analysis) on variable genes
  3. Determine significant principal components
  4. Graph based clustering to classify distinct groups of cells
  5. Non-linear dimensional reduction (t-SNE, t-Distributed Stochastic Neighbor Embedding) for cluster visualization
  6. Discovery of marker genes for the clusters, visualization, and downstream analysis



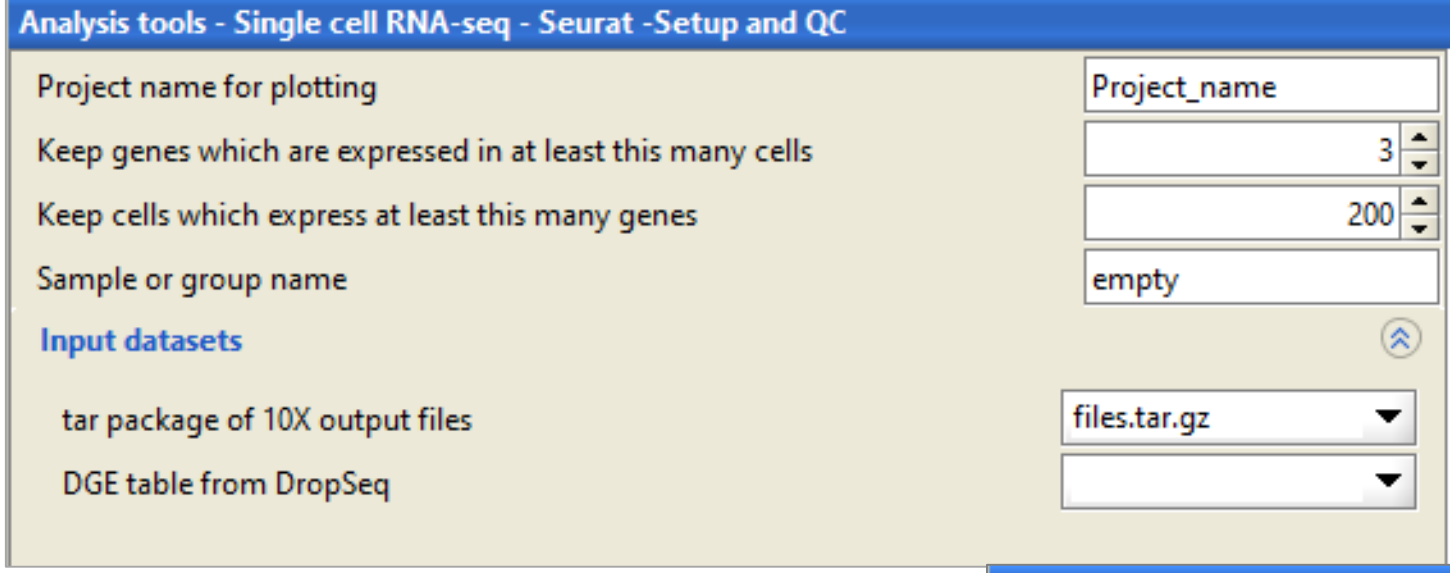
Clustering analysis with Seurat tools

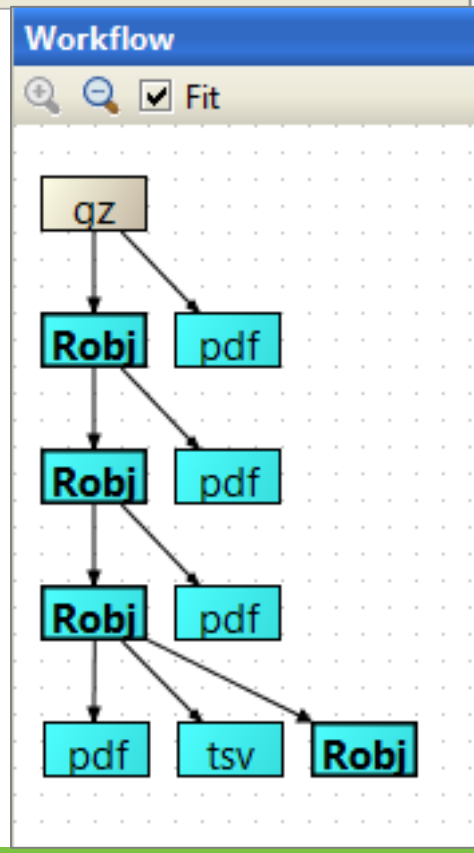
- Setup & preprocessing
- Quality control
- Filter cells
- Normalize expression values
- Regress unwanted sources of variation
- Detect variable genes
- Linear dimensional reduction (PCA)
- Determine statistically significant principle components (=identify the true dimensions of the data)
- Cluster the cells
- Non-linear dimensional reduction (tSNE)
- Find differentially expressed genes (biomarkers for the clusters)

- Setup & QC
- Filtering, regression & detection of variable genes
- Linear dimensional reduction (PCA)
- Clustering the cells
- Visualize biomarkers

Setting up a Seurat object

- Input file options
  - DGE matrix of DropSeq data
  - Tar package of 10X Genomics data
- Filtering
  - Keep genes which are detected in at least X cells
  - Keep cells where at least Y genes are detected
  - Some filtering might have happened already before, now with Seurat we can do more
- Give a name for the project (used in some plots)
- Sample or group name: if you have (two) samples you wish to compare
- Seurat-based tools use an R object (.Robj) to store data
  - it can't be opened in Chipster like normal files
    (can be exported from Chipster & imported to R though)

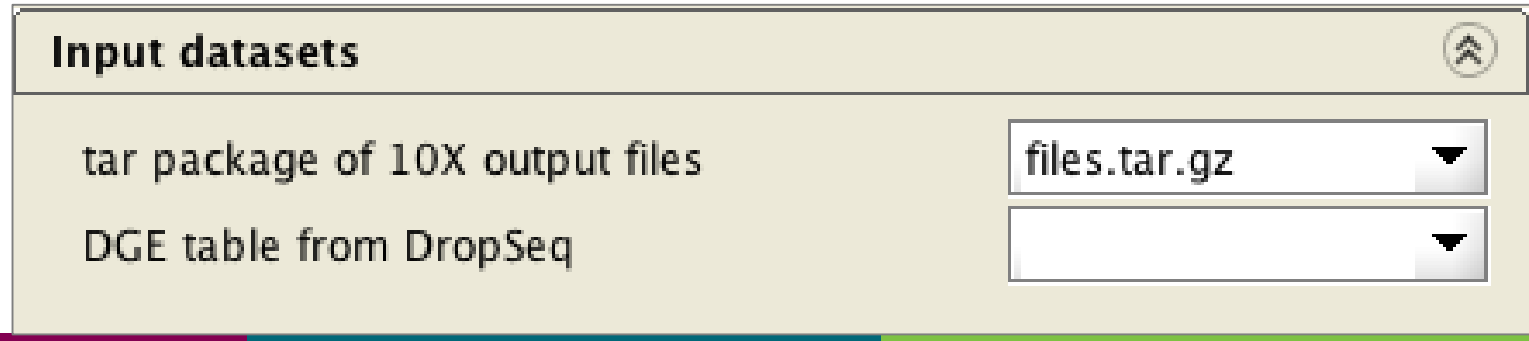




Two input file options

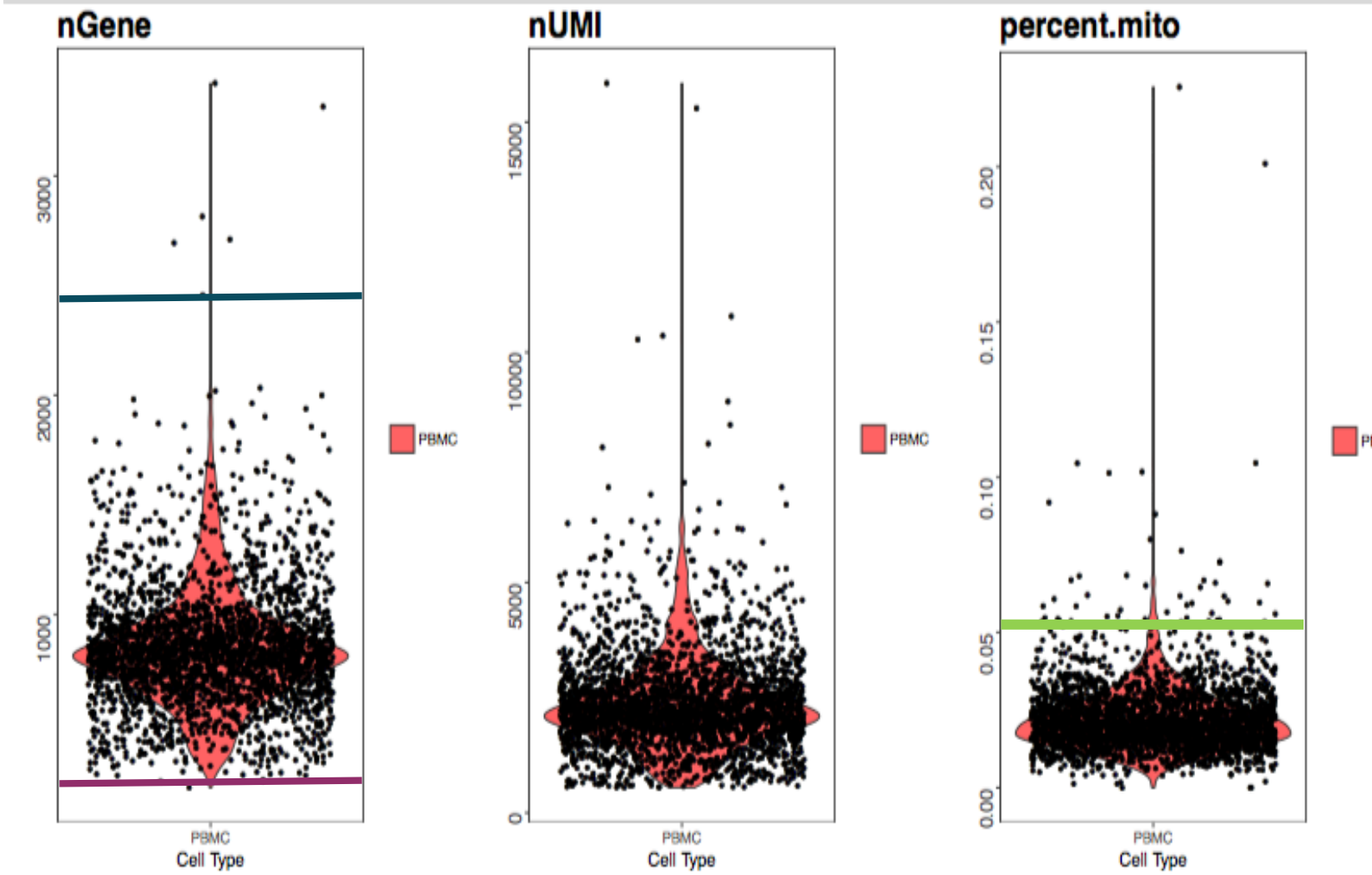
1. 10X Genomics output files
   - three files needed: barcodes.tsv, genes.tsv and matrix.mtx
   - make a tar package of these files and import it to Chipster
2. DGE matrix from the DropSeq tools
   - DGE matrix made in Chipster (like in the exercises), or import a ready-made DGE matrix

- Check that the input file is correctly assigned!



Quality control & filtering for empties, multiplets and broken cells

- Empties (no cell in droplet) → low gene count (<200)
- Multiplet (more than one cell in droplet) →large gene count (>2500)
- Broken cell in droplet → large percentage of mitochondrial transcripts (>5%)



A dot = a cell

Clustering analysis with Seurat tools

- Setup & preprocessing
- Quality control
- Filter cells
- Normalize expression values
- Regress unwanted sources of variation
- Detect variable genes
- Linear dimensional reduction (PCA)
- Determine statistically significant principle components (=identify the true dimensions of the data)
- Cluster the cells
- Non-linear dimensional reduction (tSNE)
- Find differentially expressed genes (biomarkers for the clusters)

- Setup & QC
- Filtering, regression & detection of variable genes
- Linear dimensional reduction (PCA)
- Clustering the cells
- Visualize biomarkers

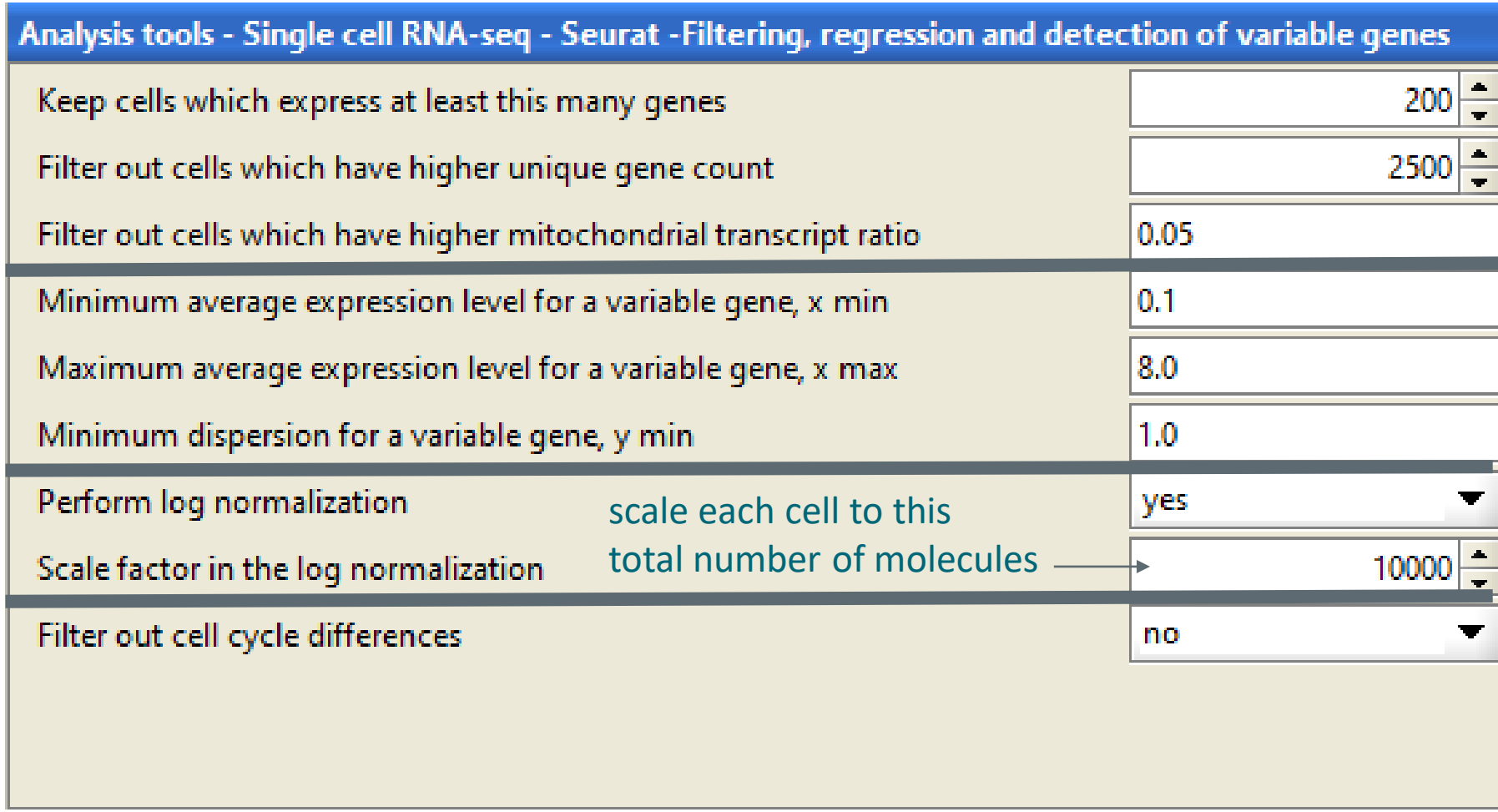
Filtering, regression & detection of variable genes

