Omics and Data Science Concepts

Matthew Galbraith
Linda Crnic Institute for Down Syndrome

Data Science for Developing Scholars in
Down Syndrome Research (DS3) 2025

INCLUDE
INvestigation of Co-occurring conditions across the Lifespan to Understand Down syndromE

Linda Crnic Institute for Down Syndrome
UNIVERSITY OF COLORADO ANSCHUTZ MEDICAL CAMPUS

GLOBAL
DOWN SYNDROME FOUNDATION®

The importance of understanding all aspects of your data

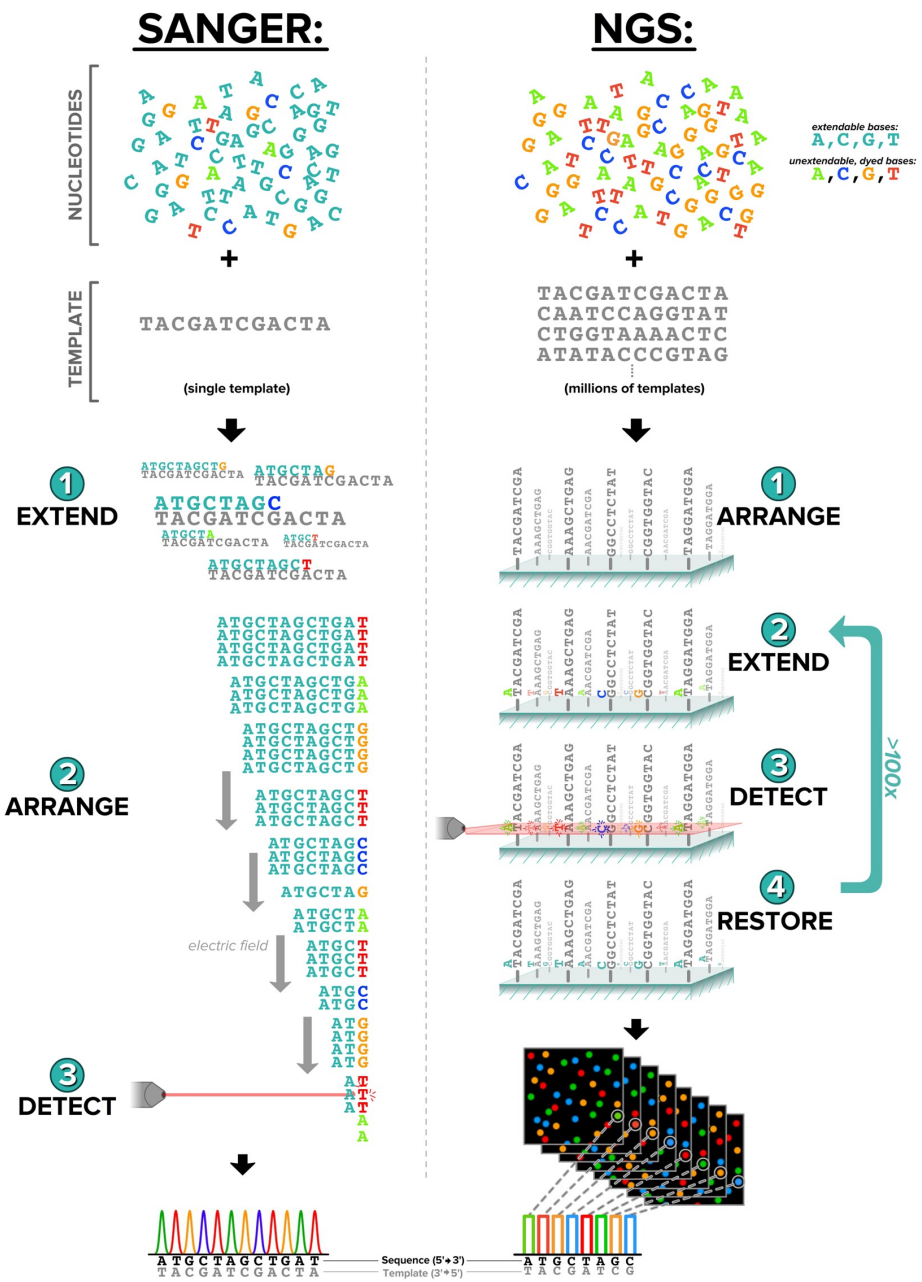
Some questions to ask yourself

- What is the biological question being addressed?
- What is the sample source?
  e.g. blood, plasma, tissue, cells, etc
- What variables are potentially important?
  e.g. sex, age, batch, other confounders etc
- What is being measured?
  e.g. DNA/RNA, polyA, selection/depletion, amplicon, protein etc
- What is the nature of the measurement?
  e.g. counts, relative abundance, absolute concentration etc
- What are the file types/formats?
  e.g. Fastq, Sam/Bam, text/tab/csv, proprietary format etc
- What is the genome reference and annotation database?
- What is the expected data distribution?
- What is the expected level of missingness in the data?
- Do the data require preprocessing?
  What has already been done? What is required?

What is Next Generation Sequencing (NGS)?
And why use it?

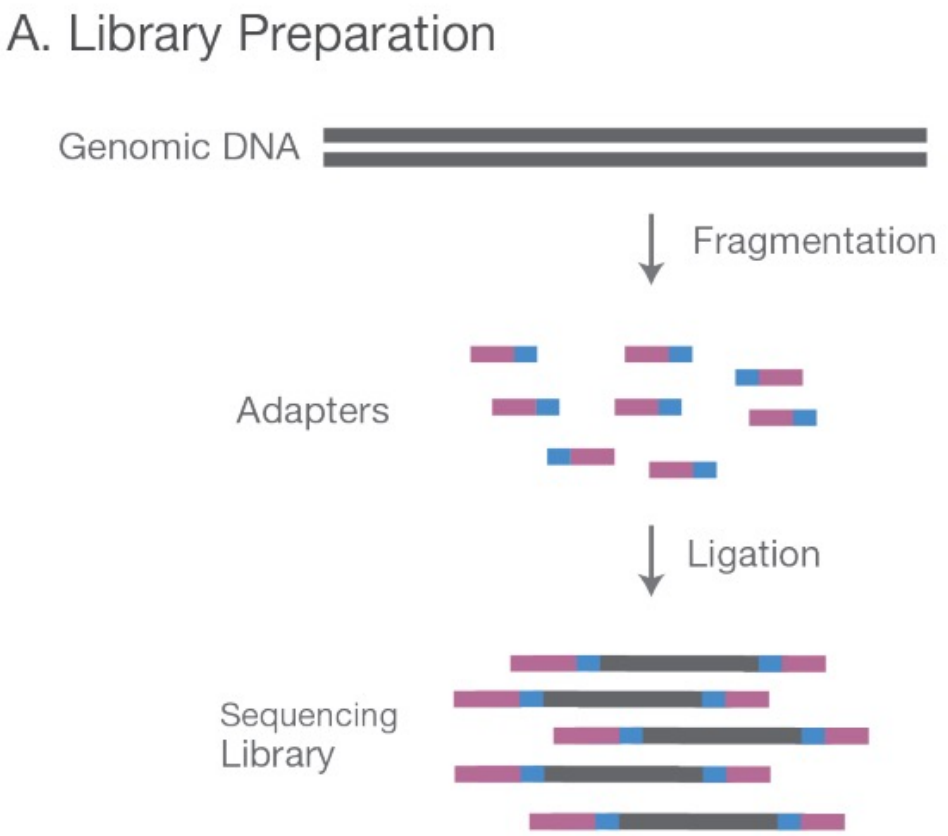
What is Next Generation Sequencing (NGS)? And why use it?

	Sanger sequencing	NGS
Features	• Single template population • 'Chain termination' using labelled dNTPs • Not immobilized • Resolved by capillary gel electrophoresis • Laser detection	• Mixed template population (>10^6) • 'Sequencing by synthesis' with labelled dNTPs • Immobilized • Resolved & detected by microscopy (typically)
Advantages	• Long reads (700-900+ bases) • Fast and cheap • Simple analysis	• Very high throughput (massively parallel) • High discovery power
Disadvantages	• Low throughput • Low discovery power	• Shorter reads (typically) • Per read accuracy • More expensive • Complex analysis

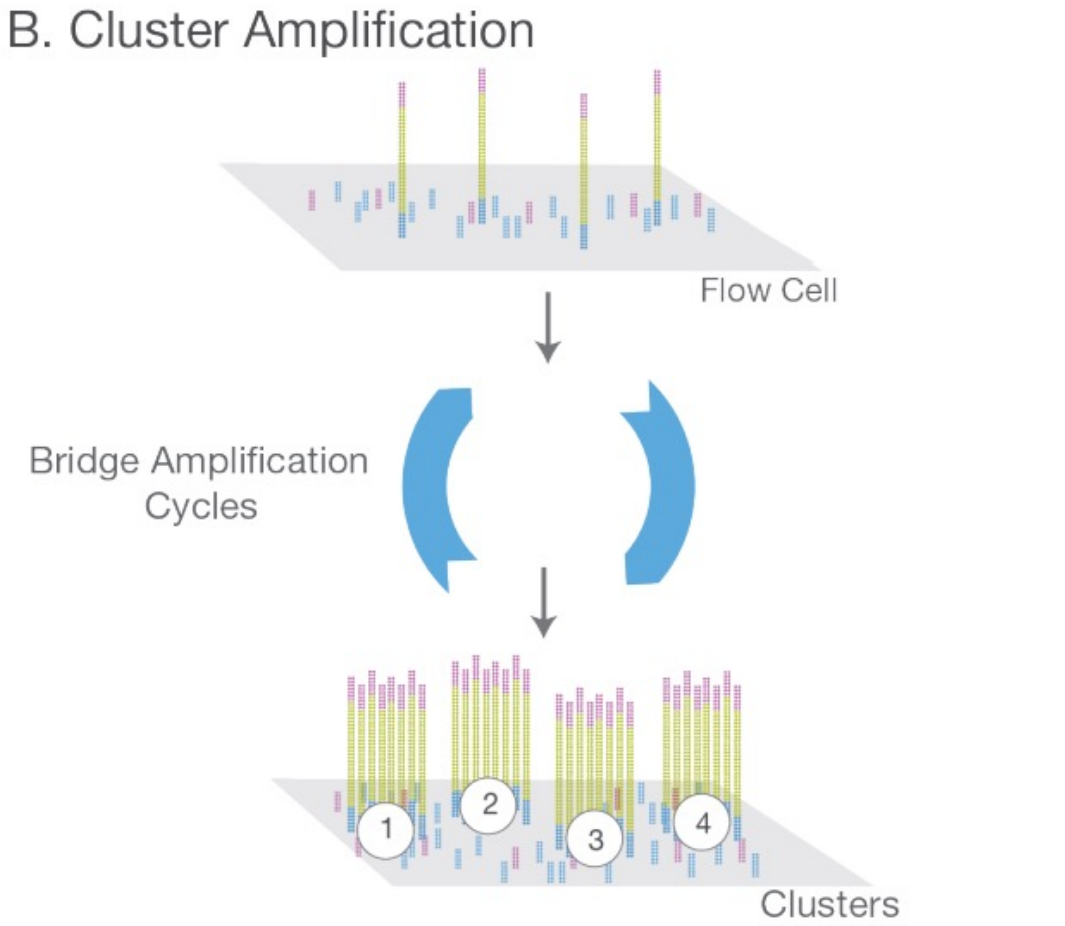


https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4633438/

Illumina NGS: general workflow



NGS library is prepared by fragmenting a gDNA sample and ligating specialized adapters to both fragment ends.



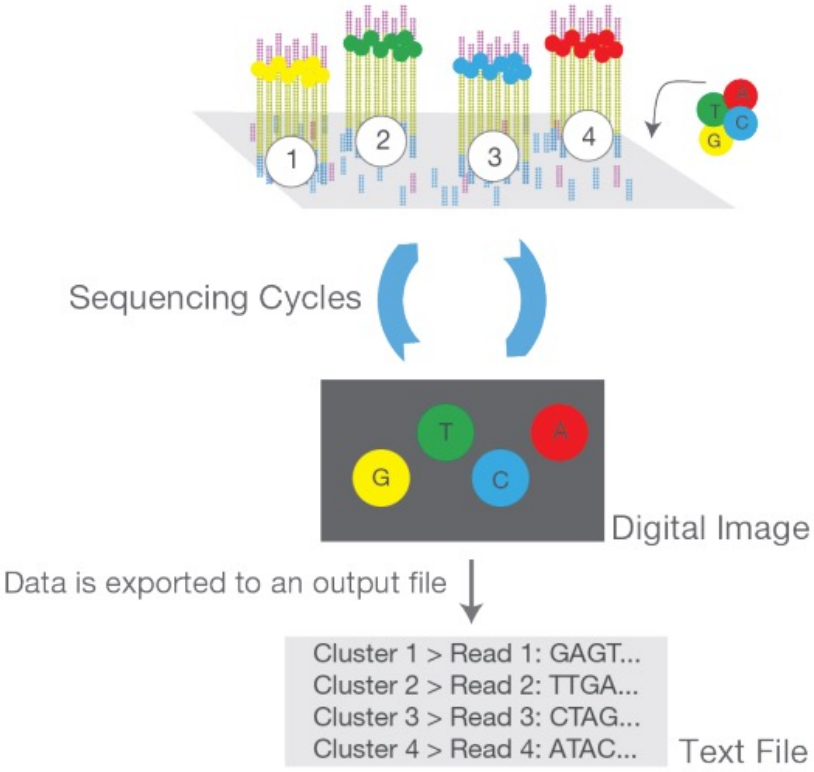
Library is loaded into a flow cell and the fragments are hybridized to the flow cell surface. Each bound fragment is amplified into a clonal cluster through bridge amplification.

https://www.illumina.com/content/dam/illumina-marketing/documents/products/illumina_sequencing_introduction.pdf

Illumina NGS: general workflow

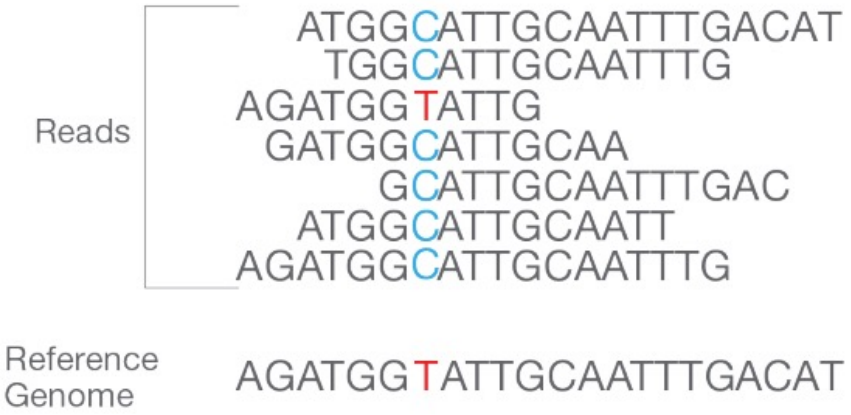
‘Sequencing by synthesis”

C. Sequencing



Sequencing reagents, including fluorescently labeled nucleotides, are added and the first base is incorporated. The flow cell is imaged and the emission from each cluster is recorded. The emission wavelength and intensity are used to identify the base. This cycle is repeated “n” times to create a read length of “n” bases.

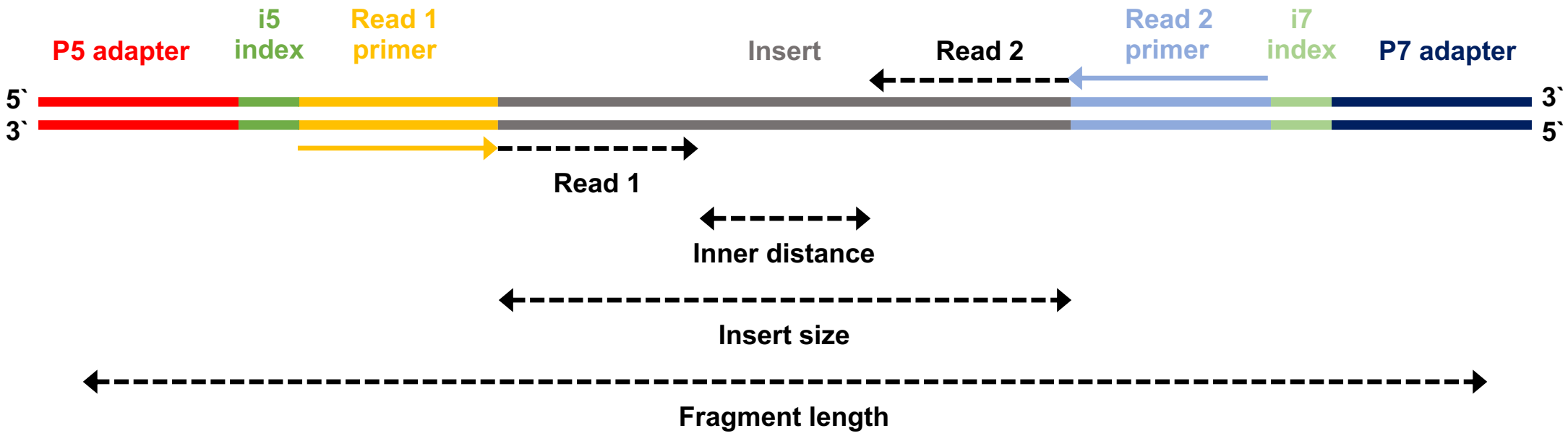
D. Alignment and Data Anaylsis



Reads are aligned to a reference sequence with bioinformatics software. After alignment, differences between the reference genome and the newly sequenced reads can be identified.

https://www.illumina.com/content/dam/illumina-marketing/documents/products/illumina_sequencing_introduction.pdf

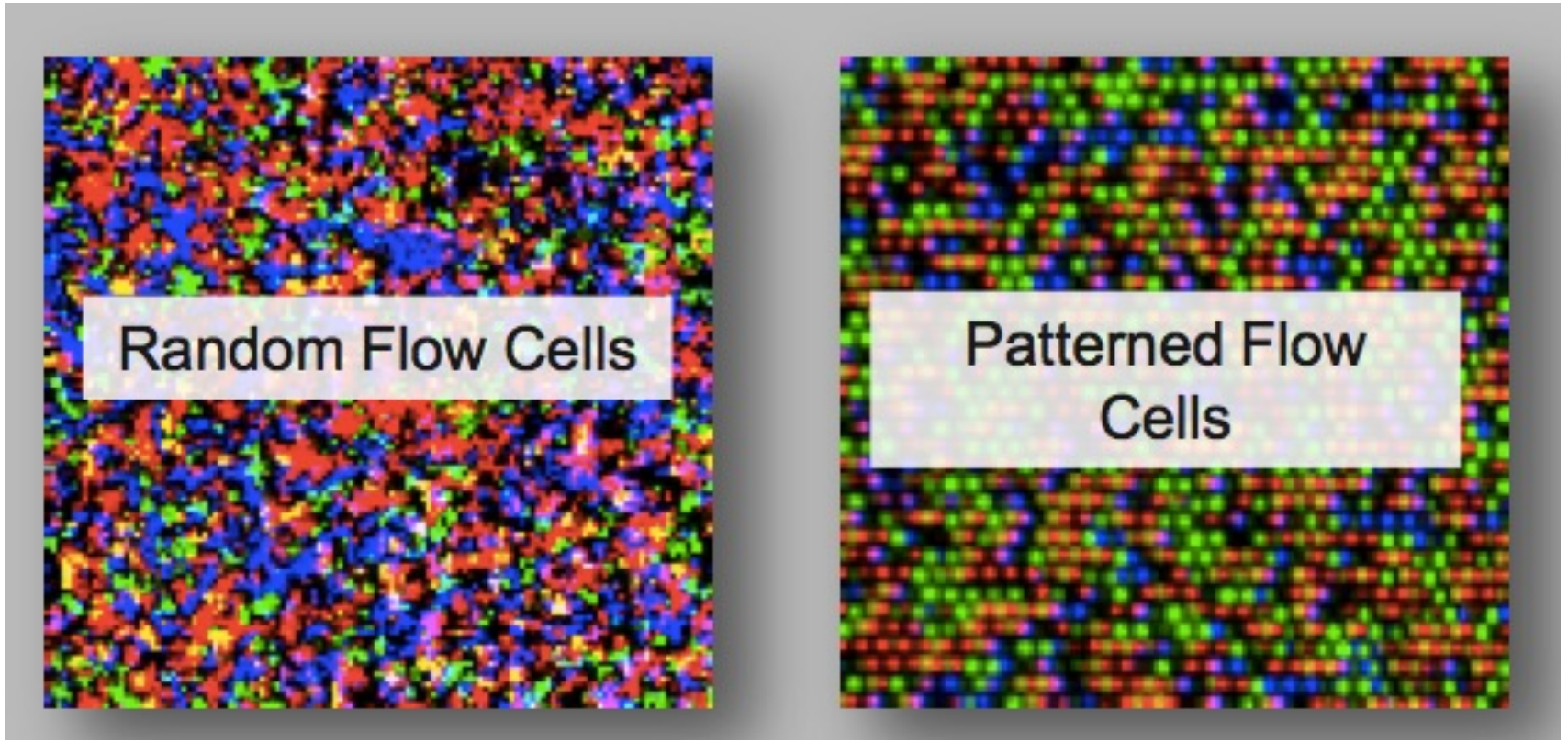
Illumina NGS: Anatomy of a library fragment



NB: Indices have additional primers and sequencing reads

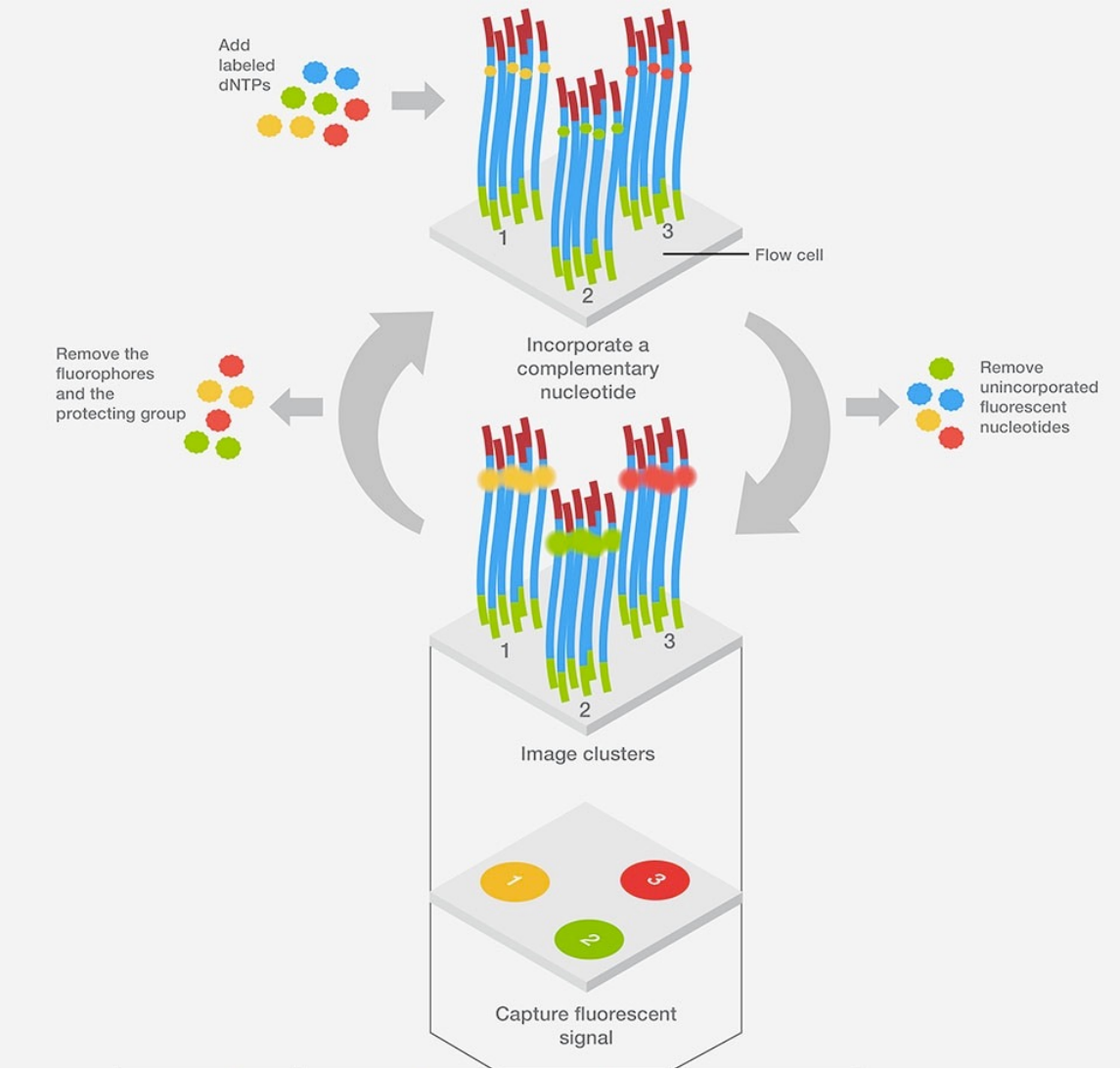
Illumina NGS: 'Sequencing by synthesis' on a microscope

- Flow cells are coated with oligos complementary to adapters
- Library fragments are immobilized and undergo cluster amplification
- Step-wise addition of labelled nucleotides is imaged



https://core-genomics.blogspot.com/2016/01/almost-everything-you-wanted-to-know.html