Filter outlier cells based on the plots from the previous tools

For selection of variable genes (plot after running this tool)

For normalization

For cell cycle phase regression



Normalizing expression values

- After removing unwanted cells from the dataset, the next step is to normalize the data.
- Global scaling normalization method "LogNormalize"
  - divide gene's expression value in a cell by the the total number of transcripts in that cell
  - multiply the ratio by a scale factor (10,000 by default). This scales each cell to this total number of transcripts
  - log-transform the result

Remove unwanted sources of variation

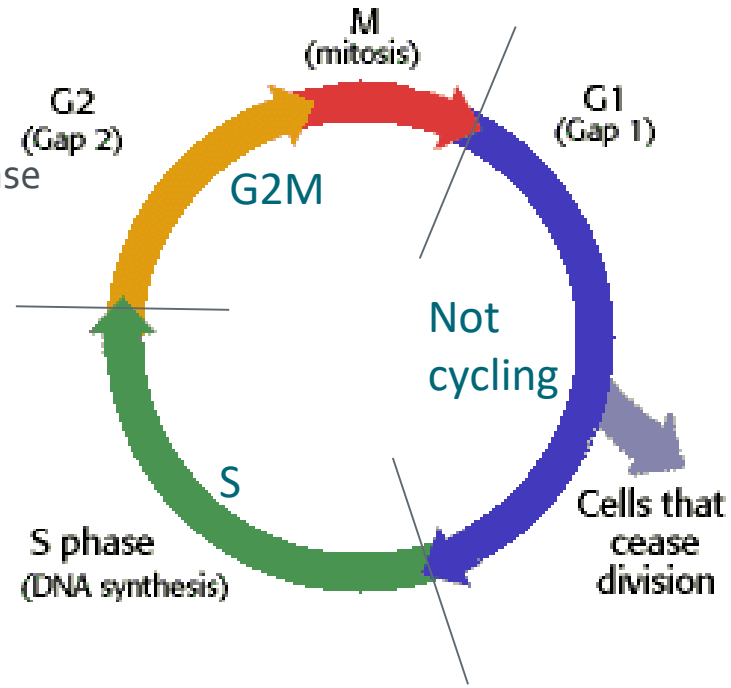
- Single cell data typically contains 'uninteresting' variation
  - technical noise
  - batch effects
  - cell cycle stage, etc
- Removing this variation improves downstream analysis
- Seurat constructs linear models to predict gene expression based on user-defined variables
  - batch, cell alignment rate, number of detected molecules per cell, mitochondrial transcript percentage
  - Seurat regresses the given variables individually against each gene, and the resulting residuals are scaled
  - scaled z-scored residuals of these models are used for dimensionality reduction and clustering
  - **In Chipster** the following effects are removed:
    - number of detected molecules per cell
    - mitochondrial transcript percentage
    - cell cycle stage (optional)

Mitigating the effects of cell cycle heterogeneity

1. Compute cell cycle phase scores for each cell based on its expression of G2/M and S phase marker genes (cells expressing neither → not cycling, G1)
   - Markers are well conserved across tissues and species

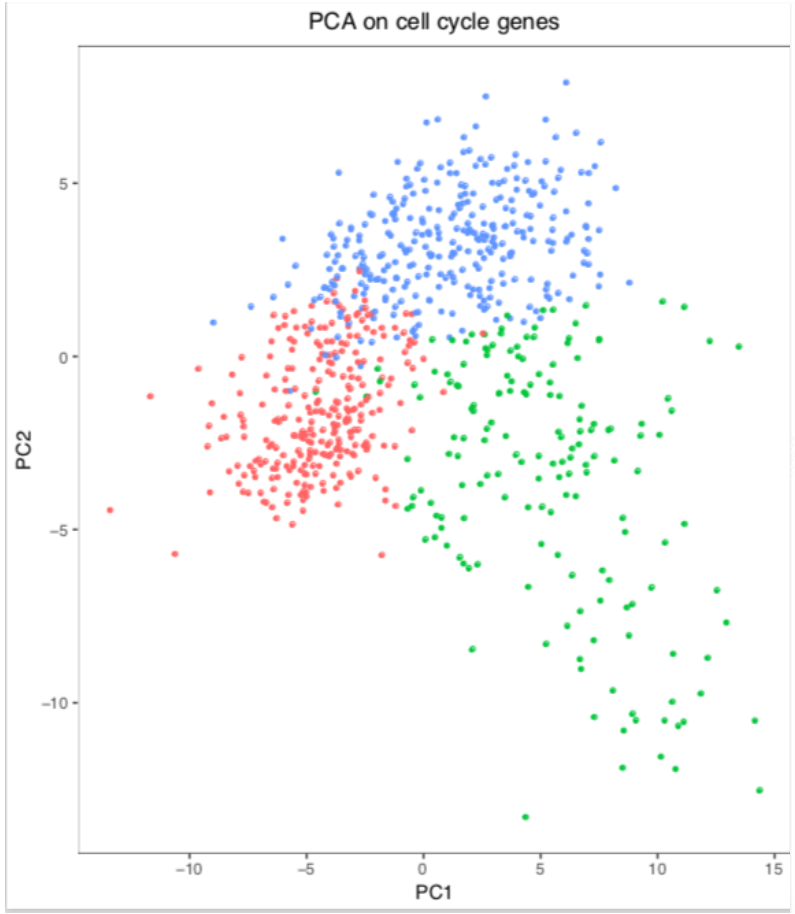
2. Model each gene's relationship between expression and the cell cycle score

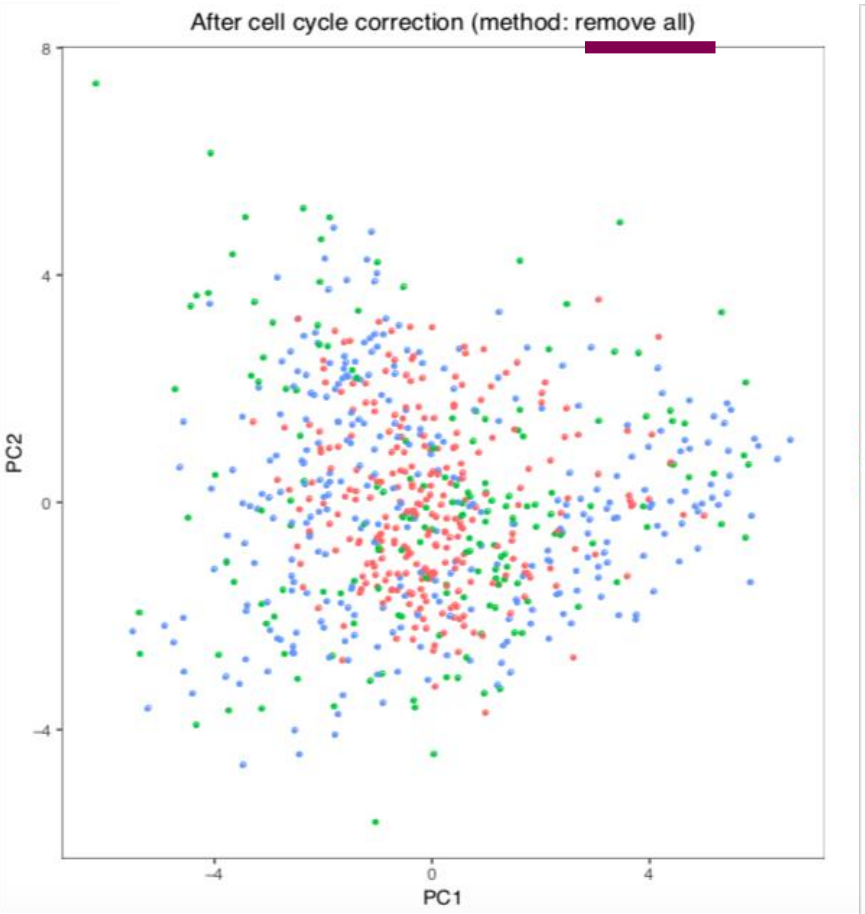
3. Regress out the variation caused by the different cell cycle stages
   A. Remove ALL signals associated with cell cycle stage, OR
   B. Remove the DIFFERENCE between the G2M and S phase scores
      - Preserves signals for non-cycling vs cycling cells, only differences in cell cycle phase amongst the dividing cells are removed. Recommended when studying differentiation processes

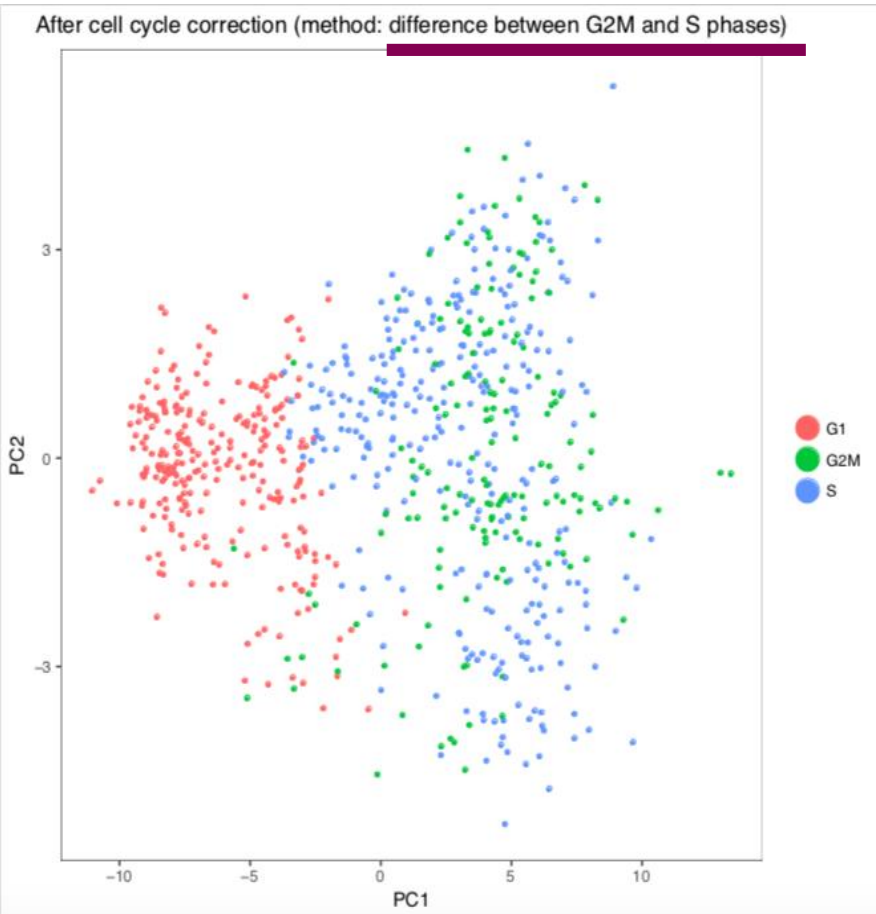


Regressing out the variation caused by different cell cycle stages

PCA on cell cycle genes (dot = cell, colors = phases)

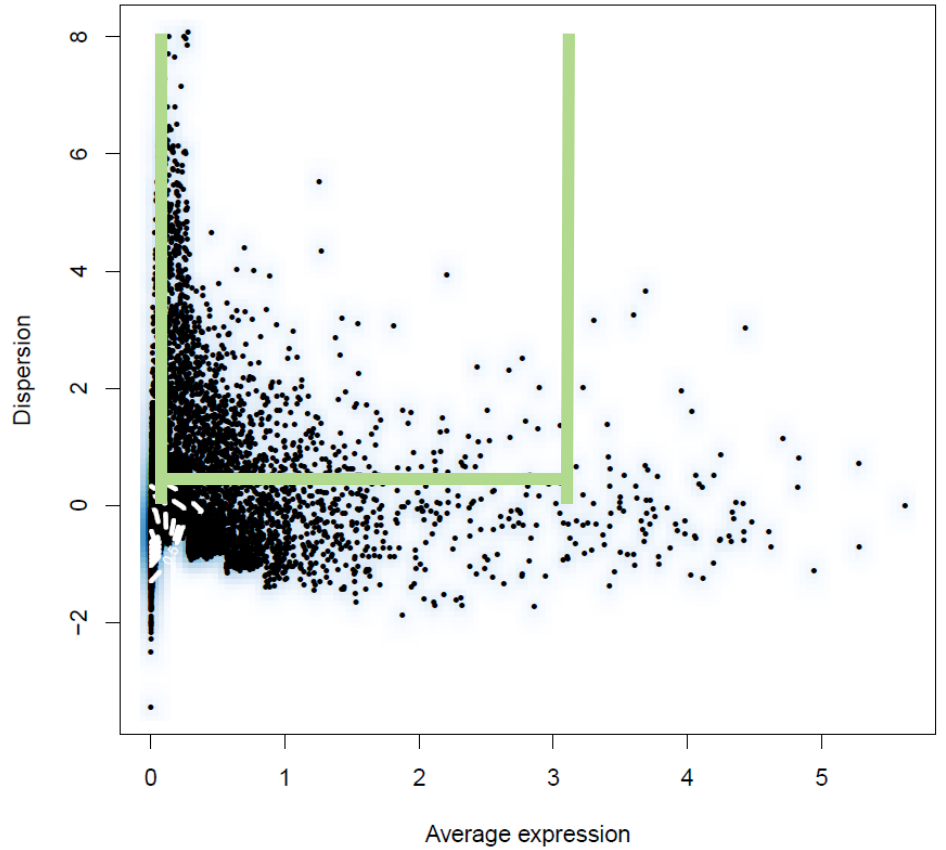






Detection of variable genes

- Downstream analysis focuses on highly variable genes
- Seurat finds them in the following way
  1. calculate the average expression and dispersion for each gene
     - Dispersion = log of variance/mean ratio
  2. place genes into 20 bins based on expression
  3. calculate a z-score for dispersion within each bin