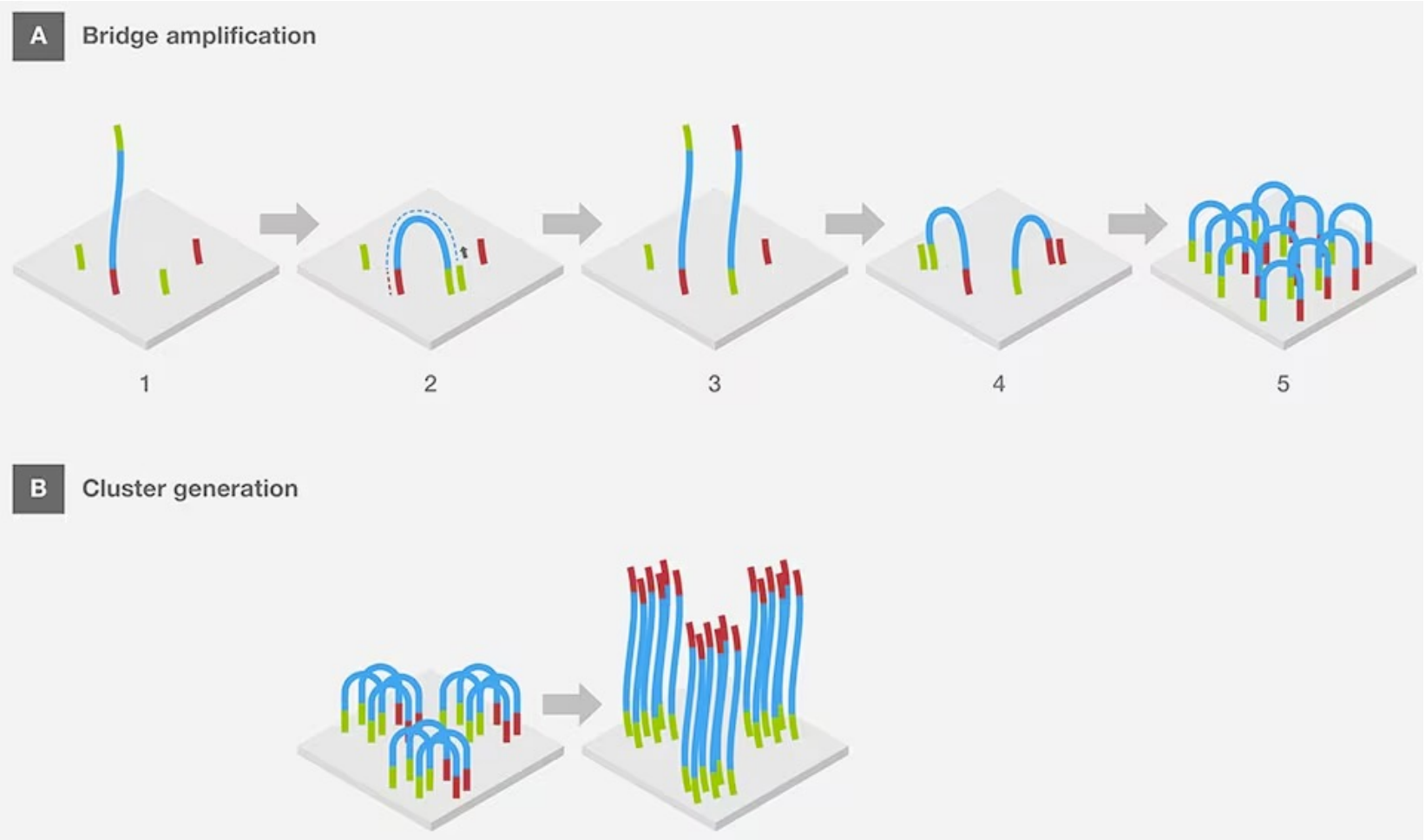
Illumina NGS: 'Sequencing by synthesis' on a microscope



https://www.thermofisher.com/us/en/home/life-science/cloning/cloning-learning-center/invitrogen-school-of-molecular-biology/next-generation-sequencing/illumina-workflow.html

Cluster generation: Random flow cells (older Illumina systems)

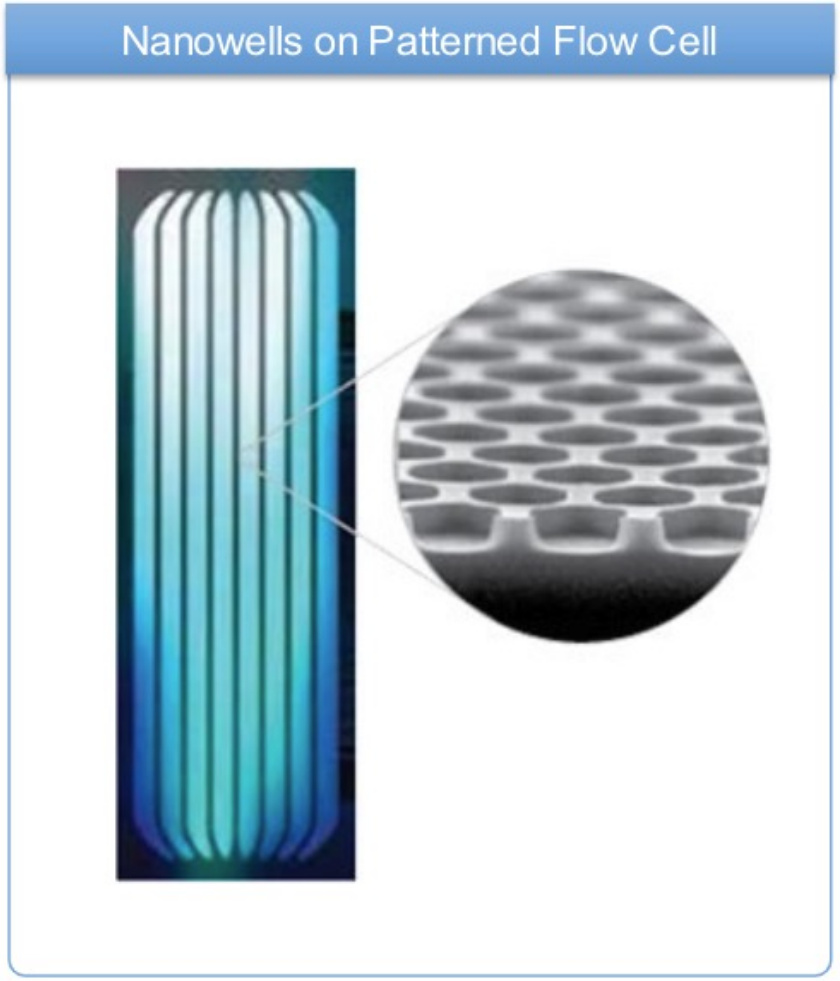
- Flow cell impregnated with random ‘lawn’ of oligos complementary to library adaptors
- Single-stranded library allowed to anneal
- Clonal amplification occurs by ‘bridge amplification’
- Cluster density determined by library concentration at loading



https://www.thermofisher.com/us/en/home/life-science/cloning/cloning-learning-center/invitrogen-school-of-molecular-biology/next-generation-sequencing/illumina-workflow.html

Cluster generation: Patterned flow cells (newer Illumina systems - HiSeq, Novaseq)

- Contain billions of nanowells at fixed locations
- Have structured organization that:
  - provides for even cluster spacing
  - allows for generation of uniformly sized clusters
  - enables accurate resolution of flow cells clustered at extremely high densities—all of the cluster positions are already known
- Use a proprietary clustering method called exclusion amplification (ExAmp), which is performed on the cBot

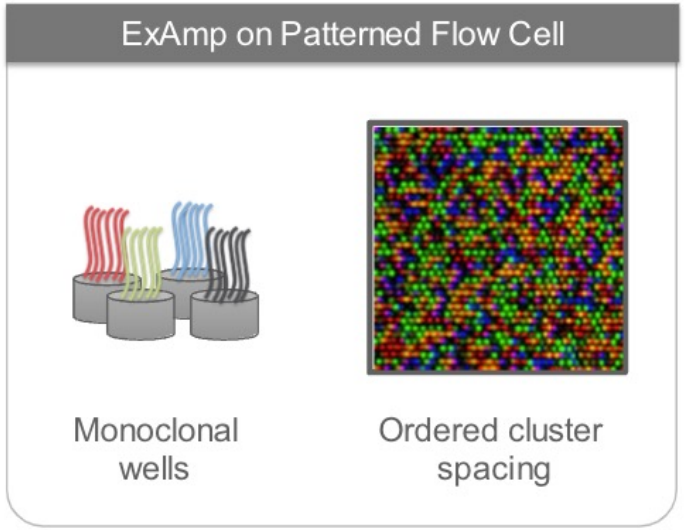


https://support.illumina.com/content/dam/illumina-support/courses/examp-cluster-workflow/story_html5.html

Cluster generation: Patterned flow cells (newer Illumina systems - HiSeq, Novaseq)

ExAmp Cluster Amplification

- Ensures only a single DNA template binds and forms a cluster within a single well
- Uses almost instantaneous amplification to exclude other DNA fragments from hybridizing within a well
- Results in high well occupancy and maximum data output
- Bridge amplification would result in too many polyclonal cells



https://support.illumina.com/content/dam/illumina-support/courses/examp-cluster-workflow/story_html5.html

Potential limitations of ExAmp chemistry

- **Proprietary**
  Few details provided

- **Intolerance to variable insert sizes**
  Smaller fragments favored
  Problem if >1 population of fragment lengths expected (eg Amplicons, PhiX)

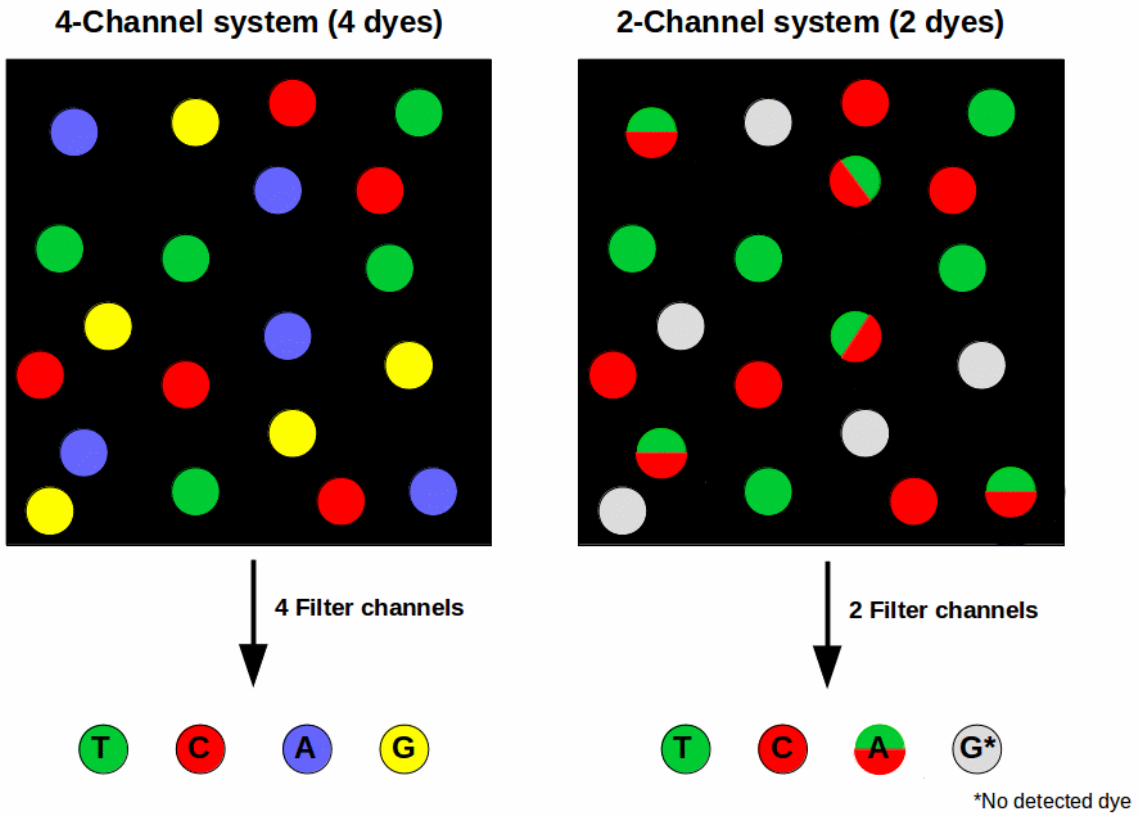
- **Higher susceptibility to adapter contamination**
  Problem with adapter contamination >1% (eg 5% could lead to 60% of reads)
  Solution: Use extra cleanup if needed (eg SPRI bead cleanup)

- **Increased duplication rates**
  Duplicates reduce genome coverage
  Solution: Use tools to mark and/or remove duplicates

- **Higher rates of 'index hopping'**
  Polymerase may transfer to different template after index extension
  Can lead to low rate of multiplexed sample misidentification
  Solution: Use unique dual indexing

https://core-genomics.blogspot.com/2016/01/almost-everything-you-wanted-to-know.html
https://sequencing.qcfail.com/articles/illumina-patterned-flow-cells-generate-duplicated-sequences/
https://sequencing.qcfail.com/articles/the-latest-illumina-sequencers-muddle-samples/

Newer Illumina systems use only 2 dyes



Pros
Faster sequencing
Cheaper optics

Potential Cons
Distortion of base qualities
Lower sequence accuracy

https://www.ecseq.com/support/ngs/do-you-have-two-colors-or-four-colors-in-Illumina