- Check visual outliers in the dispersion plot
  - Default parameters are x = 0.1 - 8 and y>1. Settings may vary based on the data type, heterogeneity in the sample, and normalization strategy
  - these parameters are typical for UMI data that is normalized to a total of 10 000 molecules:

Minimum average expression level for a variable gene, x min	0.0125
Maximum average expression level for a variable gene, x max	3.0
Minimum dispersion for a variable gene, y min	0.5

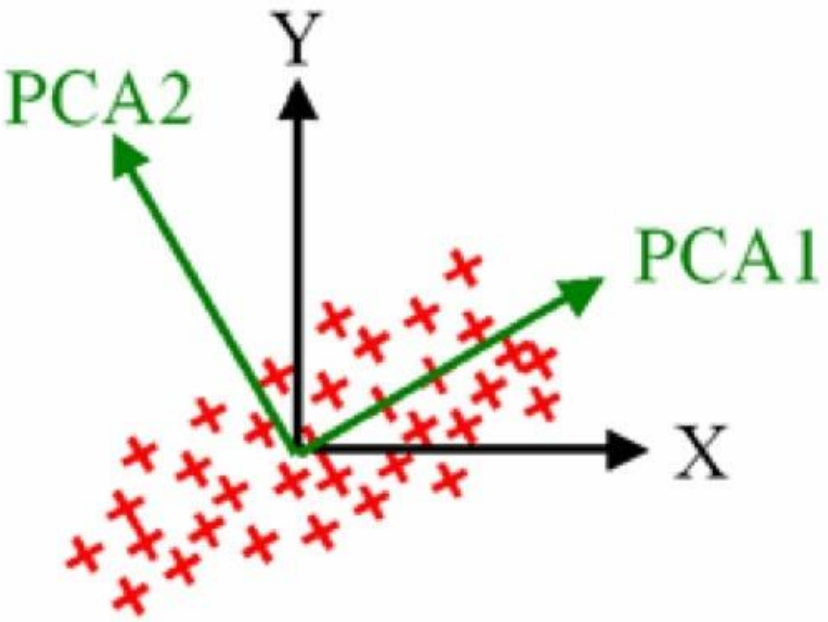
Clustering analysis with Seurat tools

- Setup & preprocessing
- Quality control
- Filter cells
- Normalize expression values
- Regress unwanted sources of variation
- Detect variable genes
- Linear dimensional reduction (PCA)
- Determine statistically significant principle components (=identify the true dimensions of the data)
- Cluster the cells
- Non-linear dimensional reduction (tSNE)
- Find differentially expressed genes (biomarkers for the clusters)

- Setup & QC
- Filtering, regression & detection of variable genes
- Linear dimensional reduction (PCA)
- Clustering the cells
- Visualize biomarkers

PCA

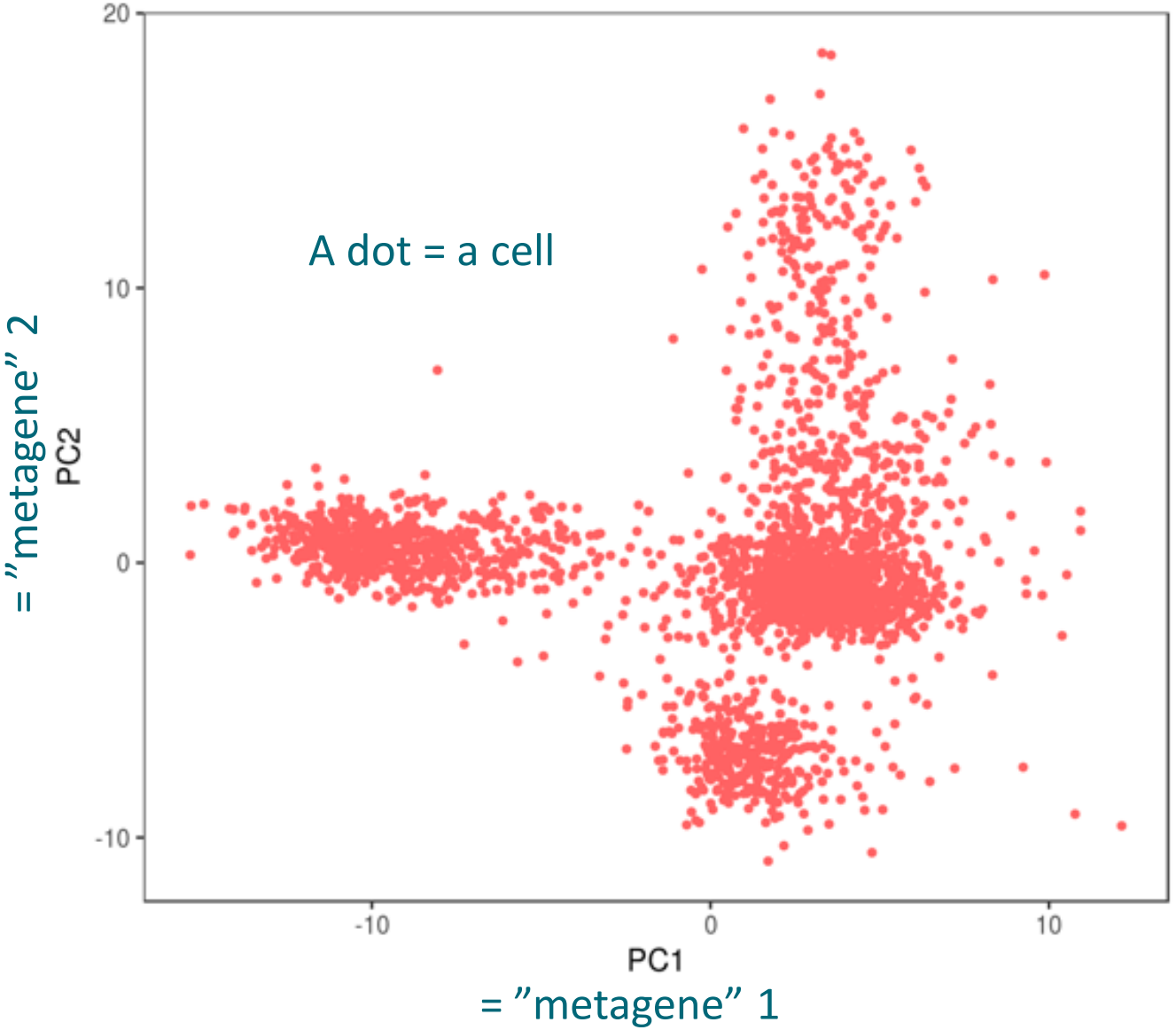
- PCA = Principal Component Analysis
- finds the principal components of data
- PCs = the directions where there is the most variance = the directions where the data is most spread out
- Why we use PCA here?
  - Clustering step is very tricky, due to all the technical and biological noise, and huge number of dimensions in the data
  - By reducing and selecting the dimensions, we get less noise and the data can be visualized



In our case instead of X and Y:
gene1, gene2, gene3… gene1838
= MANY dimensions!

Linear dimensionality reduction (PCA) on variable genes

- Reduce the numerous, possibly correlating variables (= counts for each gene) into a smaller number of linearly uncorrelated dimensions (= principal components)
- Essentially, each PC represents a robust 'metagene' — a linear combination of hundreds to thousands of individual transcripts

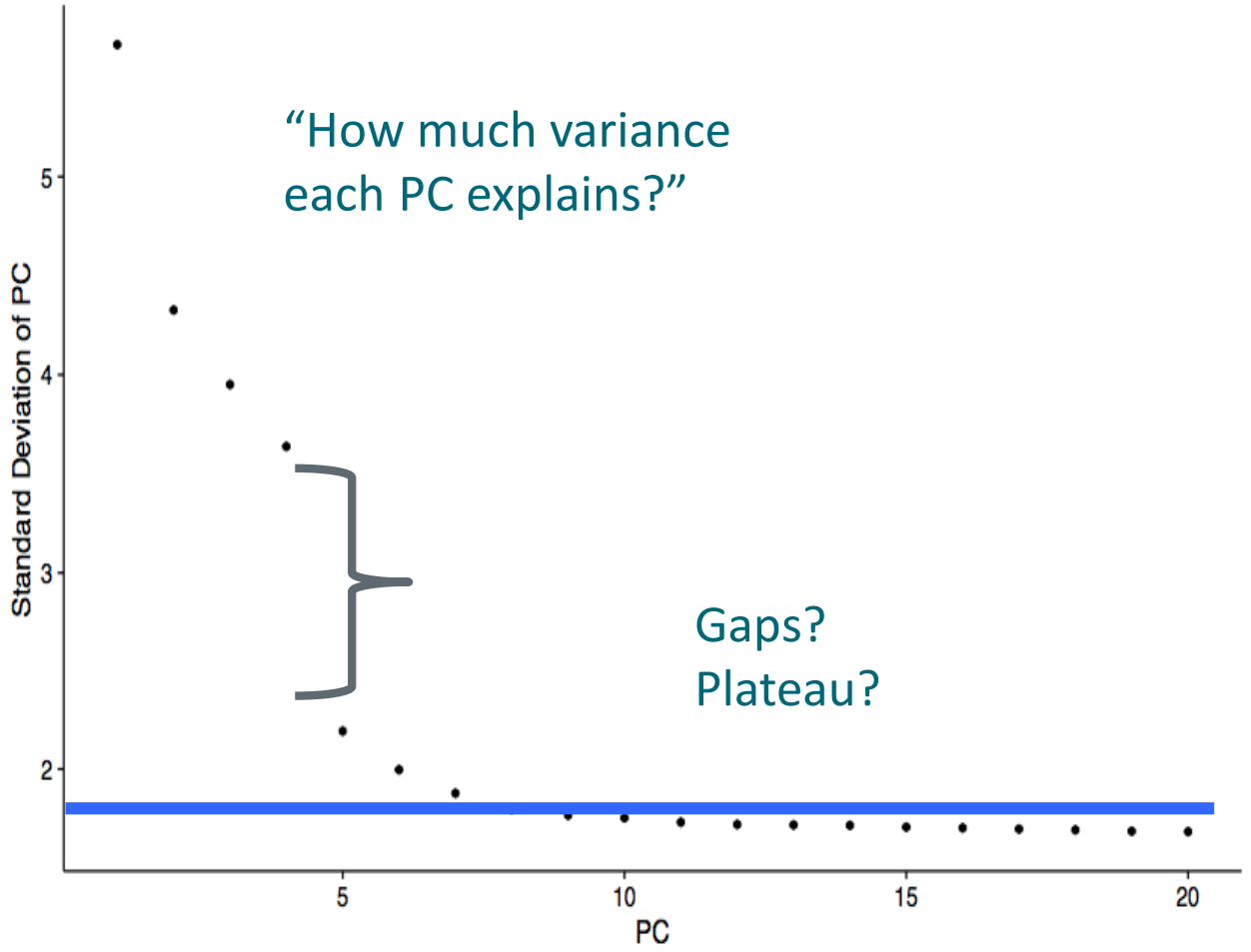


Determine significant principal components

- A key step to this clustering approach involves selecting a set of principal components (PCs) for downstream clustering analysis
- However, estimating the true dimensionality of a dataset is a challenging and common problem in machine learning.
- The tool provides a couple of plots to aid in this:
  - Elbow plot
  - PCHeatmap

Determine significant principal components 1: Elbow plot

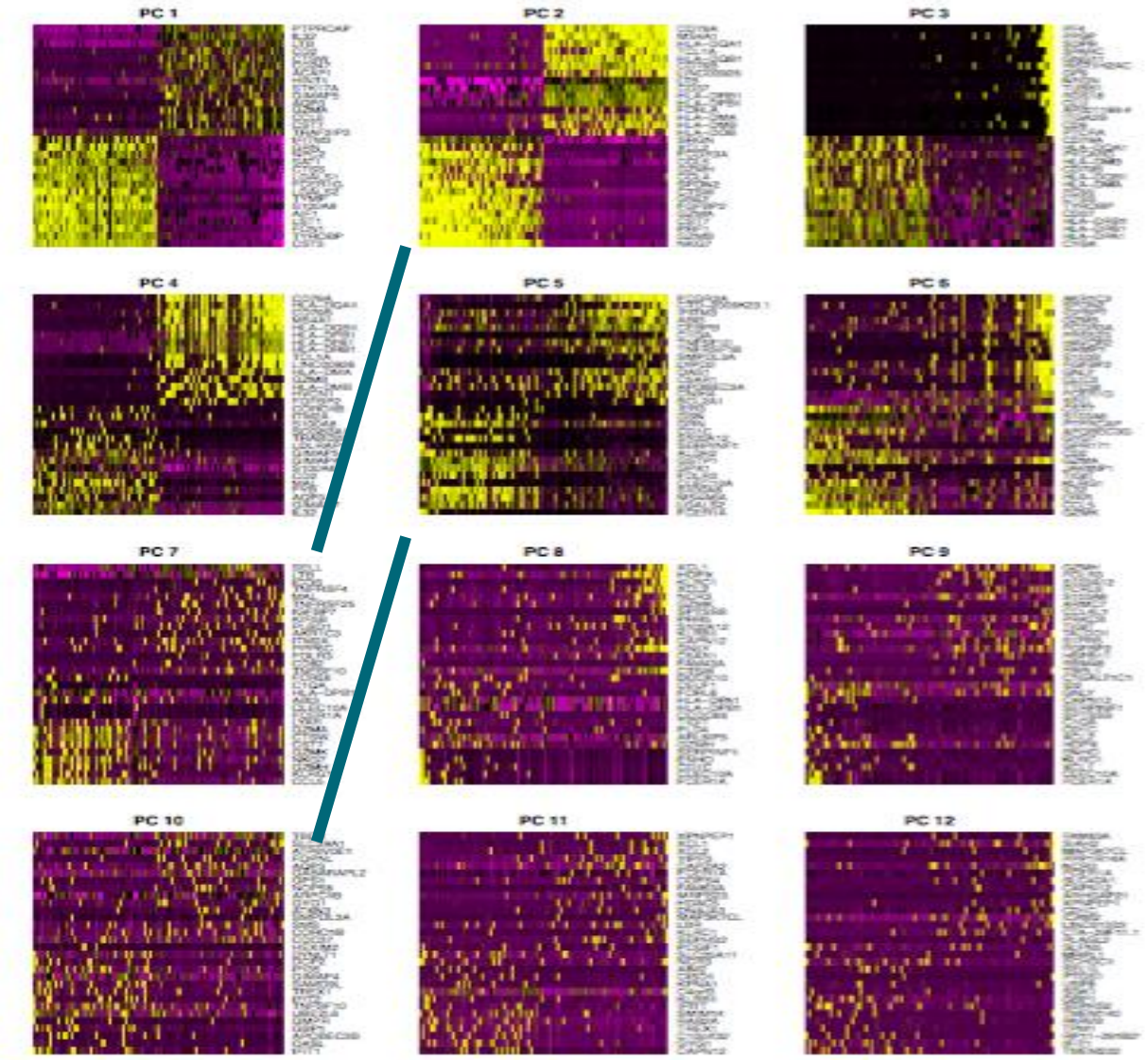
- The **elbow** in the plot tends to reflect a transition from informative PCs to those that explain comparatively little variance.



Determine significant principal components 2: PCHeatmap

- Displays the **extremes** across **both genes and cells**, and can be useful to help exclude PCs that may be driven primarily by ribosomal/mitochondrial or cell cycle genes.

"Is there still a difference between the extremes?"



Clustering analysis of with Seurat tools

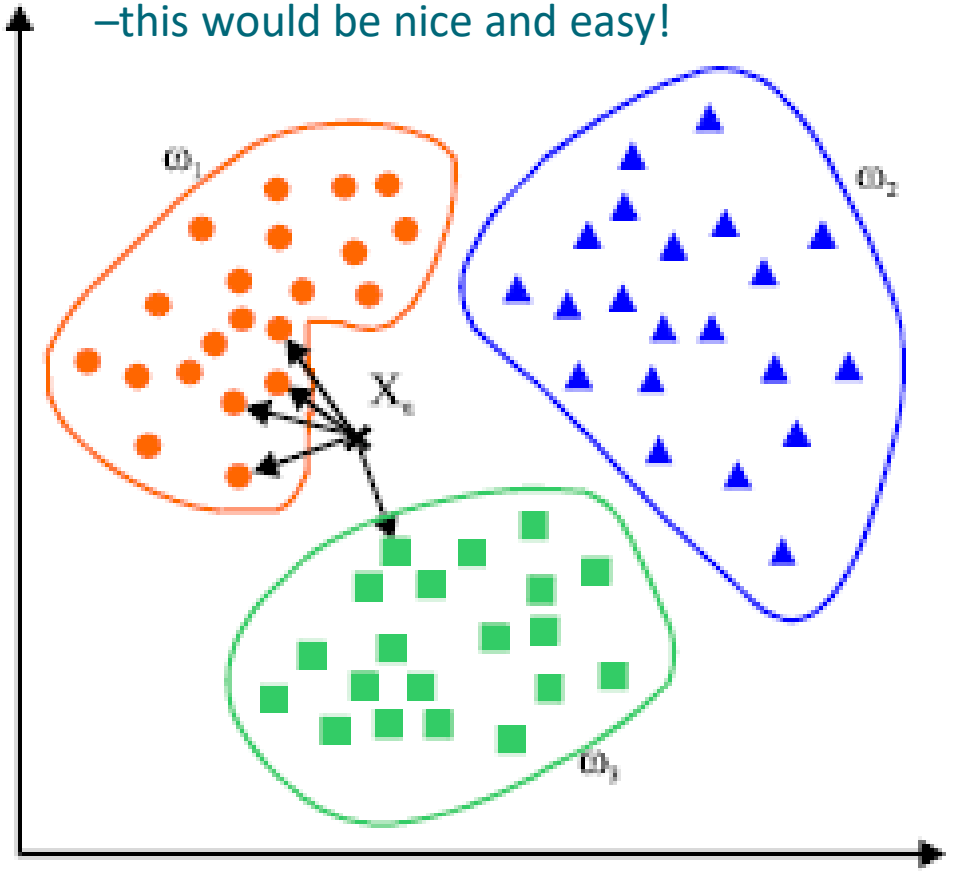
- Setup & preprocessing
- Quality control
- Filter cells
- Normalize expression values
- Regress unwanted sources of variation
- Detect variable genes
- Linear dimensional reduction (PCA)
- Determine statistically significant principle components (=identify the true dimensions of the data)
- Cluster the cells
- Non-linear dimensional reduction (tSNE)
- Find differentially expressed genes (biomarkers for the clusters)

- Setup & QC
- Filtering, regression & detection of variable genes
- Linear dimensional reduction (PCA)
- Clustering the cells
- Visualize biomarkers

Clustering - why is it so tricky?

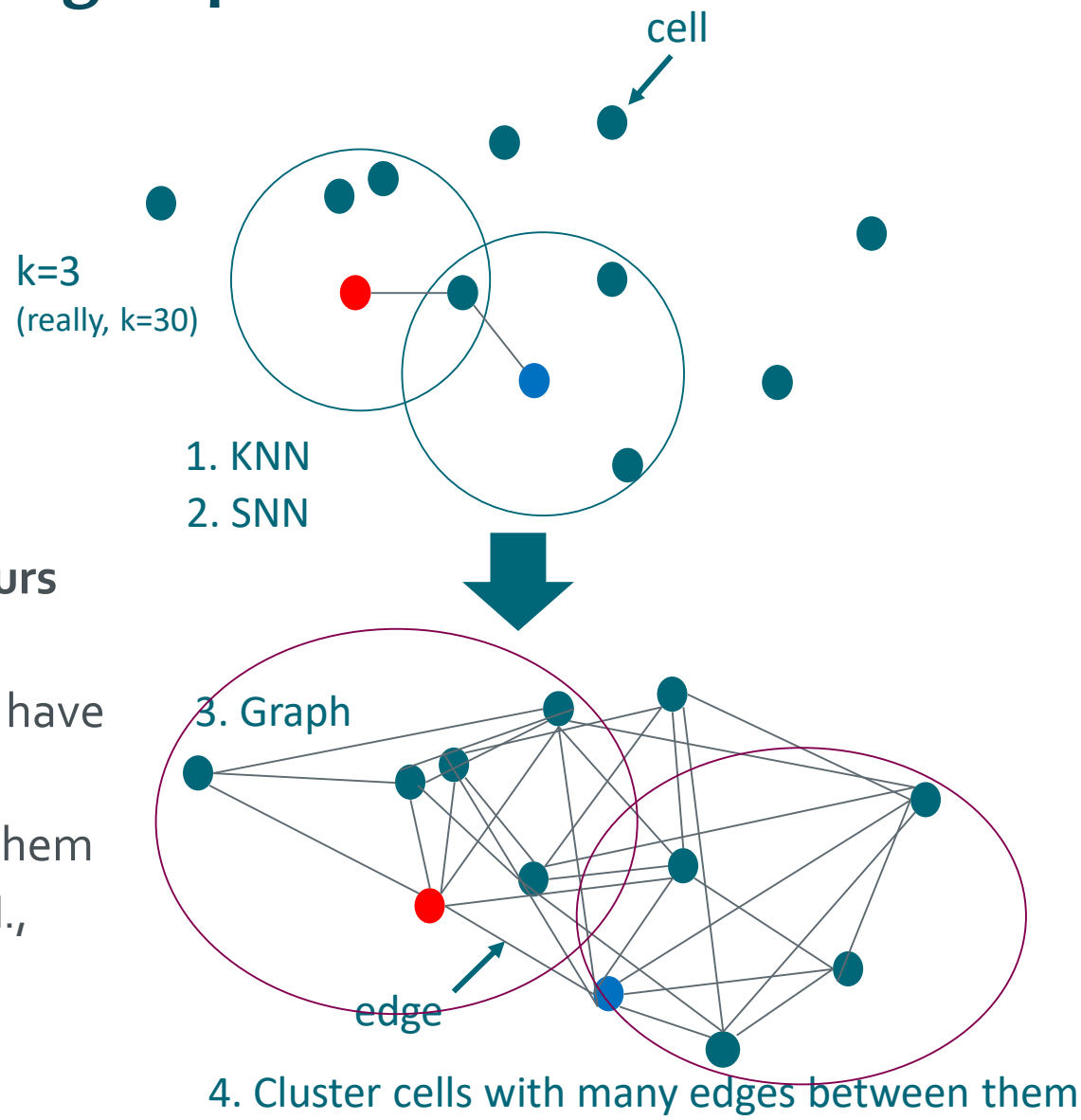
- Need to use **unsupervised** methods (we don't know beforehand, how many clusters there are)
- Our data is big and complex:
  - Lots of cells
  - Lots of dimensions (=genes)
  - Lots of noise (both technical and biological)
- …which is why:
  - The algorithm is a bit tricky, and
  - We reduce and select the dimensions to use in clustering (= we do the PCA and select X components to use, instead of using all thousands of genes)

KNN in 2D with known clusters –this would be nice and easy!



Graph based clustering to classify distinct groups of cells

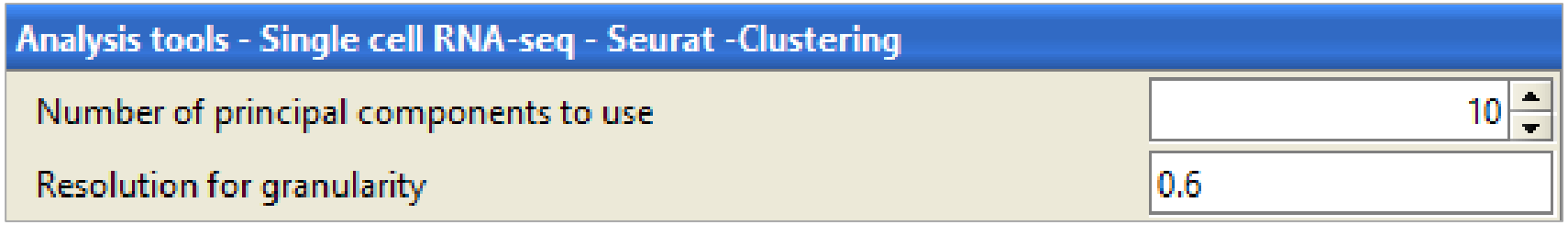
- Seurat clustering = similar to
  - **SNN-Cliq** (C. Xu and Su, Bioinformatics 2015) and
  - **PhenoGraph** (Levine et al. Cell 2015)
- ...which are graph based methods:
  1. Identify **k-nearest neighbours** of each cell
     - Distance measure: Euclidean + Jaccard distance
  2. Calculate the number of **Shared Nearest Neighbours (SNN)** between each pair of cells
  3. Build the graph: add an **edge** between cells, if they have at least one SNN
  4. Clusters: group of cells with many edges between them
     - **Smart Local Moving algorithm** (SLM, Blondel et al., Journal of Statistical Mechanics)



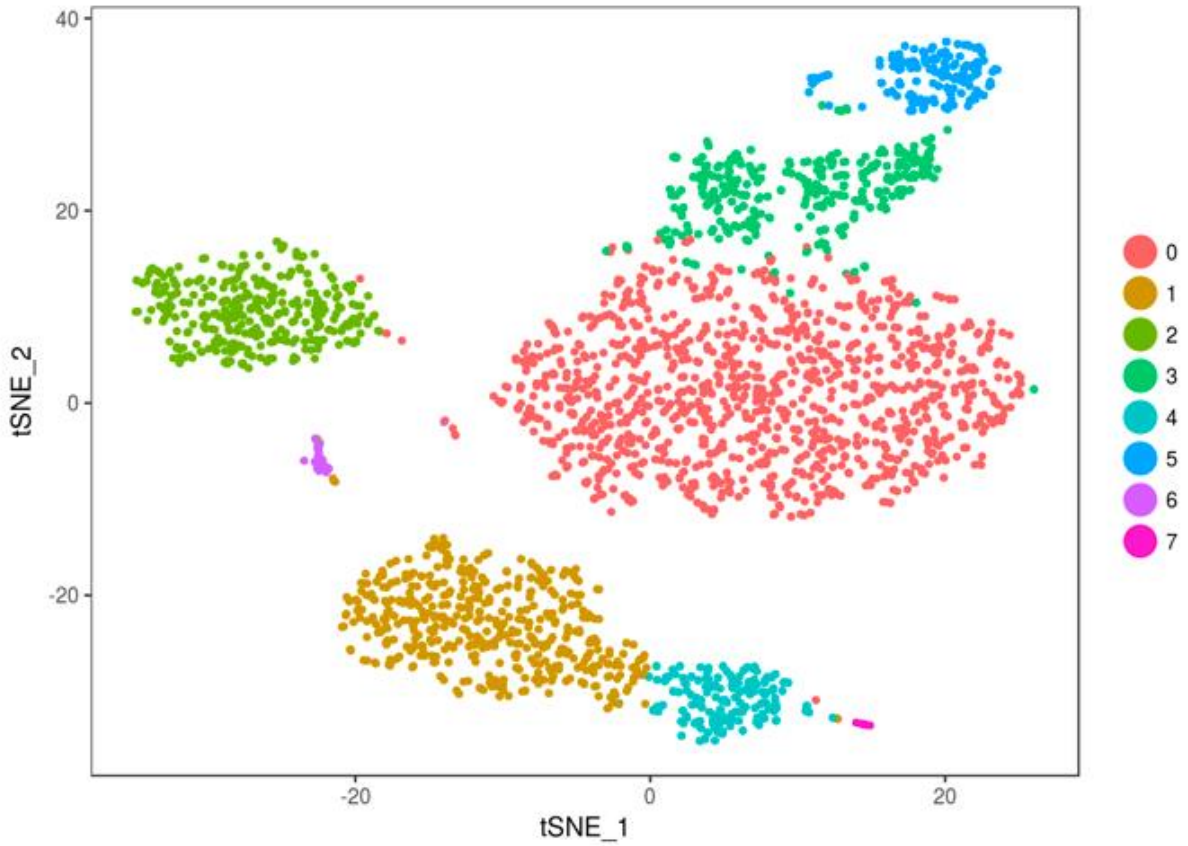
Clustering parameters

FindClusters(object, genes.use = NULL, reduction.type = "pca", **dims.use = NULL**, k.param = 30, k.scale = 25, plot.SNN = FALSE, prune.SNN = 1/15, print.output = TRUE, distance.matrix = NULL, save.SNN = FALSE, reuse.SNN = FALSE, force.recalc = FALSE, modularity.fxn = 1, **resolution = 0.8**, algorithm = 1, n.start = 100, n.iter = 10, random.seed = 0, temp.file.location = NULL)

- Lots of parameters in the original tool…
- 2 in Chipster atm:
  - **Number of principal components to use** (dims.use)
  - **Resolution for granularity** (resolution):
    - Increased values lead to a greater number of clusters.
    - Values 0.6-1.2 typically return good results for single cell datasets of around 3K cells
    - Optimal resolution often increases for larger datasets
    - (If you get very few or very many clusters, try adjusting)



Non-linear dimensional reduction (t-SNE) for cluster visualization



What am I looking for?
Check if the coloring (= the clustering result) matches to what you see (= how tSNE sees the data).