Illumina NGS: Multiplexing samples

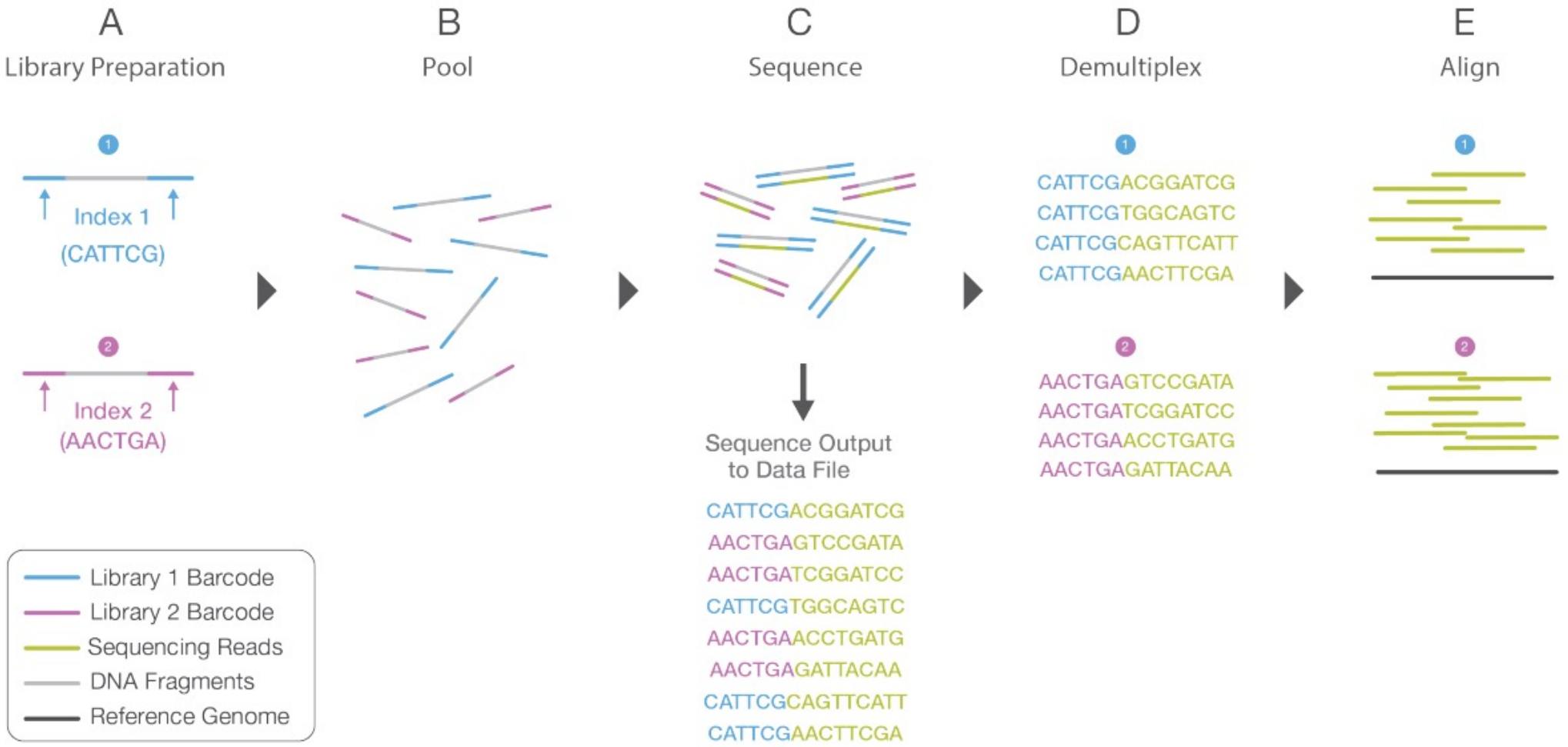
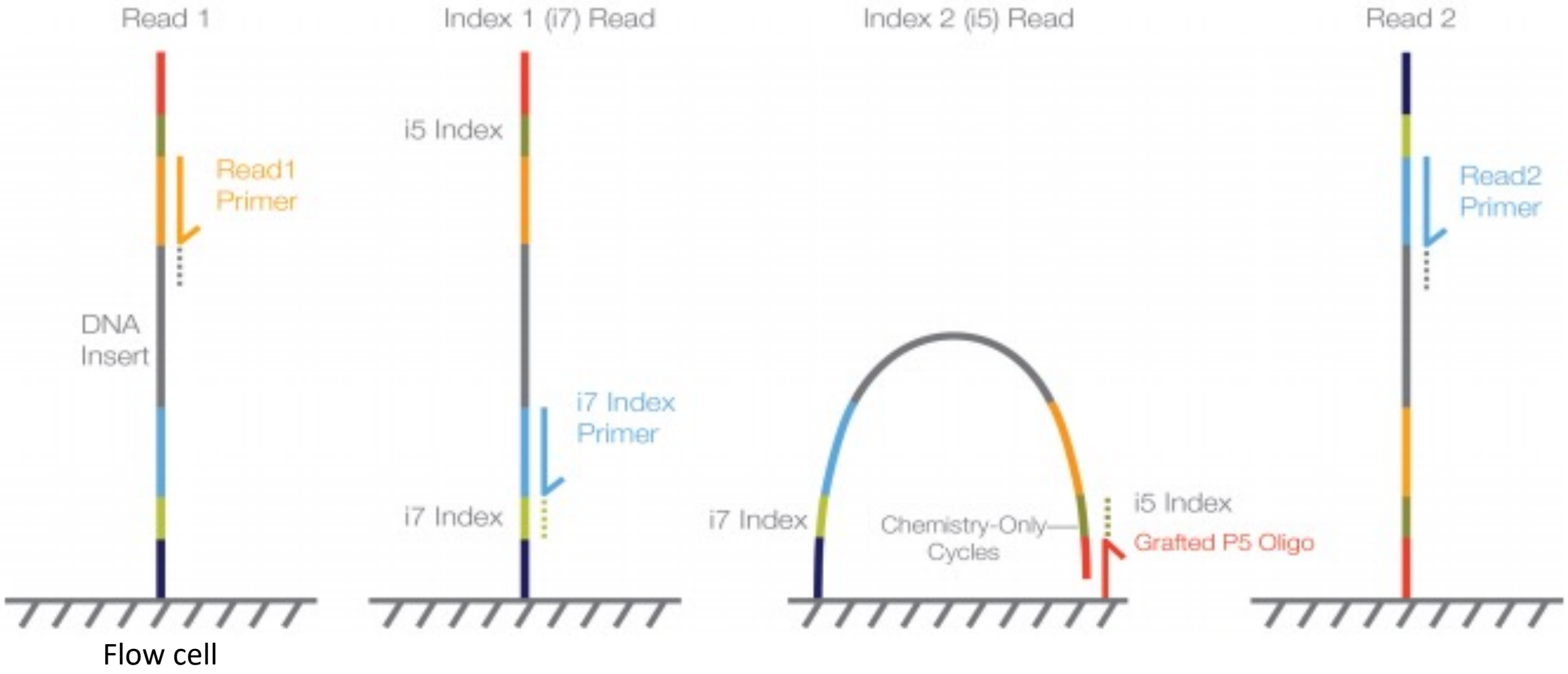


Figure 5: Library Multiplexing Overview—(A) Unique index sequences are added to two different libraries during library preparation. (B) Libraries are pooled together and loaded into the same flow cell lane. (C) Libraries are sequenced together during a single instrument run. All sequences are exported to a single output file. (D) A demultiplexing algorithm sorts the reads into different files according to their indexes. (E) Each set of reads is aligned to the appropriate reference sequence.

https://www.illumina.com/content/dam/illumina-marketing/documents/products/illumina_sequencing_introduction.pdf

Illumina NGS: Dual indexed paired-end workflow



https://knowledge.illumina.com/library-preparation/general/library-preparation-general-reference_material-list/000002099

Deduplication using Unique Molecular Identifiers (UMIs)

- Unique molecular identifiers (UMIs), or molecular barcodes (MBC) are short, random sequences incorporated into library fragments
- This allows for computational deduplication
- Especially important for single-cell approaches

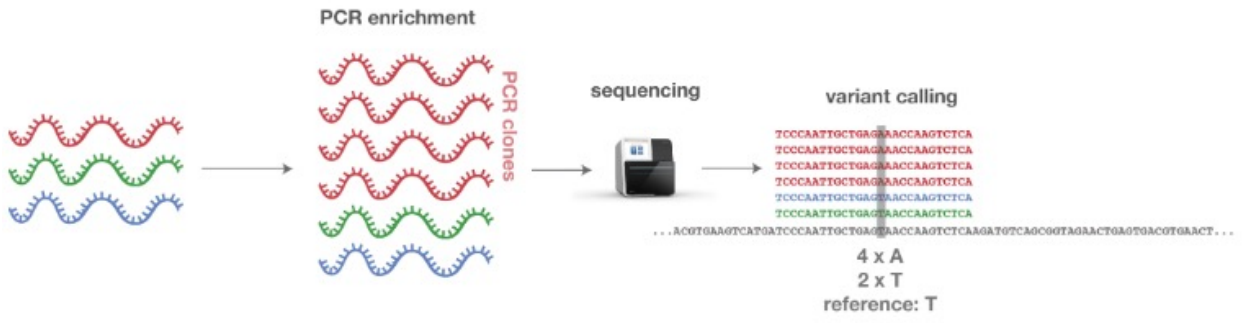


Figure 1: Error produced by PCR clones.

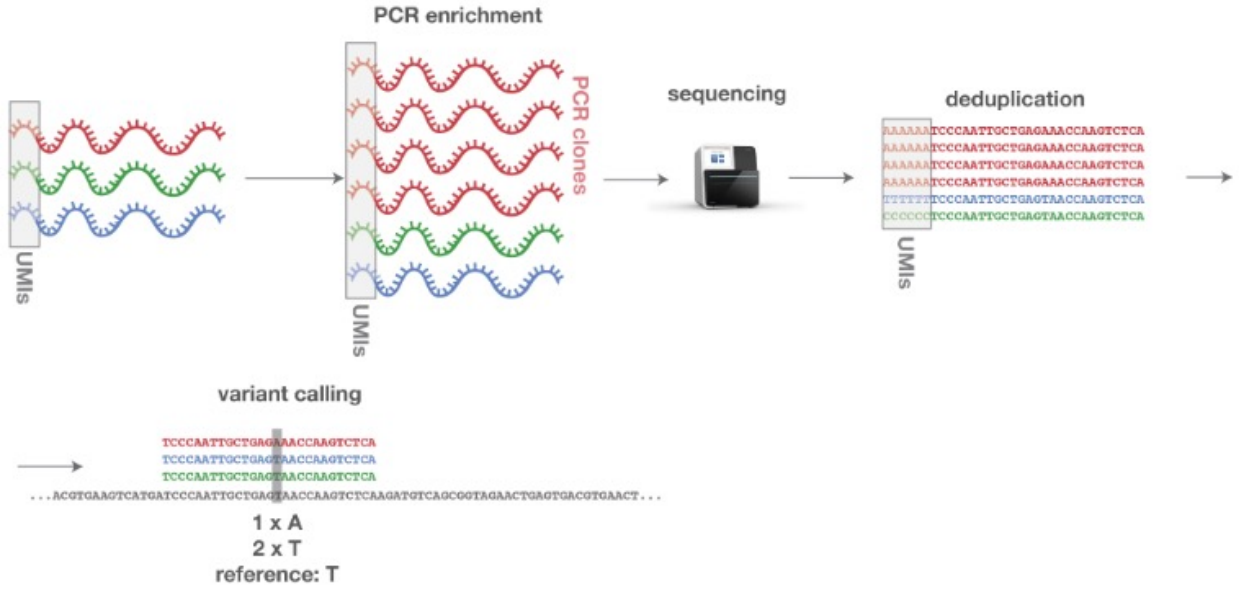
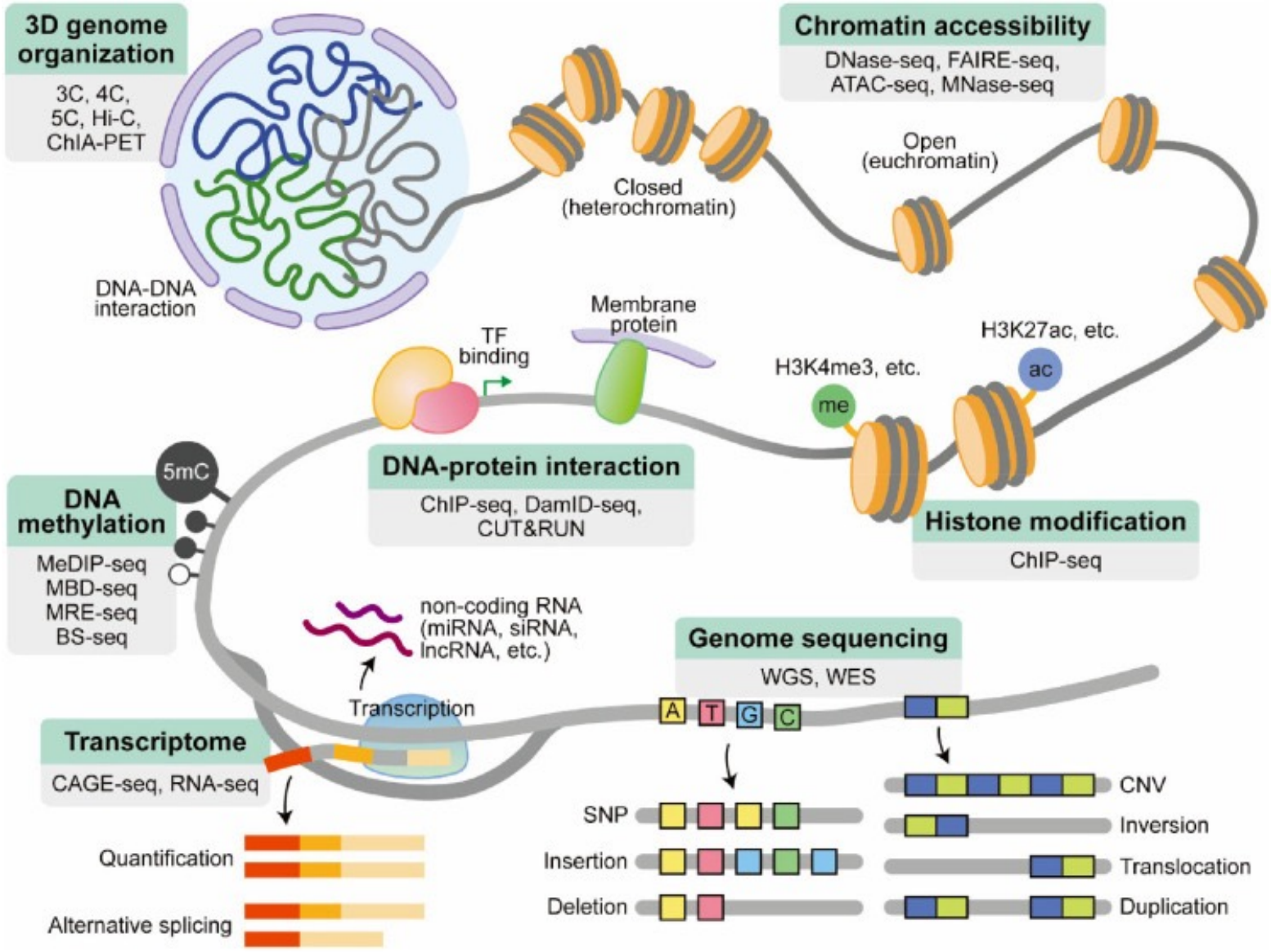


Figure 2: After de-duplication using UMIs, variants can be correctly called.

https://www.ecseq.com/support/ngs/how-can-unique-molecular-identifiers-help-to-reduce-quantitative-biases

NGS: Applications in Eukaryotes



+ many more!