- t-SNE = t-distributed Stochastic Neighbor Embedding
  - Non-linear algorithm, different transformations to different regions
- Why do we now use t-SNE, why not the good old PCA?
  - We give the selected PCA "metagenes" to t-SNE
  - t-SNE is more faithful to the original data
  - Takes into account non-linear relationships
  - PCA can find clusters too, but t-SNE does just that – it reduces the dimensions so that the clusters become visible
- Good text about reading t-SNE's: https://distill.pub/2016/misread-tsne/

Clustering analysis with Seurat tools

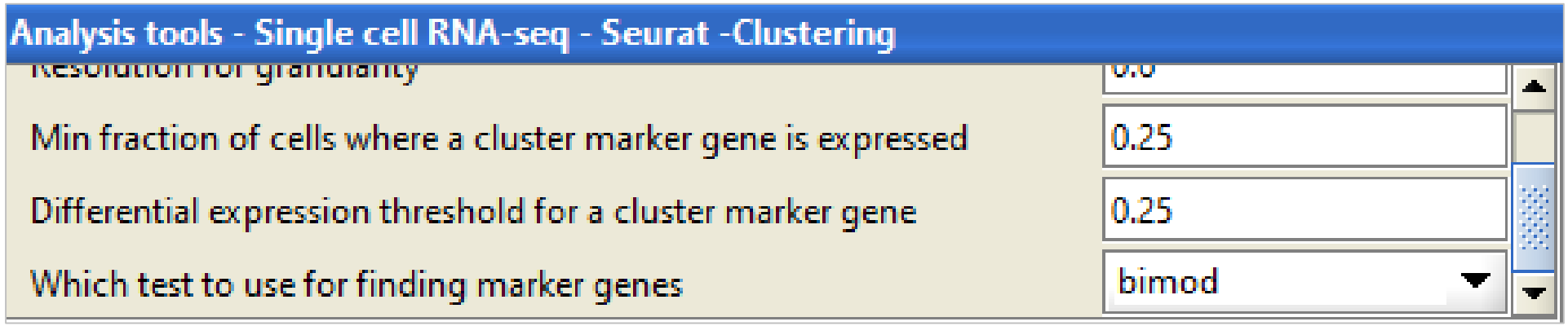
- Setup & preprocessing
- Quality control
- Filter cells
- Normalize expression values
- Regress unwanted sources of variation
- Detect variable genes
- Linear dimensional reduction (PCA)
- Determine statistically significant principle components (=identify the true dimensions of the data)
- Cluster the cells
- Non-linear dimensional reduction (tSNE)
- Find differentially expressed genes (biomarkers for the clusters)

- Setup & QC
- Filtering, regression & detection of variable genes
- Linear dimensional reduction (PCA)
- Clustering the cells
- Visualize biomarkers

Finding differentially expressed genes (biomarkers for the clusters)

- Parameters:
  - **min.pct** : requires a gene to be detected at least THIS minimum percentage in either of the two groups of cells (default: 0.25)
  - **thresh.test** : requires a gene to be differentially expressed (on average) by THIS amount between the two groups (default: 0.25)
  - You can set both of these to zero, but with a dramatic increase in time (this will test a large number of genes that are unlikely to be highly discriminatory)
  - Options for tests: bimod, roc, Students t-test, Tobit-test, Poisson, negative-binomial distribution

NOTE:
We compare now a cluster to all other cells.
So for example cluster 1 vs all others.



Markers for a particular cluster

- As a result, you will get a big table with all the differentially expressed genes to all the clusters in the data
- You can filter the result list to get only the biomarkers for a certain cluster:

Utilities / Filter table by column value

Column to filter by	cluster
Does the first column have a title	no
Cutoff	2.0
Filtering criteria	equal-to

= pick those rows from the table where the value in ”cluster” column = 2

=differentially expressed genes for cluster 2

Showing 3933 rows of 3933 and all 8 columns

	p_val	p_val_adj	avg_logFC	cluster
DDX5.3	9.772371e-03	1.000000e+00	-0.3667870	6
PSME2.5	9.751363e-03	1.000000e+00	-0.8285587	7
NDUFA4.2	9.530572e-03	1.000000e+00	0.4087960	7
YWHAZ	9.520594e-03	1.000000e+00	0.3671280	7
S1PR4.2	9.510319e-03	1.000000e+00	-0.5120146	6
IFITM2.4	9.188401e-03	1.000000e+00	-0.9508010	7
SEPT6	9.159823e-03	1.000000e+00	0.2989993	5

Showing 359 rows of 359 and all 8 columns

	p_val	p_val_adj	avg_logFC	cluster
CD79A	0	0	2.9778519	2
CD74.1	0	0	2.027942	2
CD79B.2	3.821947e-299	5.241418e-295	2.4006436	2
HLA-DRA.2	4.099945e-294	5.622664e-290	1.9183262	2
MS4A1	4.192323e-283	5.749351e-279	2.3376466	2
HLA-DQA1.1	3.215761e-233	4.410095e-229	2.1266067	2
HLA-DQB1.2	2.778244e-225	3.810084e-221	2.1392801	2

Differentially expressed genes, looking at the table

- You can also filter the table based on p-values and fold changes
- p_val = *p*-value
  - A small *p*-value (typically ≤ 0.05) indicates strong evidence
  - "how likely this gene is appearing as differentially expressed just by chance"
- p_val_ajd = adjusted *p*-value
  - Multiple testing correction performed <- use this
  - We repeat the same statistical test many many times on the different genes → likelihood of getting "hits" by chance increases, so we need to correct.
- logFC = log fold change
  - Log2 ratio between expression changes:
  - logFC= $log_2 \frac{expression\ in\ cluster\ X}{expression\ in\ other\ clusters}$

Showing 359 rows of 359 and all 8 columns

	p_val	p_val_adj	avg_logFC	cluster
CD79A	0	0	2.9778519	2
CD74.1	0	0	2.027942	2
CD79B.2	3.821947e-299	5.241418e-295	2.4006436	2
HLA-DRA.2	4.099945e-294	5.622664e-290	1.9183262	2
MS4A1	4.192323e-283	5.749351e-279	2.3376466	2
HLA-DQA1.1	3.215761e-233	4.410095e-229	2.1266067	2
HLA-DQB1.2	2.778244e-225	3.810084e-221	2.1392801	2

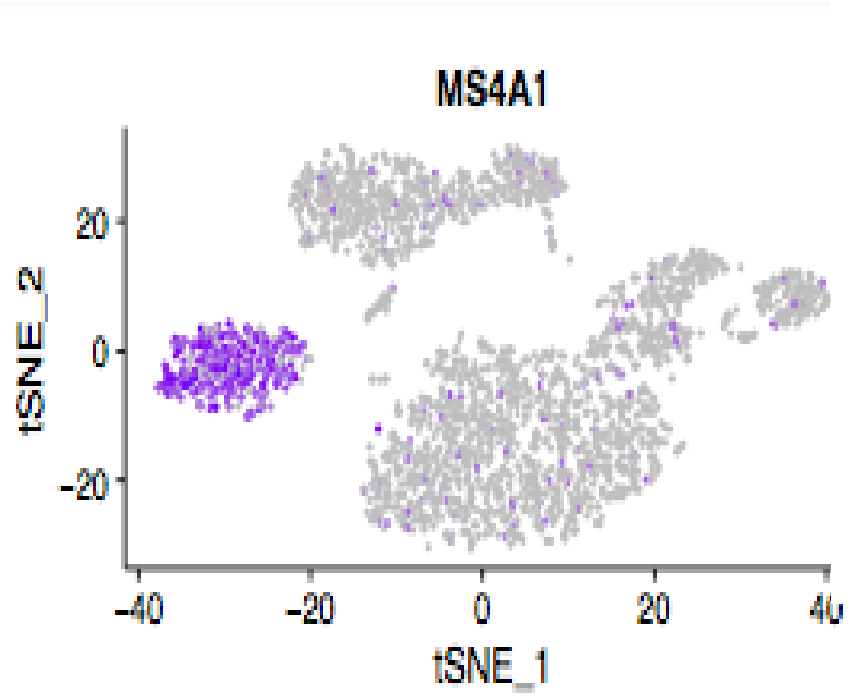
Clustering analysis of with Seurat tools

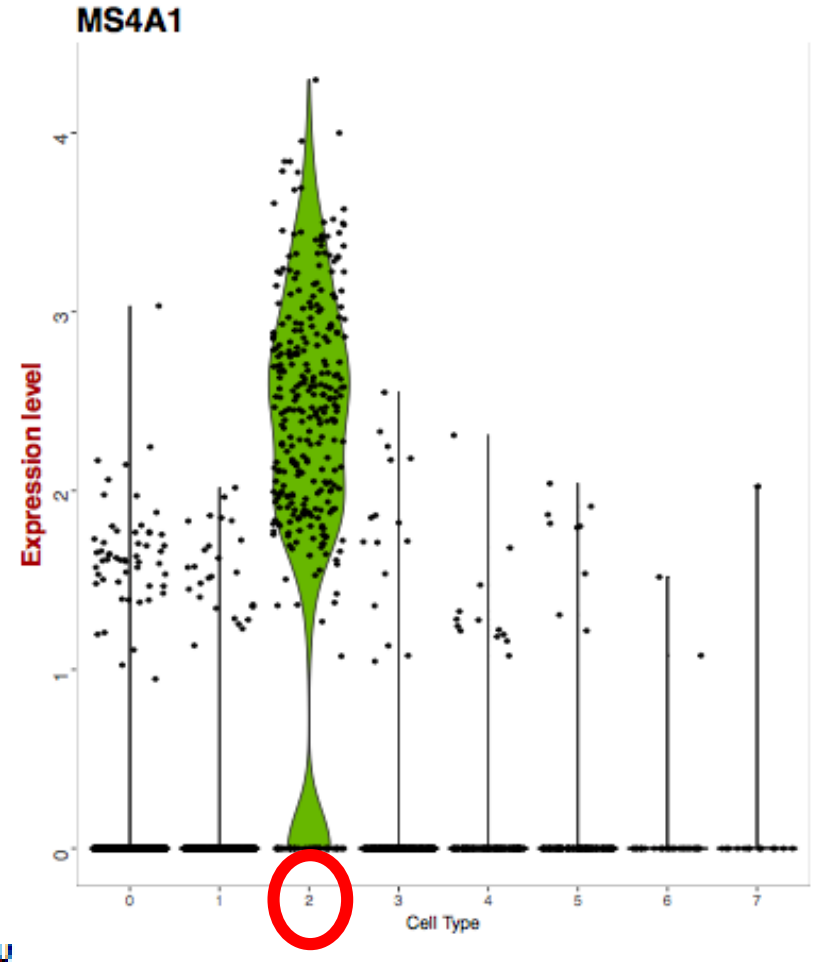
- Setup & preprocessing
- Quality control
- Filter cells
- Normalize expression values
- Regress unwanted sources of variation
- Detect variable genes
- Linear dimensional reduction (PCA)
- Determine statistically significant principle components (=identify the true dimensions of the data)
- Cluster the cells
- Non-linear dimensional reduction (tSNE)
- Find differentially expressed genes (biomarkers for the clusters)

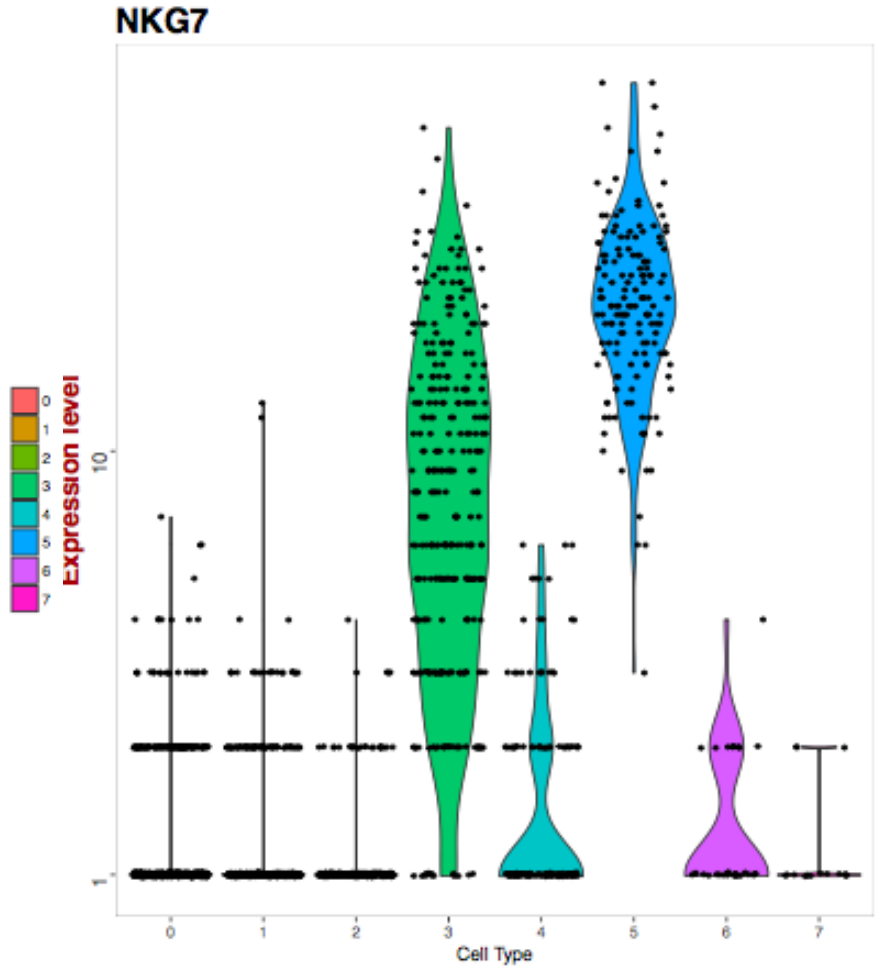
- Setup & QC
- Filtering, regression & detection of variable genes
- Linear dimensional reduction (PCA)
- Clustering the cells
- Visualize biomarkers