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10703097/

RNA Sequencing: Illumina TruSeq Stranded mRNA

Selection for mRNA
Alternative: deplete rRNA
Optional: deplete globin RNA

NB: Read 1 corresponds to the *antisense* strand.

Figure 1 Purifying and Fragmenting mRNA

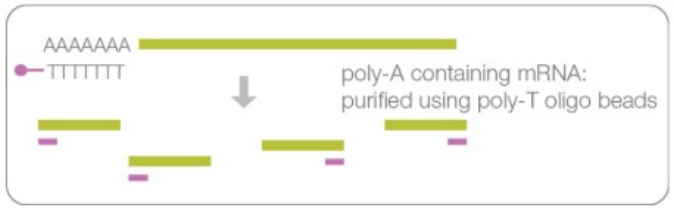


Figure 2 Synthesizing First Strand cDNA

Figure 3 Synthesizing Second Strand cDNA

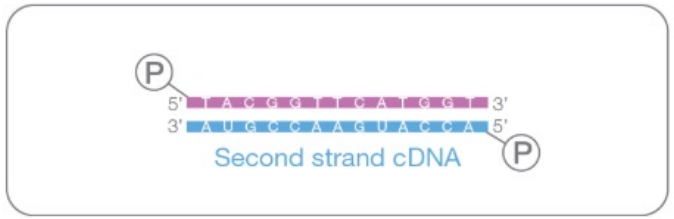


Figure 4 Adenylating 3' Ends

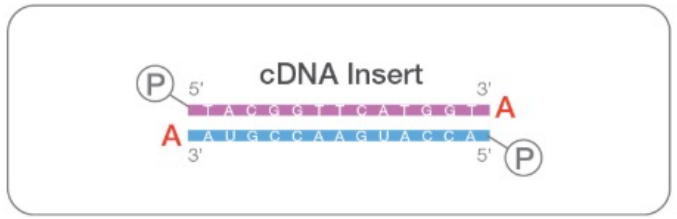


Figure 5 Ligating Adapters

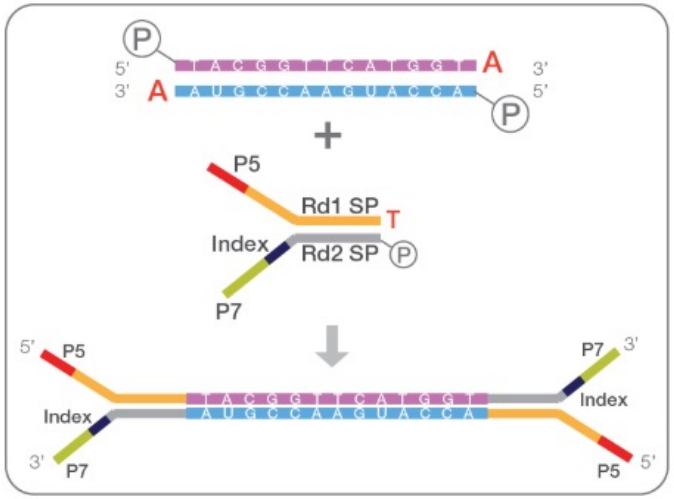
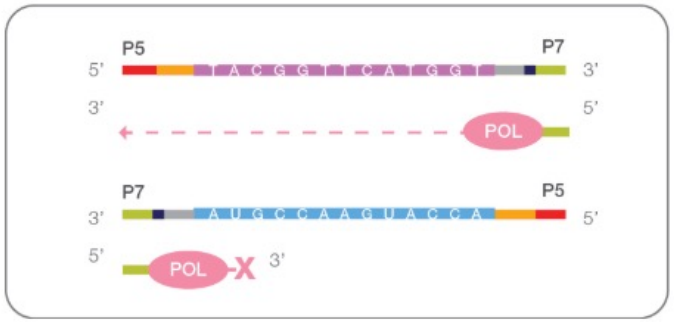
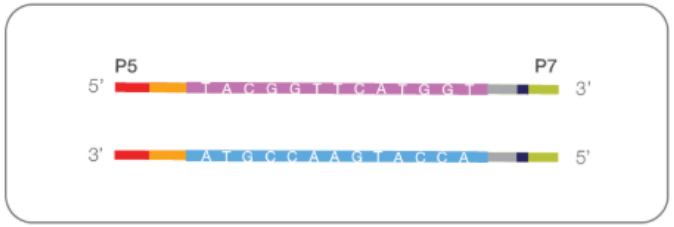


Figure 6 Enriching DNA Fragments



Strand specific
Polymerase used in the assay does not incorporate past dUTP. Therefore, the second strand is effectively quenched during amplification.

Figure 7 LS Final Library



Indexing/Multiplexing
The LS library features a single-index adapter, as shown in this workflow. The HS library features a dual-index adapter, which contains a unique index at each end (not shown).

https://support.illumina.com/sequencing/sequencing_kits/truseq-stranded-mrna/documentation.html

RNA Sequencing: Tecan Universal Plus™ mRNA-Seq

Figure 1. Universal Plus mRNA-Seq with NuQuant Workflow.

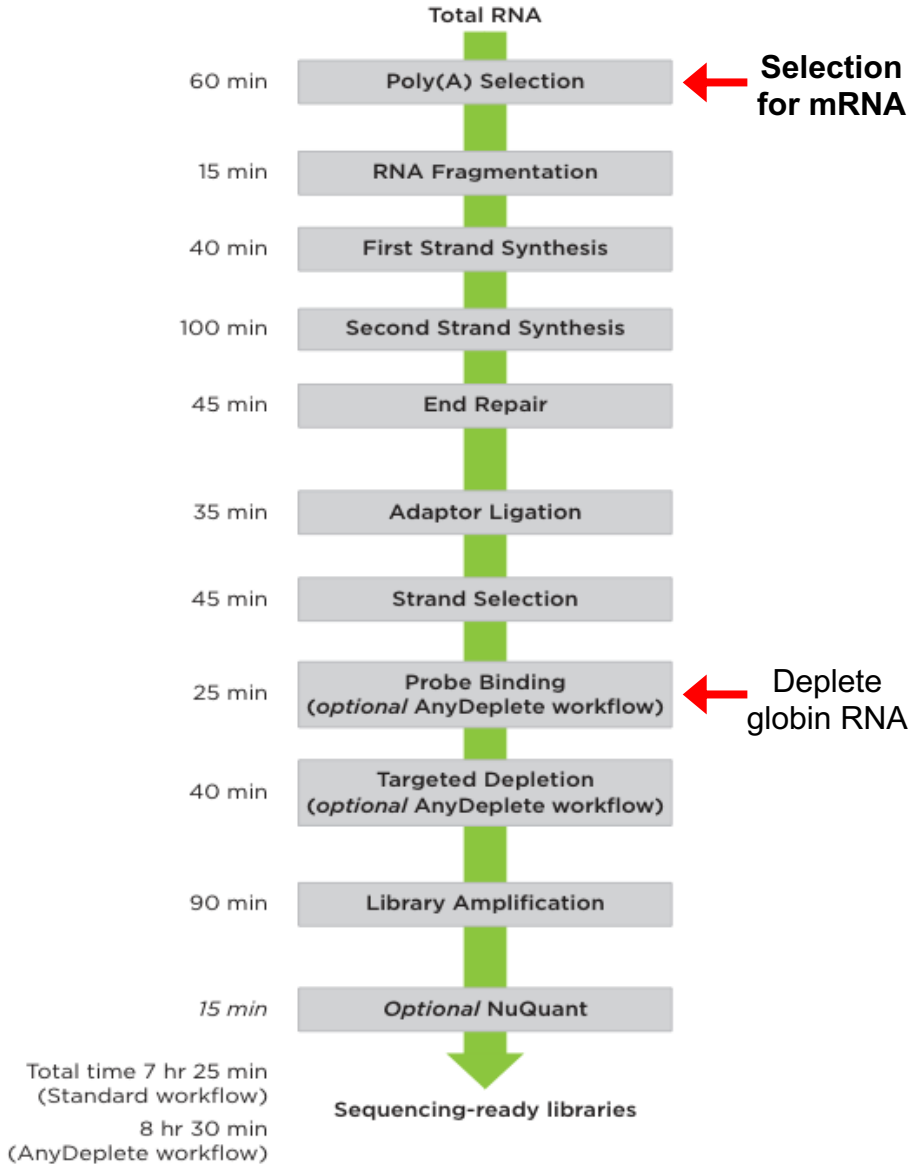
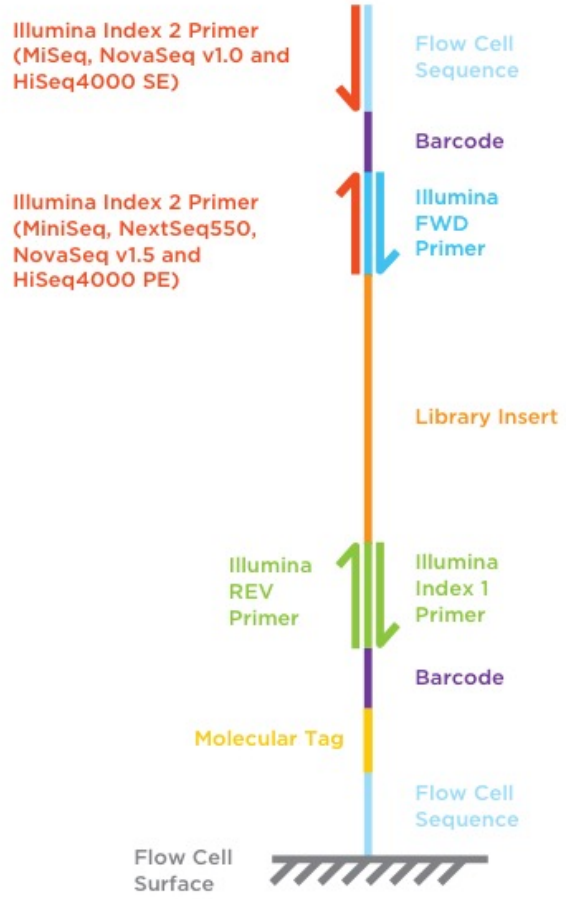


Figure 2. Universal Plus mRNA-Seq with NuQuant Library Structure.



Note: The forward read from Universal Plus mRNA-Seq with NuQuant libraries represents the sense strand. This may be opposite to stranded data from other library preparation kits and may require slight modification to the data analysis workflow. Contact Tecan NGS Technical Support for more information.

NB: Read 1 corresponds to the *sense* strand.

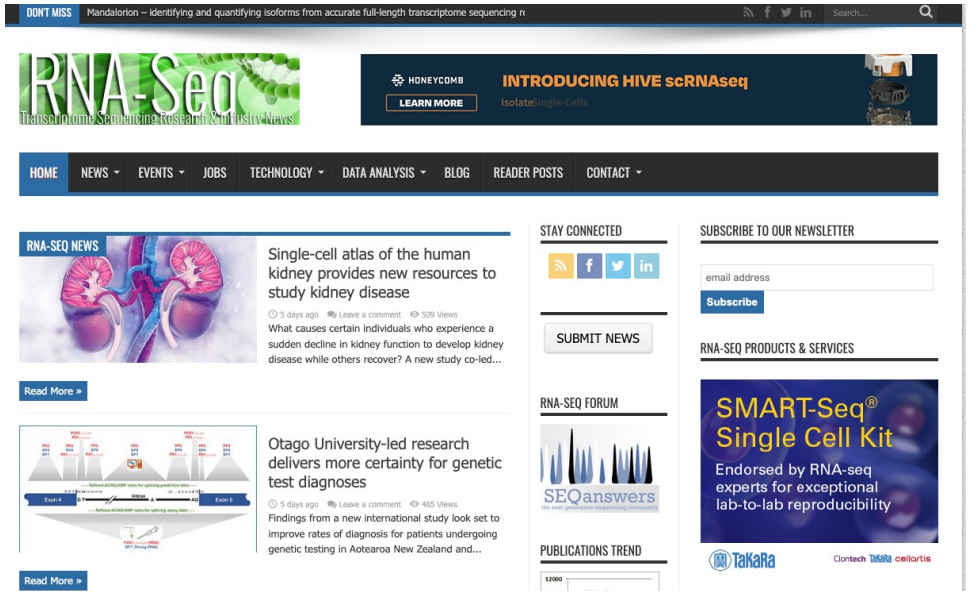
https://lifesciences.tecan.com/universal-plus-mrna-seq-library-preparation-nuquant

NGS: Useful resources to learn more

https://enseqlopedia.com/

https://www.seqanswers.com

https://sequencing.qcfail.com/



https://www.rna-seqblog.com/

Human Trisome Project: plasma proteomics using SOMAscan

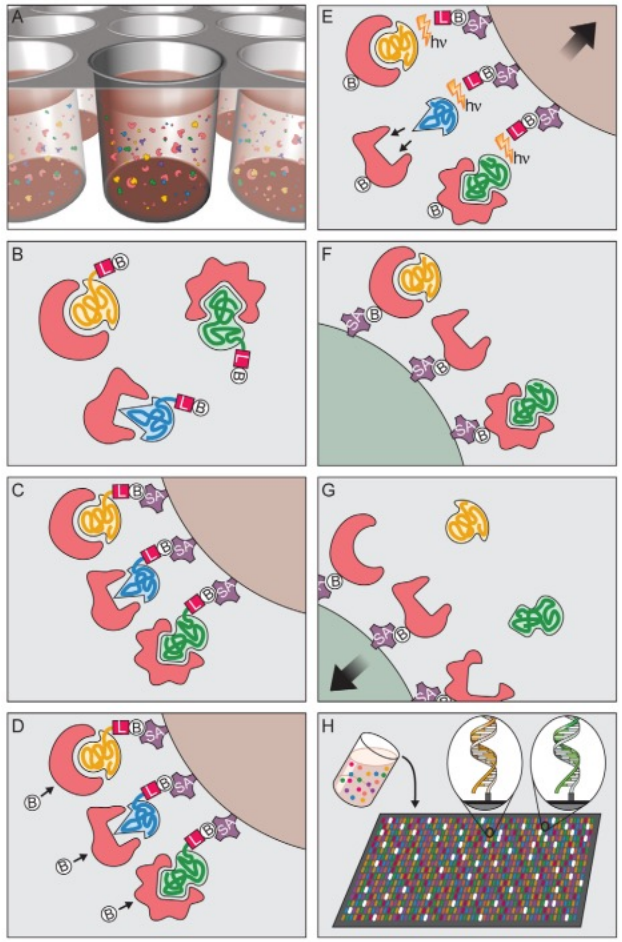
somalogic

SomaScan® Proteomics - SomaLogic®

- Multiplex proteomic assay
- Slow Off-rate Modified Aptamers
- Aptamers are short single-stranded oligonucleotides, which fold into structures that bind with high affinity and specificity to proteins, peptides, and small molecules
- Aptamers are selected *in vitro* by SELEX

Data preprocessing:

- Normalization and calibration were performed according to SOMAscan Data Standardization and File Specification Technical Note (SSM-020)
- Normalized data (***relative* fluorescence units**) in the SOMAscan® adat file format
- Imported to R using SomaDataIO R package



A. Binding. **SOMAmers and samples are mixed in 96-well microwell plates and allowed to bind**. Cognate and non-cognate SOMAmer-target protein complexes form. Free SOMAmer and protein are also present.
B. SOMAmer-protein binding: DNA-based SOMAmer molecules (gold, blue, and green) have unique shapes selected to bind to a specific protein. SOMAmers contain biotin (B), a photo-cleavable linker (L) and a fluorescent tag at the 5' end. Most SOMAmers (gold and green) bind to cognate proteins (red), but some (blue) form non-cognate complexes.
C. Catch-1. **SOMAmers are captured onto a bead coated with streptavidin (SA) which binds biotin. Un-complexed proteins are washed away**.
D. Proteins are tagged with NHS-biotin.
E. Photocleavage and kinetic challenge. UV light (hv) cleaves the linker and SOMAmers are released from beads, leaving biotin on bead. Samples are challenged with anionic competitor (dextran sulfate). Non-cognate complexes (blue SOMAmer) preferentially dissociate.
F. Catch-2 **SOMAmer-protein complexes are captured onto new avidin coated beads by protein biotin tag**. Free SOMAmers are washed away.
G. SOMAmers are released from complexes into solution at high pH.
H. Remaining **SOMAmers are quantified by hybridization to microarray containing single-stranded DNA probes complementary to SOMAmer DNA sequence**, which form a double-stranded helix. Hybridized SOMAmers are detected by fluorescent tags when the array is scanned.

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3000457/

Human Trisome Project: plasma cytokines and related immune factors

Meso Scale Discovery

V-PLEX Human Biomarker 54-Plex

Application(s)	Cytokines & Chemokines, Immunology/Inflammation
Analyte(s)	CRP, Eotaxin, Eotaxin-3, FGF (basic), GM-CSF, ICAM-1, IFN-γ, IL-1α, IL-1β, IL-1RA, IL-2, IL-3, IL-4, IL-5, IL-6, IL-7, IL-8, IL-8 (HA), IL-9, IL-10, IL-12/IL-23p40, IL-12p70, IL-13, IL-15, IL-16, IL-17A, IL-17A/F, IL-17B, IL-17C, IL-17D, IL-21, IL-22, IL-23, IL-27, IL-31, IP-10, MCP-1, MCP-4, MDC, MIP-1α, MIP-1β, MIP-3α, PIGF, SAA, TARC, Tie-2, TNF-α, TNF-β, TSLP, VCAM-1, VEGF-A, VEGF-C, VEGF-D, VEGFR-1/Flt-1
Species	Human
Instrument	MESO QuickPlex SQ 120, SECTOR Imager 2400, SECTOR Imager 6000, MESO SECTOR S 600
Plate Type	96-well
Sample Type	Serum, Plasma, Cell Culture Supernatant, Urine

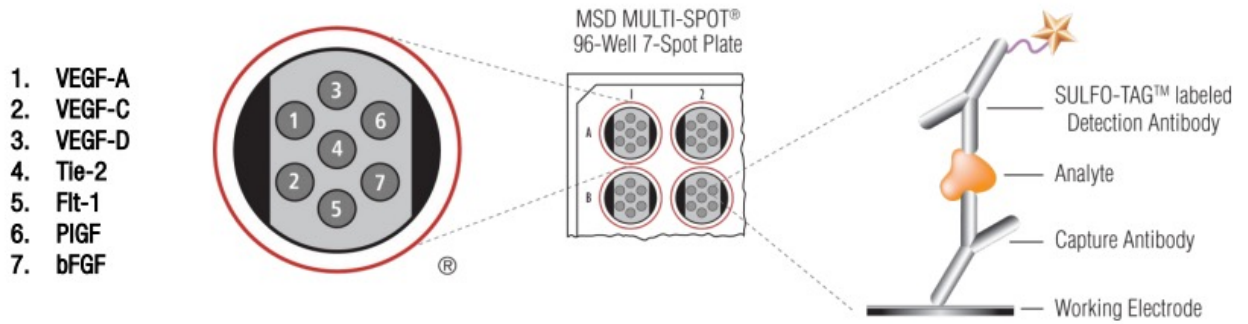


Figure 1. *Multiplex plate spot diagram showing placement of analyte capture antibodies. The numbering convention for the different spots is maintained in the software visualization tools, on the plate packaging, and in the data files.*

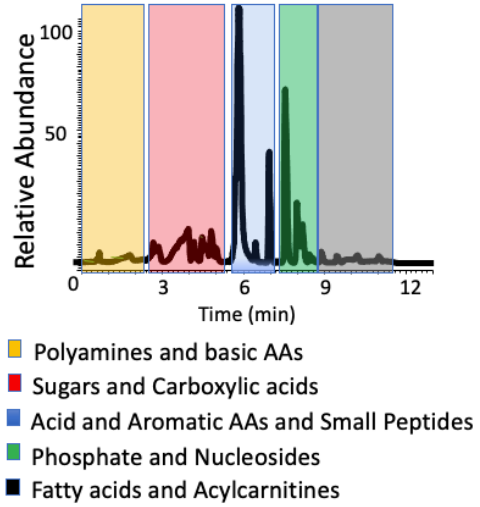
- Multiplexed sandwich ELISA
- Capture Abs printed on plate electrode
- Detection Abs conjugated with electrochemiluminescent tags
- Applied voltage → light emission
- Supplied with calibration standards for absolute quantification

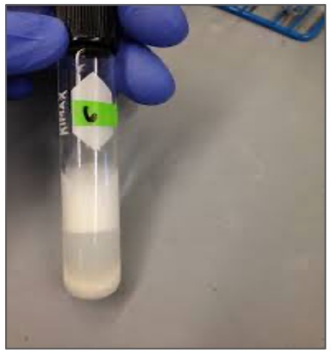
Data preprocessing:

- Output data from MESO QuickPlex SQ 120 instrument converted to ***absolute*** **concentration values** in picograms/ml using standard curve
- Analytes with >10% of values outside of detection or fit curve ranges are flagged
- For each analyte, missing values were replaced with either the minimum (if below fit curve range) or maximum (if above fit curve range) calculated concentration per plate/batch
- Means of technical duplicate wells used for subsequent analysis

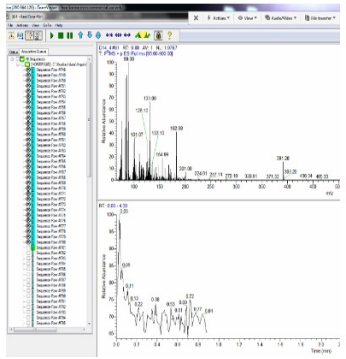
https://www.mesoscale.com/en/technical_resources/our_technology/our_immunoassays

Human Trisome Project: plasma LCMS metabolomics

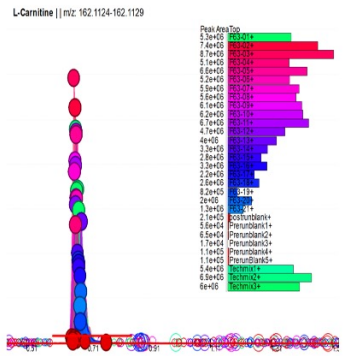




Metabolite extraction



UHPLC-MS data acquisition
in + and - ion modes



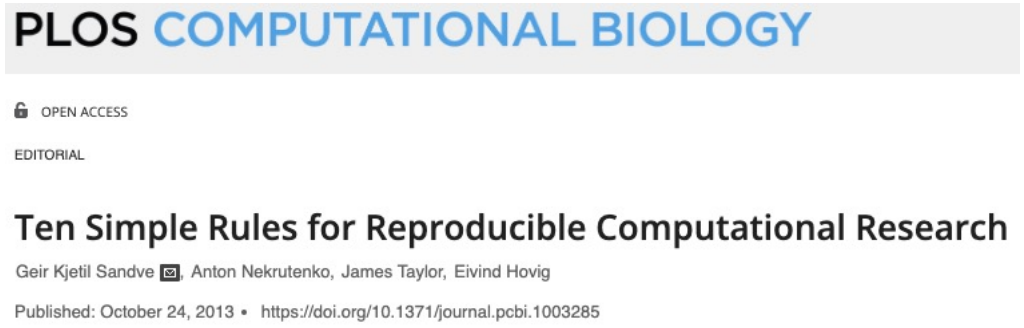
Data analysis

Data preprocessing:

- Output data are ***relative*** **abundance values** (peak intensity)
- Analytes with >10% of missing values are flagged
- For each metabolite, missing values replaced with a random value sampled from between 0 and 0.5x the minimum non-zero intensity value for that metabolite
- Data was then normalized using a scaling factor derived by dividing the global median intensity value across all metabolites by each sample median intensity

Principles of reproducible data science / bioinformatics

Principles of reproducible data science / bioinformatics



PLOS COMPUTATIONAL BIOLOGY

OPEN ACCESS

EDITORIAL

Ten Simple Rules for Reproducible Computational Research

Geir Kjetil Sandve, Anton Nekrutenko, James Taylor, Eivind Hovig

Published: October 24, 2013 • https://doi.org/10.1371/journal.pcbi.1003285

1. **For Every Result, Keep Track of How It Was Produced**
2. **Avoid Manual Data Manipulation Steps**
3. Archive the Exact Versions of All External Programs Used
4. **Version Control All Custom Scripts**
5. Record All Intermediate Results, When Possible in Standardised Formats
6. For Analyses That Include Randomness, Note Underlying Random Seeds
7. **Always Store Raw Data** behind Plots
8. Generate Hierarchical Analysis Output, Allowing Layers of Increasing Detail to Be Inspected
9. Connect Textual Statements to Underlying Results
10. Provide Public Access to Scripts, Runs, and Results

https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1003285

Principles of reproducible data science / bioinformatics


PLOS BIOLOGY

OPEN ACCESS

COMMUNITY PAGE

Best Practices for Scientific Computing

Greg Wilson, D. A. Aruliah, C. Titus Brown, Neil P. Chue Hong, Matt Davis, Richard T. Guy, Steven H. D. Haddock, Kathryn D. Huff, Ian M. Mitchell, Mark D. Plumbley, Ben Waugh, Ethan P. White, Paul Wilson