Visualize biomarkers

- Select a marker gene from the lists







Integrated analysis of two samples

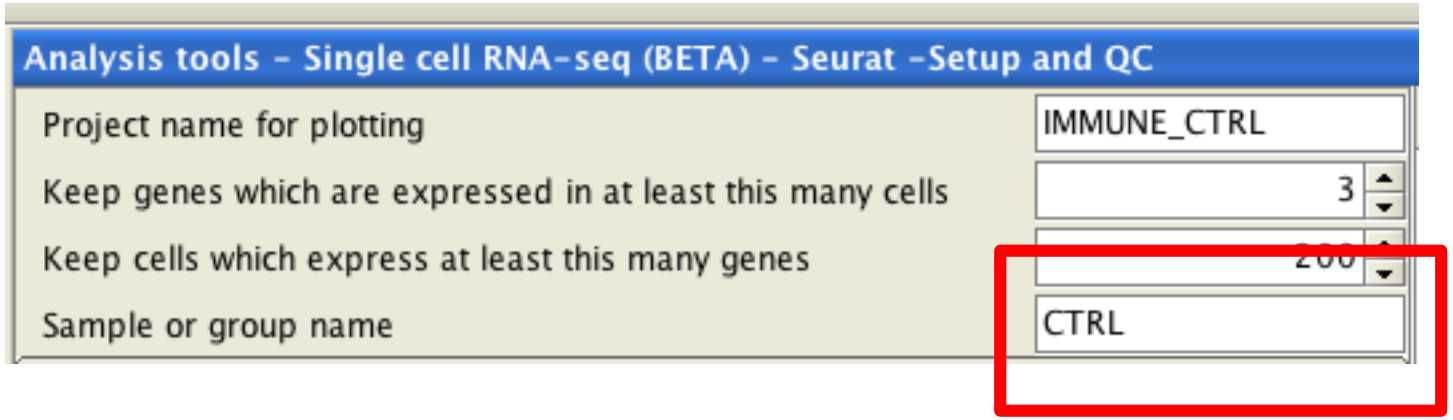
- We wish to:
  - Identify cell types that are present in both datasets
  - Obtain cell type markers that are conserved in both control and stimulated cells
  - Compare the datasets to find cell-type specific responses to stimulation
- Steps:
  - Setup + QC & Filtering steps
  - **Combine samples & perform CCA**
  - Integrated analysis (**aligning**, clustering, tSNE)
  - Find conserved biomarkers for clusters
  - Find differentially expressed genes between samples, within clusters
  - Visualize some interesting genes

Our example data:

- From the Seurat tutorial: https://satijalab.org/seurat/immune_alignment.html
- Two samples of peripheral blood mononuclear cells (PBMCs):
  - Controls (**CTRL**)
  - Treated cells (**STIM**), stimulated with interferon beta
- → clustering based on
  1. Cell types
  2. Treatment status

Integrated analysis: Setup, QC, filtering

- Same as before, just remember to name the samples in the setup tool
  - Good names: CTRL, TREAT, ctrl, treat, ctrl_1, treat_2
  - Bad name: control 1
- Perform the Setup+QC & Filtering steps separately for the two samples

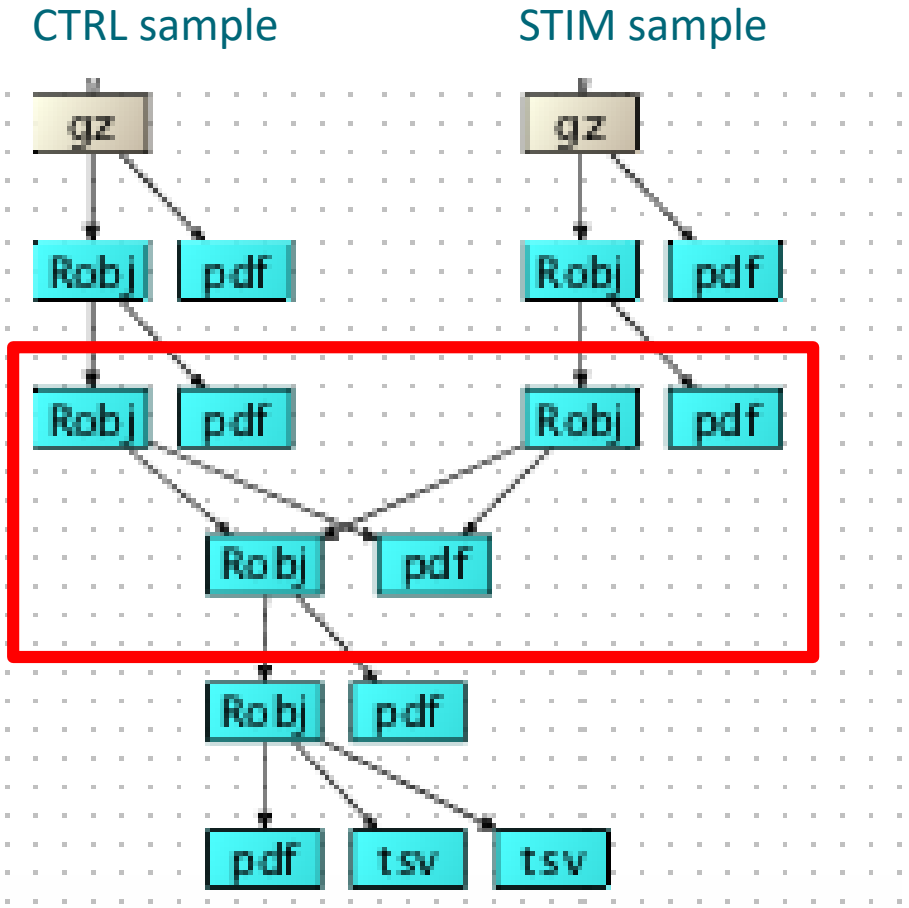


Integrated analysis of two samples

- Steps:
  - Setup + QC & Filtering steps
  - **Combine samples & perform CCA**
  - Integrated analysis (aligning, clustering, tSNE)
  - Find conserved biomarkers for clusters
  - Find differentially expressed genes between samples, within clusters
  - Visualize some interesting genes

Integrated analysis: Combine samples & perform CCA (1/3)

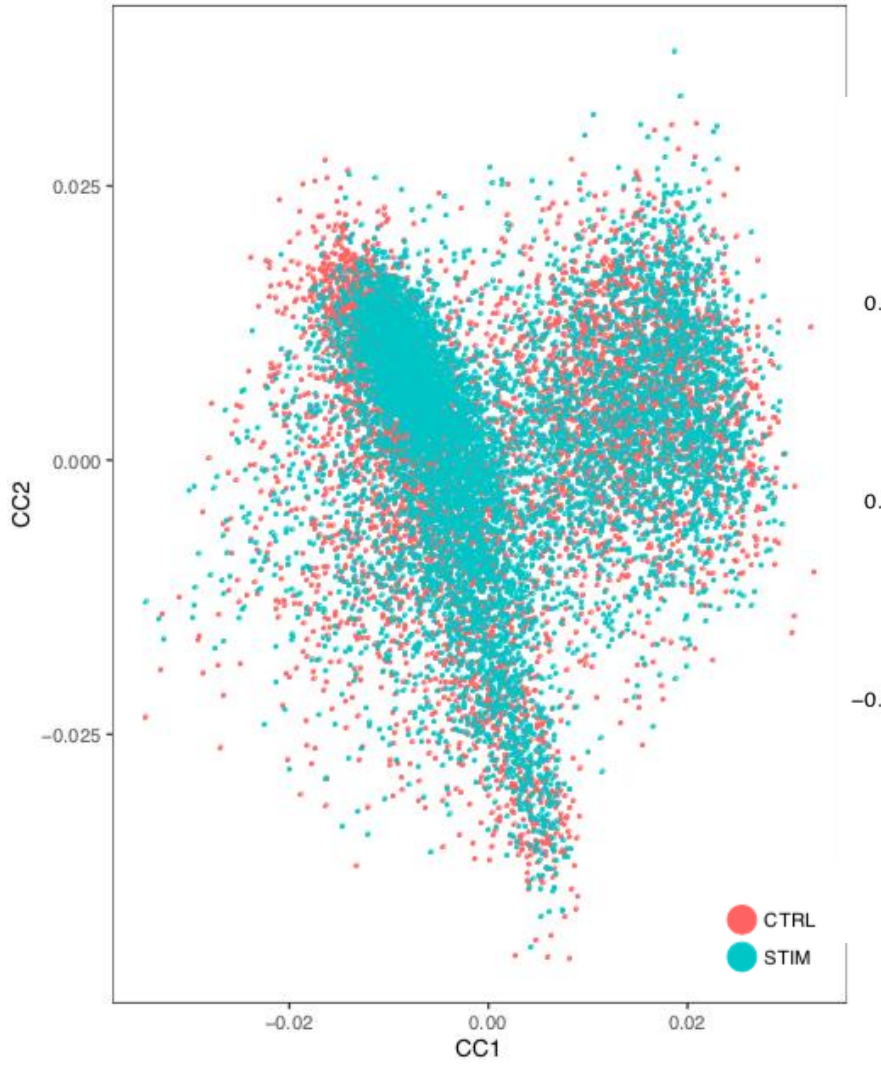
- Choose the two samples and run the tool **Combine two samples and perform CCA**
- The tool takes the union of the top 1000 genes with the highest dispersion (=var/mean) from both datasets, and performs a canonical correlation analysis (CCA**) to identify common sources of variation between the two datasets**.
- After this, we have one combined Seurat object instead of the two separate ones.

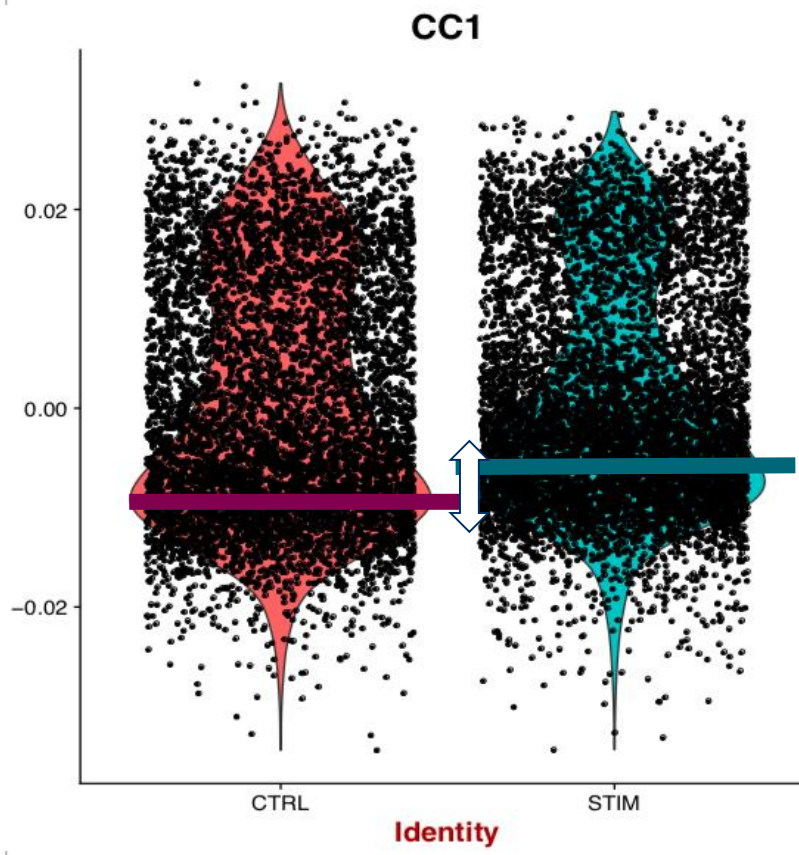


Integrated analysis: Combine samples & perform CCA (2/3)

- CC1 and CC2 in plots: canonical correlation vectors 1 & 2
- From the plots we can see that the cell types are separated to similar looking clusters, but they are a bit "shifted" between the samples

→ we need to "align" them (in the next tool)

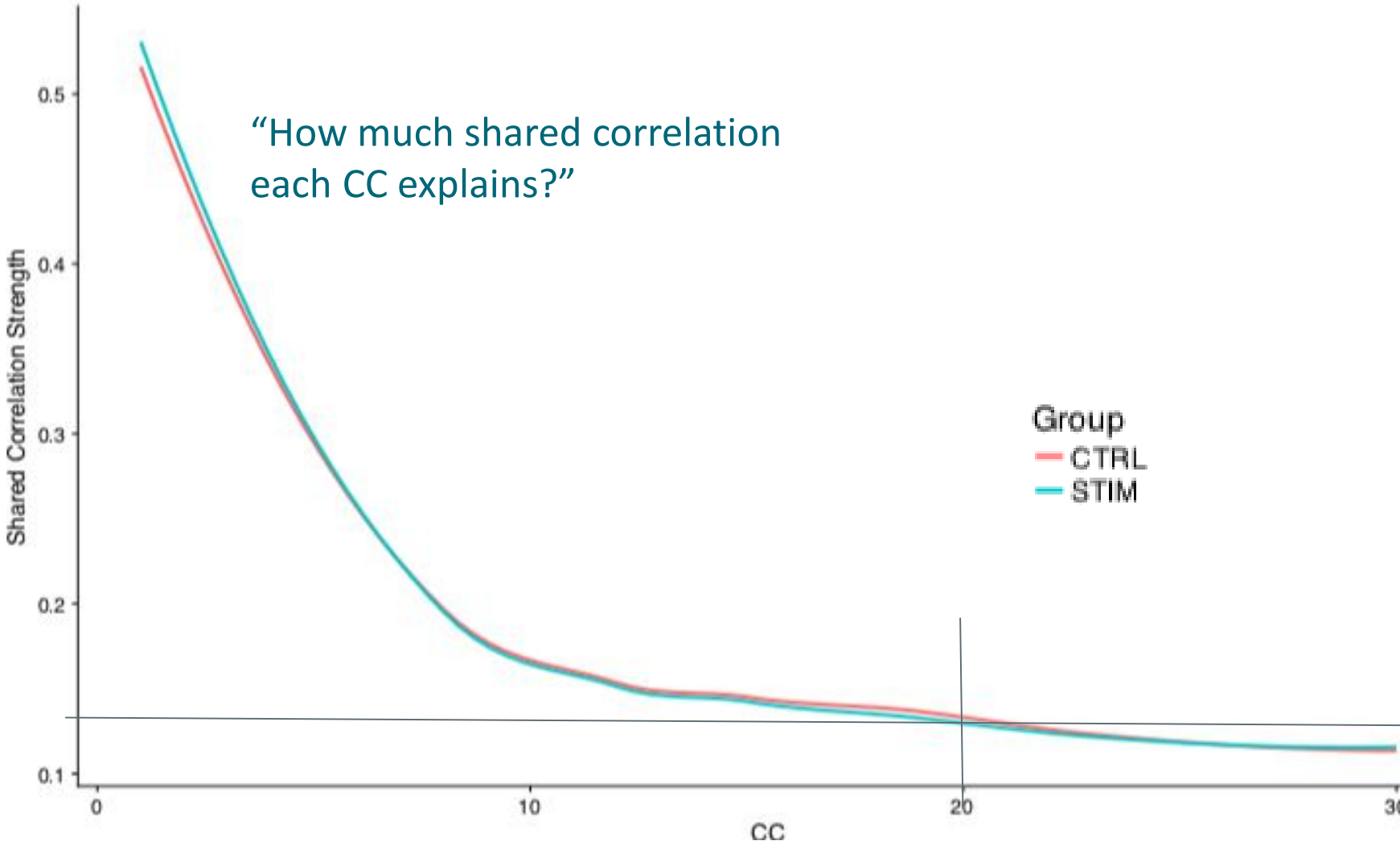




Integrated analysis: Combine samples & perform CCA (3/3)