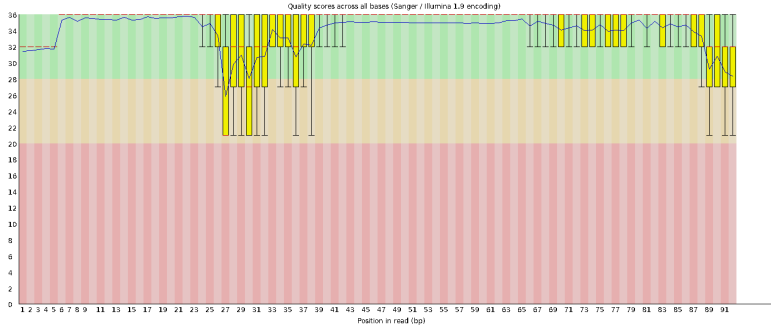
Published: January 7, 2014 • https://doi.org/10.1371/journal.pbio.1001745

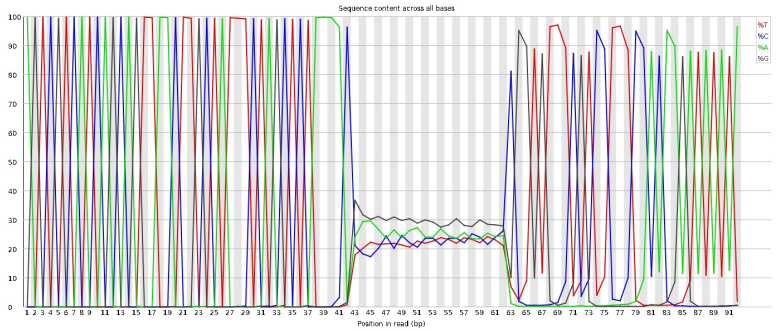
1. **Write programs for people, not computers.**
    - Make names consistent, distinctive, and meaningful.
    - Make code style and formatting consistent.
2. Let the computer do the work.
    - **Make the computer repeat tasks.**
    - **Save recent commands in a file for re-use.**
    - Use a (build) tool to automate workflows.
3. Make incremental changes.
    - **Work in small steps with frequent feedback and course correction.**
    - **Use a version control system.**
4. Don't repeat yourself (or others).
    - **Modularize code rather than copying and pasting.**
    - Re-use code instead of rewriting it.
5. Plan for mistakes.
6. Optimize software only after it works correctly.
7. Document design and purpose, not mechanics.
    - **Refactor code in preference to explaining how it works.**
    - Embed the documentation for a piece of software in that software.
8. Collaborate.

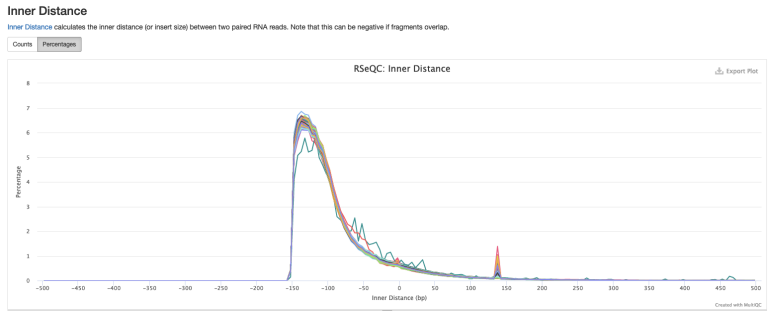
https://journals.plos.org/plosbiology/article?id=10.1371/journal.pbio.1001745

High-throughput/Next-generation sequencing primary analysis

- Can be run on local machine but usually on server/HPC/cloud
- Virtual environments/Containerization/portability (eg Conda, Docker, Singularity)
- Main steps usually:
  1. Sequence preprocessing, filtering, quality control
  2. Alignment to reference
  3. Alignment filtering, quality control
  4. Counting at gene/transcript/exon level or peak calling etc
- Don’t rely on defaults – understand/review options
- Record all options used
- Track (and archive versions) of software tools
- Collect log files (and check them
- Automate the workflow (BASH/Python/NextFlow etc)
- Collate and check QC information – MultiQC (https://multiqc.info/)

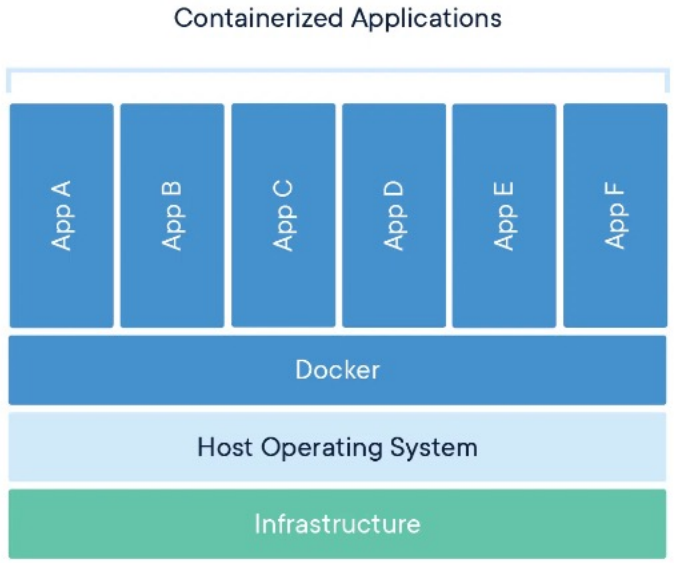






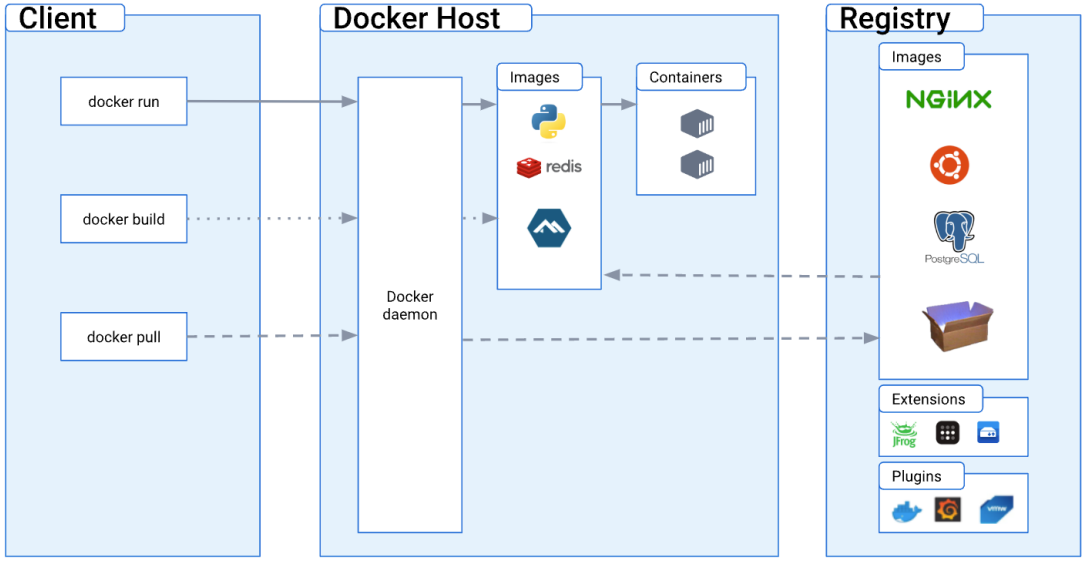
Reproducible data analysis: Containerization

- Self-contained packaging of code/app + dependencies
- Provide an isolated and minimal environment
- Portable, shareable
- Reproducible (if versions are specified)
- Containers run based on a fixed/static image i.e. not persistent



Docker

- Provides tools for building and running images
- Execution generally requires root access (often a problem on HPC)
- Larger available image repositories
  https://biocontainers.pro/
  https://rocker-project.org/



Apptainer (formerly Singularity)

- Execution does not require root access (ideal for HPC)
- Easy to convert Docker images

For more, see:
https://github.com/mattgalbraith/multiqc-docker-singularity
https://www.docker.com/resources/what-container/
https://apptainer.org/
https://f1000research.com/articles/7-742/v2
https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1008316

NGS pipeline / workflow management

- How to string together individual processing tasks into a ‘pipeline’?
- How to automate?

Challenges:

- Orchestrating data movement and processing
- Managing task dependencies (incl. modularity)
- Resource allocation (incl. parallelization)
- Tracking data provenance and settings
- Tracking and troubleshooting execution errors
- Consider also portability, scalability, robustness, debugging, monitoring, reproducibility, documentation, support

nextflow

https://www.nextflow.io/index.html

Curated pipelines: https://nf-co.re/

CWL **Common Workflow Language**

https://www.commonwl.org/

Design considerations for workflow management systems use in production genomics research and the clinic. *Scientific Reports* 11:21680 2021.

https://www.nature.com/articles/s41598-021-99288-8

Data acceptance / ingest / validation

- Using NGS here as example
- Equally important to all data types
- Record **ALL** information
  store sample manifests with data
  store PDF copies of emails with data

Data acceptance / ingest / validation

Experiment metadata

Excel spreadsheet to facilitate entry by *anyone*

Useful to include other potentially important variables, eg RIN, batch, Age, Sex, Treatment etc for import to R later

Additional tabs for eg MD5 sum checks, sample_locations.txt, multi-lane merging

Sample Name (Given to sequencing core)	Sequencing Core / Facility	Run name	Library Protocol (ChIP; DNA; RNA total/polyA; kit etc)	Seq Protocol (platform; read length etc)	Library Type (Paired End / Single End)	Library Kit	Strand specific (yes/no)	Replicate	Final name
COVS000238	CU_Anschutz	L3_plate1_48_96_08172020	polyA RNA-seq	NovaSeq6000, 150bp	Paired end	Tecan/Nugen Universal Plus mRNA-Seq with NuQuant, Hu Globin AnyDeplete #0521B-A01	yes	1	COVS000238
COVS000267	CU_Anschutz	L3_plate1_48_96_08172020	polyA RNA-seq	NovaSeq6000, 150bp	Paired end	Tecan/Nugen Universal Plus mRNA-Seq with NuQuant, Hu Globin AnyDeplete #0521B-A01	yes	1	COVS000267
COVS000383	CU_Anschutz	L3_plate1_48_96_08172020	polyA RNA-seq	NovaSeq6000, 150bp	Paired end	Tecan/Nugen Universal Plus mRNA-Seq with NuQuant, Hu Globin AnyDeplete #0521B-A01	yes	1	COVS000383
COVS000586	CU_Anschutz	L3_plate1_48_96_08172020	polyA RNA-seq	NovaSeq6000, 150bp	Paired end	Tecan/Nugen Universal Plus mRNA-Seq with NuQuant, Hu Globin AnyDeplete #0521B-A01	yes	1	COVS000586
COVS000644	CU_Anschutz	L3_plate1_48_96_08172020	polyA RNA-seq	NovaSeq6000, 150bp	Paired end	Tecan/Nugen Universal Plus mRNA-Seq with NuQuant, Hu Globin AnyDeplete #0521B-A01	yes	1	COVS000644
COVS000818	CU_Anschutz	L3_plate1_48_96_08172020	polyA RNA-seq	NovaSeq6000, 150bp	Paired end	Tecan/Nugen Universal Plus mRNA-Seq with NuQuant, Hu Globin AnyDeplete #0521B-A01	yes	1	COVS000818
COVS001050	CU_Anschutz	L3_plate1_48_96_08172020	polyA RNA-seq	NovaSeq6000, 150bp	Paired end	Tecan/Nugen Universal Plus mRNA-Seq with NuQuant, Hu Globin AnyDeplete #0521B-A01	yes	1	COVS001050
COVS001311	CU_Anschutz	L3_plate1_48_96_08172020	polyA RNA-seq	NovaSeq6000, 150bp	Paired end	Tecan/Nugen Universal Plus mRNA-Seq with NuQuant, Hu Globin AnyDeplete #0521B-A01	yes	1	COVS001311
COVS001485	CU_Anschutz	L3_plate1_48_96_08172020	polyA RNA-seq	NovaSeq6000, 150bp	Paired end	Tecan/Nugen Universal Plus mRNA-Seq with NuQuant, Hu Globin AnyDeplete #0521B-A01	yes	1	COVS001485

Other important questions

Were samples sequenced across multiple runs/lanes? (will require merging)

Are the samples multiplexed/demultiplexed?

RNAseq: Does the library kit produce reads corresponding to forward (Tecan/Nugen)
or reverse (Illumina TruSeq) strand?

Quick check of read lengths / N+1 read:

```
zcat XXXX.fastq.gz | head -n40 | paste - - - - | cut -f2,2 | awk '{print length}'
```

https://www.biostars.org/p/162922/
http://seqanswers.com/forums/showthread.php?t=61997

Data acceptance / ingest / validation

Quick check of quality scores:

```
zcat XXXX.fastq.gz | head -n400000 | paste - - - - | cut -f 4 | sed -e $'s/\(.\)/\\1\\\n/g' | sort | uniq –c

 100000
 564888 ,
 816734 :
     58 #
13718320 F
```

= NovaSeq data

Illumina quality score binning

https://www.illumina.com/documents/products/whitepapers/whitepaper_datacompression.pdf
https://www.illumina.com/content/dam/illumina-marketing/documents/products/appnotes/novaseq-hiseq-q30-app-note-770-2017-010.pdf
https://www.biostars.org/p/77248/

Table 1: Q-Score Bins for an Optimized 8-Level Mapping

Quality Score Bins	Example of Empirically Mapped Quality Scores*
N (no call)	N (no call)
2–9	6
10–19	15
20–24	22
25–29	27
30–34	33
35–39	37
≥ 40	40

By replacing the quality scores between 19 and 25 with a new score of 22, data storage space is conserved.
*The mapped quality score of each bin (except "N") is subject to change depending on individual Q-tables.