- Which CCs to choose?
  - Similar problem as the selection of PCs before

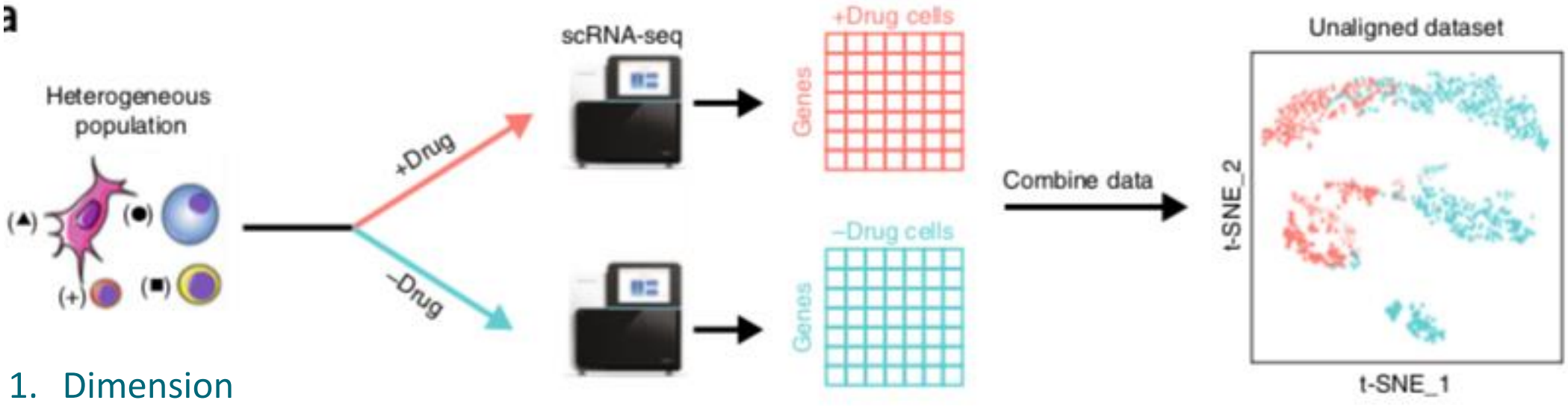
  → similar tools for selection



Canonical Correlation Analysis

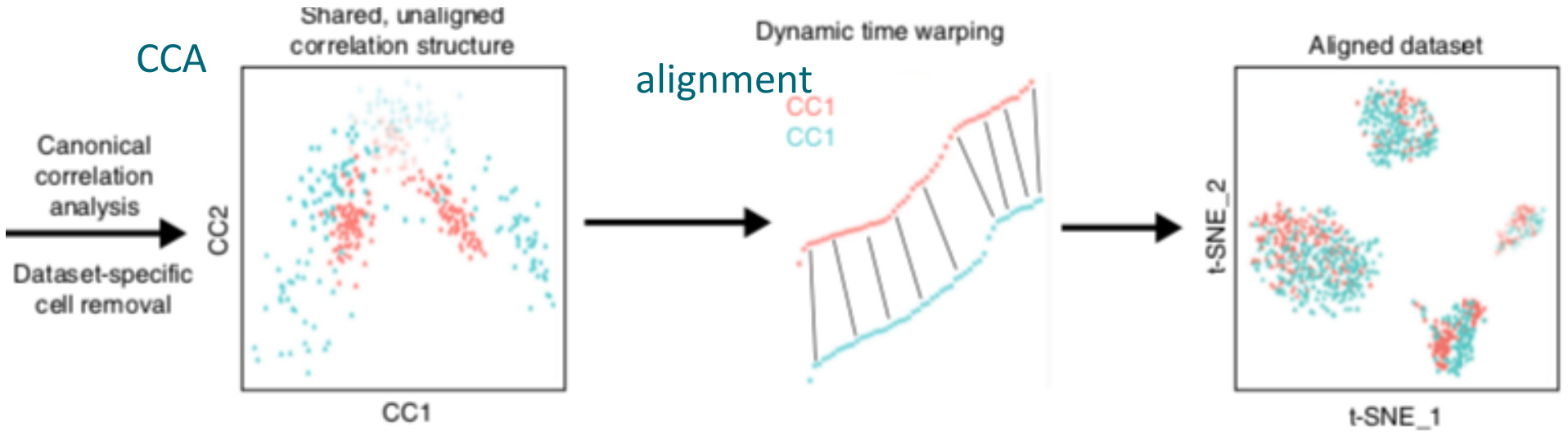
- CCA aims to find linear combinations of features across datasets that are maximally correlated = identify shared correlation structures across datasets
  - CCA has been used for multi-modal genomic analysis from bulk samples (e.g. relationships between gene expression and DNA copy number measurements)
- Here we apply CCA to identify relationships between single cells, from different datasets, based on the same set of genes
  - CCA finds two sets of *canonical 'basis' vectors*
  - embedding cells from each dataset in a low-dimensional space, such that the variation along these vectors is highly correlated between datasets
- Two purposes:
  1. dimension reduction (like with PCA) and
  2. alignment (two samples…)
- For a full description of the method, see Butler et al 2018.

CCA & alignment



1. Dimension reduction (like with PCA)
2. Alignment (reduce noise, single space)

CCA



alignment

More details: Butler et al 2018

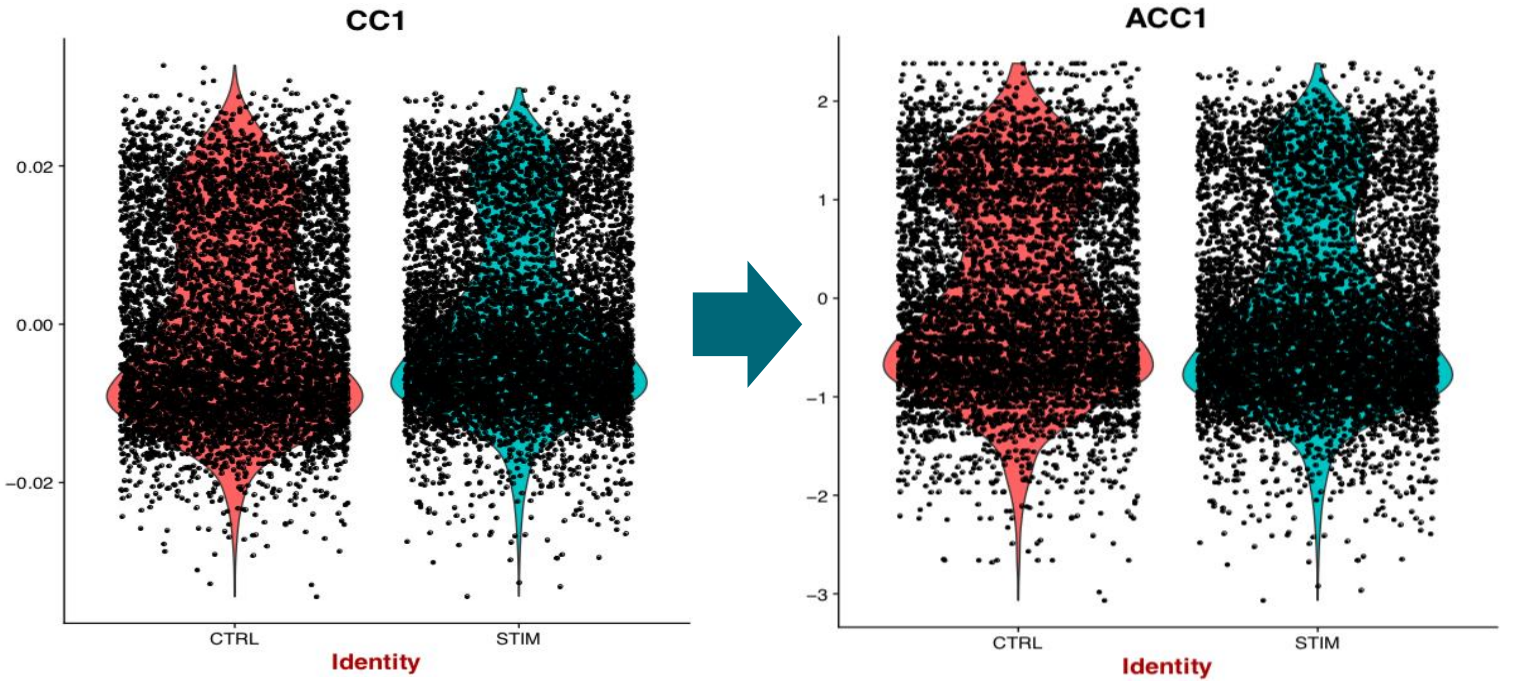
→ single, aligned, low-dimensional space representing both datasets

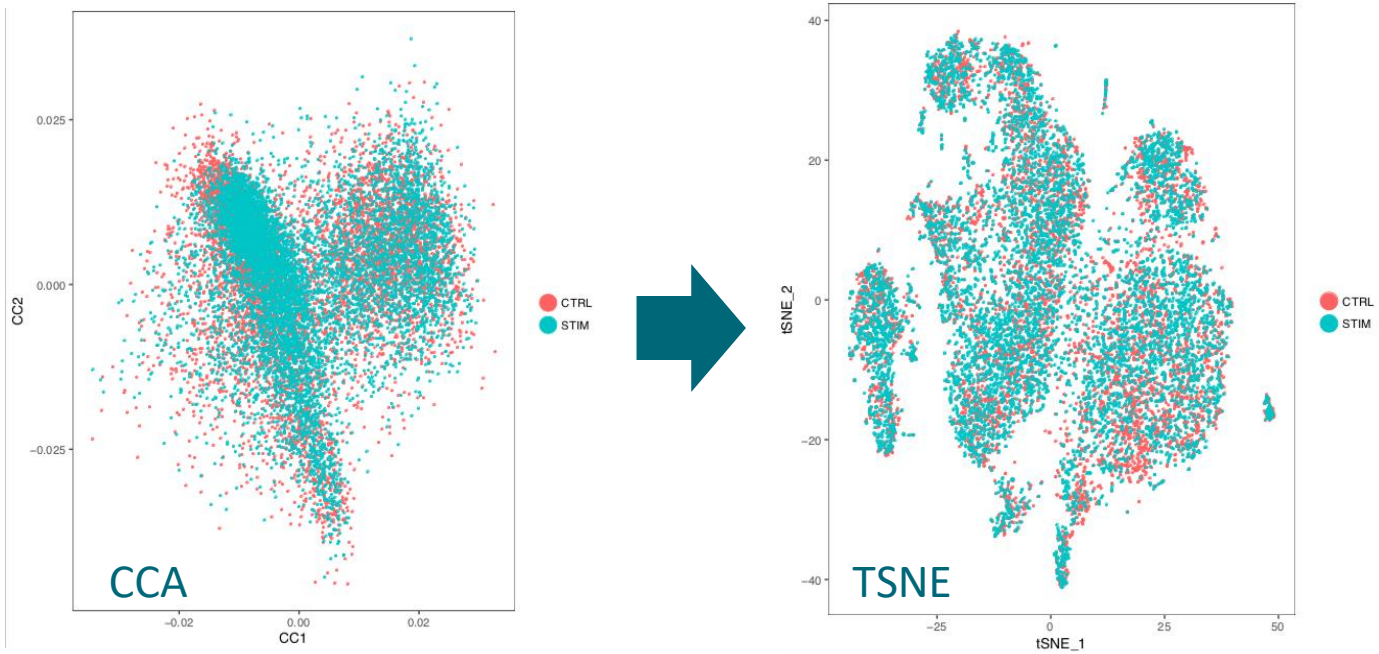
Integrated analysis of two samples

- Steps:
  - Setup + QC & Filtering steps
  - Combine samples & perform CCA
  - **Integrated analysis (aligning, clustering, tSNE)**
  - Find conserved biomarkers for clusters
  - Find differentially expressed genes between samples, within clusters
  - Visualize some interesting genes

Integrated analysis: Integrated analysis

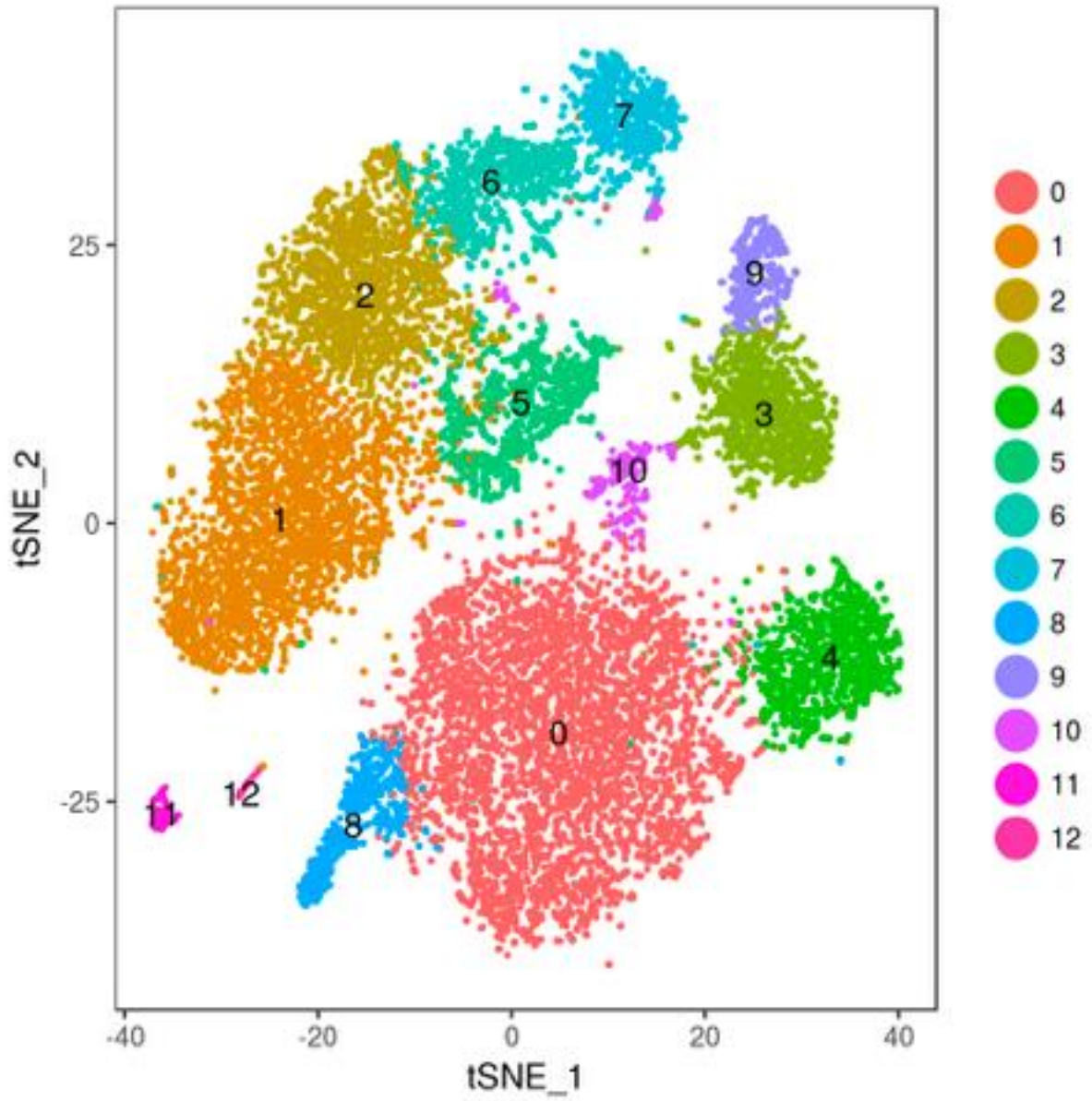
1. Alignment of the samples
2. Visualize the alignment
3. tSNE
4. Clustering





Integrated analysis: Integrated analysis

1. Alignment of the samples
2. Visualize the alignment
3. **Clustering**
4. **tSNE**



Integrated analysis of two samples

- Steps:
  - Setup + QC & Filtering steps
  - Combine samples & perform CCA
  - Integrated analysis (aligning, clustering, tSNE)
  - **Find conserved biomarkers for clusters**
  - **Find differentially expressed genes between samples, within clusters**
  - Visualize some interesting genes

Integrated analysis : Find differentially expressed genes between samples, within clusters

- This can be done **one cluster at a time**
- You can use the **Utilities / Filter table by column value** tool to find the significantly differentially expressed genes (adjusted p-value < 0.05)

For example, in cluster 3, these genes are differently expressed between groups “STIM” and “CTRL”:

	p_val_adj	avg_logFC
ISG15	4.33930013063627e-164	-3.2576275256485
IFIT3	5.26886995181638e-158	-3.11121217480885
IFI6	6.05301576538553e-155	-2.89526528252663
ISG20	6.05136792219949e-154	-2.03769825028896
IFIT1	1.0634402509909e-142	-2.844595027688
MX1	2.07790042443981e-128	-2.26530710646114

Utilities / Filter table by column value

Column to filter by	p_val_adj
Does the first column have a title	no
Cutoff	0.05
Filtering criteria	smaller-than

Integrated analysis: Find conserved biomarkers for clusters

- Also done **one cluster at a time**
- Conserved marker = marker for the cluster, regardless of the treatment
- Matrix containing a ranked list of putative conserved markers, and associated statistics
  - p-values within each group and a combined p-value (= bigger p-value of the two)

Showing 475 rows of 475 and all 13 columns

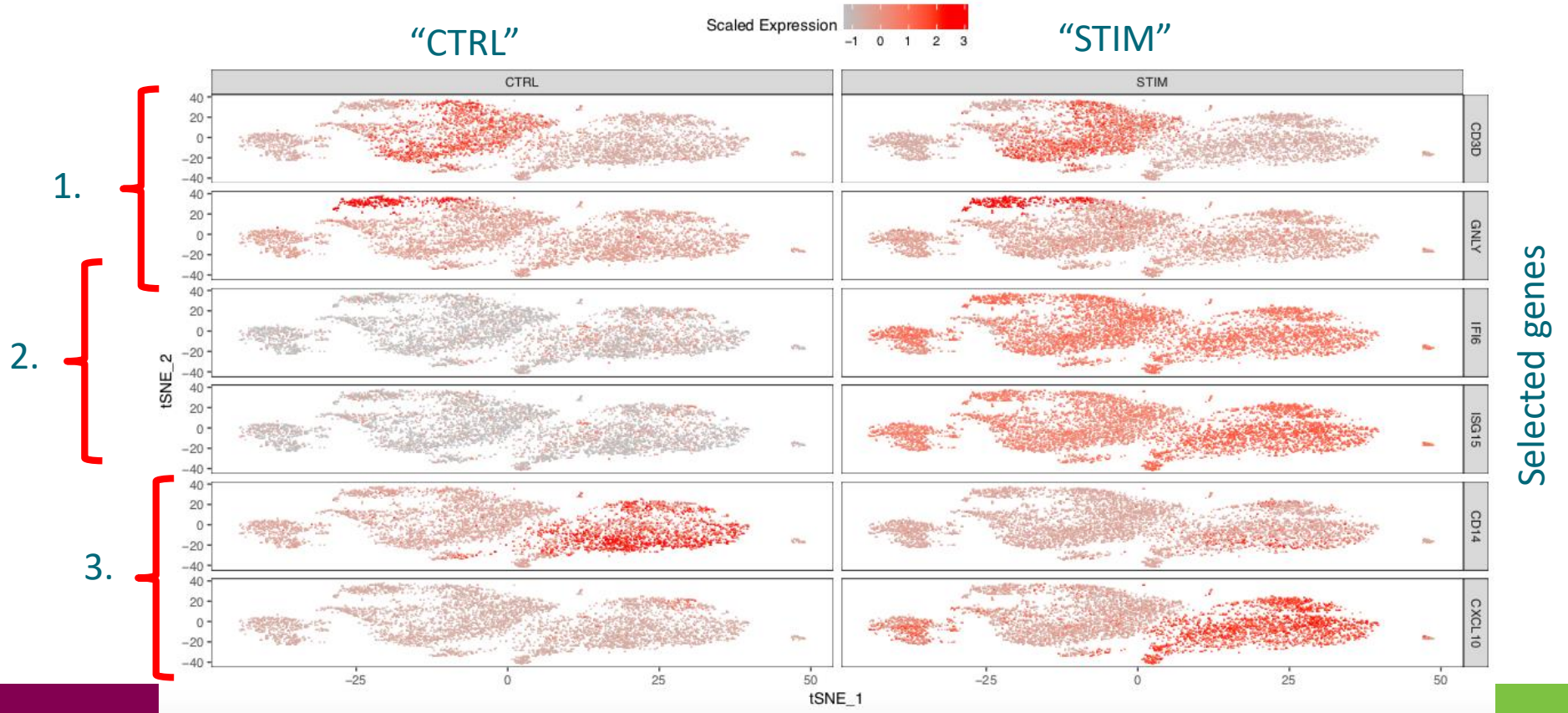
	CTRL_p_val	CTRL_avg_logFC	CTRL_pct.1	CTRL_pct.2	CTRL_p_val_adj	STIM_p_val	STIM_avg_logFC	STIM_pct.1	STIM_pct.2	STIM_p_val_adj	max_pval
CD79A	0	2.61961744...	0.822	0.03	0	0	2.32967504...	0.715	0.022	0	0
MS4A1	0	2.01558696...	0.591	0.017	0	0	1.83017689...	0.486	0.014	0	0
CD79B	0	1.59700846...	0.413	0.016	0	4.476048...	0.82576105...	0.16	0.006	5.95896306...	4.476048...
CD74	1.778017...	1.57718401...	0.998	0.661	2.36707440...	1.168511...	1.44542600...	0.993	0.665	1.55563926...	1.778017...
BANK1	2.801773...	0.93747648...	0.201	0.005	3.73000070...	1.739120...	1.10870118...	0.246	0.008	2.31529177...	2.801773...
TNFRSF13B	8.134026...	1.11014650...	0.194	0.003	1.08288301...	5.164170...	1.00460255...	0.195	0.003	6.87506017...	8.134026...
ANXA1	3.960421...	-2.2757969...	0.103	0.784	5.27250963...	2.361844...	-2.3748562...	0.104	0.816	3.14432378...	3.960421...
KIAA0226L	6.838293...	1.14621769...	0.251	0.011	9.10382048...	1.069508...	0.86038124...	0.179	0.007	1.42383603...	1.069508...
BLNK	1.742754...	0.96373006...	0.208	0.009	2.32012861...	4.602775...	1.12895086...	0.282	0.024	6.12767468...	1.742754...

Integrated analysis of two samples

- Steps:
  - Setup + QC & Filtering steps
  - Combine samples & perform CCA
  - Integrated analysis (aligning, clustering, tSNE)
  - Find conserved biomarkers for clusters
  - Find differentially expressed genes between samples, within clusters
  - **Visualize some interesting genes**

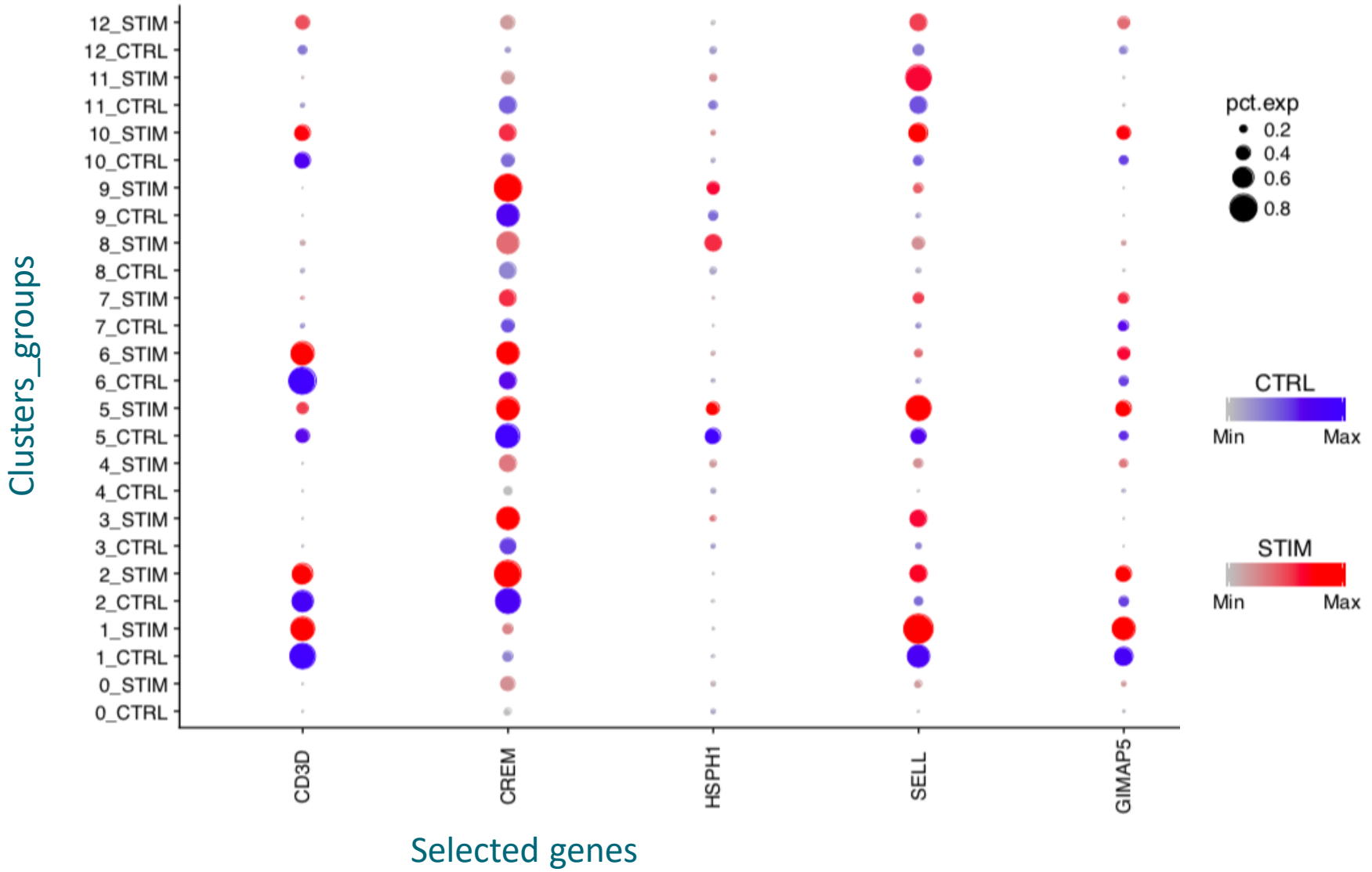
Integrated analysis: Visualize interesting genes (1/2)

1. No change between the samples: conserved cell type markers
2. Clear change between the samples: cell type independent marker for the treatment
3. Change between samples in some clusters: cell type dependent behavior to the treatment



Integrated analysis: Visualize interesting genes (2/2)

- "Split dot plot"
- Size:
  the percentage of cells in a cluster expressing given gene
- Color brightness:
  the average expression level of 'expressing' cells



Integrated analysis of two samples

- We wish to:
  - Identify cell types that are present in both datasets
  - Obtain cell type markers that are conserved in both control and stimulated cells
  - Compare the datasets to find cell-type specific responses to stimulation
- Steps:
  - Setup + QC & Filtering steps
  - **Combine samples & perform CCA**
  - Integrated analysis (**aligning**, clustering, tSNE)
  - Find conserved biomarkers for clusters
  - Find differentially expressed genes between samples, within clusters
  - Visualize some interesting genes

Our example data:

- From the Seurat tutorial: https://satijalab.org/seurat/immune_alignment.html
- Two samples of peripheral blood mononuclear cells (PBMCs):
  - Controls (**CTRL**)
  - Treated cells (**STIM**), stimulated with interferon beta
- → clustering based on
  1. Cell types
  2. Treatment status

What did I learn?

- How to operate the Chipster software
- Analysis of single cell RNA-seq data
  - Central concepts
  - File formats
  - Analysis steps, practised in exercises
    1. DropSeq data preprocessing from raw reads to expression values
    2. Clustering analysis of 10X Genomics data with Seurat tools
    3. Integrated analysis with two samples with Seurat tools