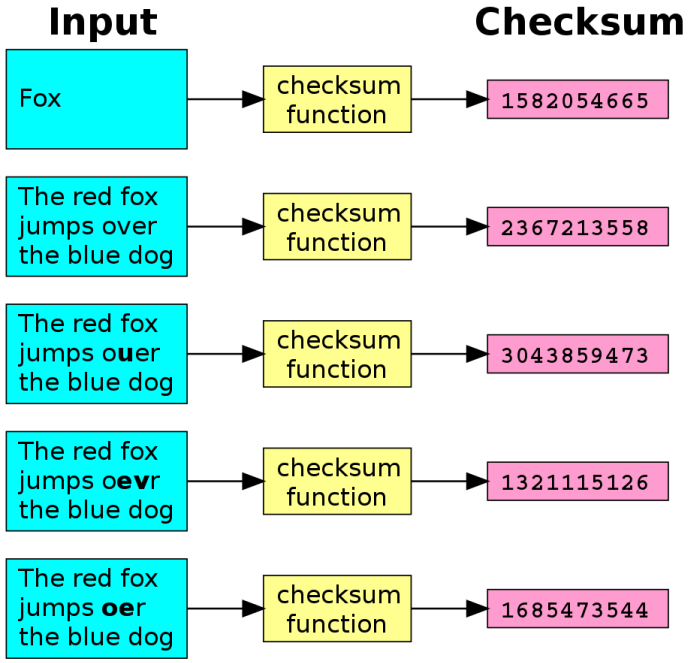
Can be an issue for certain pipelines eg DADA2 error correction in Qiime pipeline for 16S rRNA sequencing

Data acceptance / ingest / validation
File checksums

Check the integrity of downloaded files!!

- Many tools will not give notice/error if fastq files are not intact
- Checksums can be used as digital fingerprints
- Request checksums from sequencing facility (e.g. MD5)
- Generate and validate checksums for downloaded files:
  `md5sum -c original_files.md5 >> md5_check_downloaded.txt`

MD5 checksum	File
934f4295a688c65c41236f1b580b1b14	CM_42_D_FRRB202353951-1a_H5H3CDSXY_L4_1.fq.gz
f1fae3b67ba6547a4db4c6f01f0486f1	CM_42_D_FRRB202353951-1a_H5H3CDSXY_L4_2.fq.gz
665d9e0f880fec406ed722a37a58ae71	CM_42_D_FRRB202353951-1a_H5JLKDSXY_L1_1.fq.gz
c47b52155ae6307d99972b183e1c28a6	CM_42_D_FRRB202353951-1a_H5JLKDSXY_L1_2.fq.gz
9160909263019ebf85e0c3e2e50c161d	CM_42_N_FRRB202353952-1a_H5H3CDSXY_L4_1.fq.gz
3c5d4887ac2990ddf57dbbf5b67089a2	CM_42_N_FRRB202353952-1a_H5H3CDSXY_L4_2.fq.gz
ef0201d4c94c24b5688e567dc9f2663c	CM_62_D_FRRB202353959-1a_H5H3CDSXY_L4_1.fq.gz
66040a055fe0b1267cee4b96bc550f8a	CM_62_D_FRRB202353959-1a_H5H3CDSXY_L4_2.fq.gz
1af8c62055ec17d44f49746740e5ae27	CM_62_D_FRRB202353959-1a_H5JLKDSXY_L1_1.fq.gz
1e11a28a961ee50b9fbbfe9fb55fb3ec	CM_62_D_FRRB202353959-1a_H5JLKDSXY_L1_2.fq.gz



NGS Workflow/QC Examples and Intro to RNAseq analysis in R

Matthew Galbraith
Linda Crnic Institute for Down Syndrome

Data Science for Developing Scholars in Down Syndrome Research (DS3) 2025

INCLUDE
INvestigation of Co-occurring conditions across the Lifespan to Understand Down syndromE

Linda Crnic Institute for Down Syndrome
UNIVERSITY OF COLORADO ANSCHUTZ MEDICAL CAMPUS

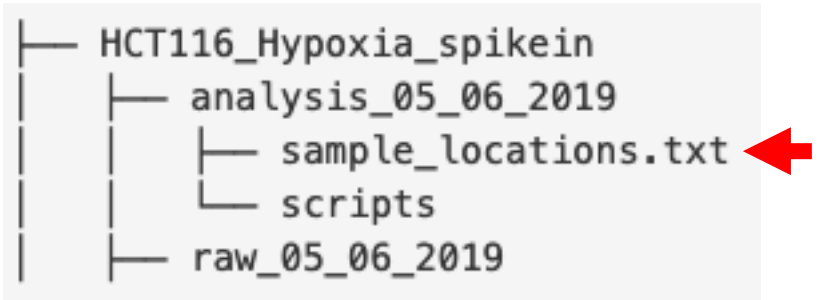
GLOBAL
DOWN SYNDROME FOUNDATION®

NGS Pipeline workflow

How to tell your pipeline about your files and sample names?

sample_locations.txt

COVS000238	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_merged_09_09_2020/COVS000238_S17_L003_R1_001_merged.fastq.gz	read1
COVS000238	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_merged_09_09_2020/COVS000238_S17_L003_R2_001_merged.fastq.gz	read2
COVS000267	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_merged_09_09_2020/COVS000267_S18_L003_R1_001_merged.fastq.gz	read1
COVS000267	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_merged_09_09_2020/COVS000267_S18_L003_R2_001_merged.fastq.gz	read2
COVS000296	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_09_08_2020/L3_plate1_48_96_08172020/COVS000296_S46_L003_R1_001.fastq.gz	read1
COVS000296	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_09_08_2020/L3_plate1_48_96_08172020/COVS000296_S46_L003_R2_001.fastq.gz	read2
COVS000325	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_09_08_2020/L3_plate1_48_96_08172020/COVS000325_S19_L003_R1_001.fastq.gz	read1
COVS000325	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_09_08_2020/L3_plate1_48_96_08172020/COVS000325_S19_L003_R2_001.fastq.gz	read2
COVS000354	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_09_08_2020/L3_plate1_48_96_08172020/COVS000354_S20_L003_R1_001.fastq.gz	read1
COVS000354	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_09_08_2020/L3_plate1_48_96_08172020/COVS000354_S20_L003_R2_001.fastq.gz	read2
COVS000383	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_merged_09_09_2020/COVS000383_S45_L003_R1_001_merged.fastq.gz	read1
COVS000383	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_merged_09_09_2020/COVS000383_S45_L003_R2_001_merged.fastq.gz	read2
COVS000499	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_09_08_2020/L3_plate1_48_96_08172020/COVS000499_S21_L003_R1_001.fastq.gz	read1
COVS000499	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_09_08_2020/L3_plate1_48_96_08172020/COVS000499_S21_L003_R2_001.fastq.gz	read2
COVS000557	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_09_08_2020/L1_plate2_08172020/COVS000557_S1_L001_R1_001.fastq.gz	read1
COVS000557	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_09_08_2020/L1_plate2_08172020/COVS000557_S1_L001_R2_001.fastq.gz	read2
COVS000586	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_merged_09_09_2020/COVS000586_S22_L003_R1_001_merged.fastq.gz	read1
COVS000586	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_merged_09_09_2020/COVS000586_S22_L003_R2_001_merged.fastq.gz	read2
COVS000644	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_merged_09_09_2020/COVS000644_S3_L003_R1_001_merged.fastq.gz	read1
COVS000644	/gpfs/jespinosa/shared/COVID_projects/RNAseq/raw_merged_09_09_2020/COVS000644_S3_L003_R2_001_merged.fastq.gz	read2



NGS Pipeline workflow example: custom pipeline

BaSH_seq Pipeline (Bash and Slurm on HPC)

- Custom pipeline written entirely in BASH
- Runs on HPC cluster and custom Google Cloud Compute (GCP)
- Uses slurm workload manager/scheduler
- Single- or paired-end reads
- Modular: can handle multiple data types and tools (incl. containers)
- Error checking + log collection
- Can run specific stages / batches
- Scalable (2 – 100s of samples)

```
analysis_05_06_2019
├── QC
├── Sample_HCT116_HIF1Ako_Hypox_2
│   ├── Alignments
│   ├── Counts
│   ├── Processed
│   ├── Raw
│   └── std_out_err
├── Sample_HCT116_HIF1Ako_Hypox_3
├── Sample_HCT116_HIF1Ako_Norm_2
├── Sample_HCT116_HIF1Ako_Norm_3
├── Sample_HCT116_WT_Hypox_2
├── Sample_HCT116_WT_Hypox_3
├── Sample_HCT116_WT_Norm_2
├── Sample_HCT116_WT_Norm_3
├── sample_locations.txt
└── scripts
```

Setup

analysis_setup.sh script
- Creates sample dirs.
- Creates symlinks to raw
- Writes the submit script

→

Per run

submit_all.sh script
- Submitted to slurm queue
- Submits a pipeline script job per sample (subshells)
- Generates summary log

→

Per sample (parallel)

RNAseq_pipeline.sh
- Creates/checks output dirs.
- Checks for input files
- Submits stage jobs (slurm)
- Waits for previous stage
- Error checking (slurm)
- Exits on error

↖ ↓ Stage job submission/exit logs

→

Per sample (parallel)

Stage scripts (eg 0_PRE_FASTQC.sh)
- Run individual processing steps
- Takes ARGS from pipeline script
- Automatic naming of output files

↓

Script copy → *std_out_err*
Detailed stage logs → *std_out_err*

https://github.com/mattgalbraith/BaSH_seq

NGS Pipeline workflow: BaSH_seq main pipeline script

RNAseq_pipeline_v0.7.sh

```
##### VARIABLES IN THIS SECTION CAN BE EDITED #####
## global pipeline variables ##
        SEQ_TYPE=PE                         # <PE/SE> for paired-end or single end
        STRAND_TYPE=strand-specific-fwd     # <strand-specific-fwd/strand-specific-rev/unstranded>  # use fwd for Nugen TTseq round 1 library; use rev f
        READ_LENGTH=150                     # <50/75/150>  # used to trim n+1 read if present
        PIPELINE_TYPE=RNAseq
        SPIKE_IN=none                       # <none/both/dm/ercc> external RNA spike-in details; used to select correct index(s) and gtf(s) for alignmen
        PIPELINE_SCRIPTS_DIR=$SHARED/PipeLinesScripts/RNAseq/stageScripts  # THIS COULD BE IMPLEMENTED AS ARG
        QC_DIR_NAME="$THIS_ANALYSIS_DIR"/QC
        ALIGNMENT_DIRNAME=="$THIS_ANALYSIS_DIR"/Sample_"$THIS_SAMPLE_NAME"/Alignments   # NEED TO IMPLEMENT IN STAGE 4
        COUNTS_DIRNAME="$THIS_ANALYSIS_DIR"/Sample_"$THIS_SAMPLE_NAME"/Counts
        TRACKS_DIRNAME="$THIS_ANALYSIS_DIR"/Tracks
        # FASTQR1_FILE=see switch and check below
        # FASTQR2_FILE=see switch and check below
        COMMENTS="This version is current default.
"     # Add comments explaining this version of the analysis

## STAGE-SPECIFIC VARIABLES TO EDIT HERE ##
# 0 PRE_FASTQC
        # uncommon options are either defaults or set in 0_PRE_FASTQC.sh

# 1 FASTQ_MCF
        # common options
        MIN_SEQ_LENGTH=30       # -l option; default = 19
        MIN_QUAL=10             # -q option; default = 10; should change for Novoseq data with binned q-scores
        MIN_ADAPTER=1.6           # -s option; Log2 scale for adapter minimum-length-match; default=2.2 ie ~4.5)
        MIN_PERC_OCCUR=0.25     # -t option; % occurance threshold before adapter clipping; default=0.25)
        MAX_PERC_DIFF=10        # -p option; Maximum adapter difference percentage; default=10)
        CONTAMINANTS_FASTA="$SHARED"/references/Contaminants/contaminants.fa
        #uncommon options are either defaults or set in 1_FASTQ_MCF.sh
        # see also: https://github.com/ExpressionAnalysis/ea-utils/blob/wiki/FastqMcf.md
```

NGS Pipeline workflow: BaSH_seq main pipeline script

RNAseq_pipeline_v0.7.sh

```
##### PIPELINE LOGFILE #####
LOGFILE="$THIS_ANALYSIS_DIR"/Sample_"$THIS_SAMPLE_NAME"/std_out_err/"$PIPELINE_TYPE"-"$(basename "$THIS_ANALYSIS_DIR")"-Stage"$START_AT_STAGE"-"$END_AT_
echo "
#####################################
"$PIPELINE_TITLE" v"$PIPELINE_VERSION"
Stage(s): "$START_AT_STAGE"-"$END_AT_STAGE"
Run: $(date)
Sample: "$THIS_SAMPLE_NAME"
#####################################
COMMENTS:
"$COMMENTS"

GLOBAL SETTINGS:
From command line (via analysis_setup.sh):
Sample name: "$THIS_SAMPLE_NAME"
Analysis directory: "$THIS_ANALYSIS_DIR"
User account: "$THIS_USER_ACCOUNT"
Begin at stage: "$START_AT_STAGE"
Finish at stage: "$END_AT_STAGE"
Set within "$PIPELINE_TITLE"v"$PIPELINE_VERSION".sh
Sequencing type: "$SEQ_TYPE"
Strand type: "$STRAND_TYPE"
Read length: "$READ_LENGTH"
Pipeline scripts directory: "$PIPELINE_SCRIPTS_DIR"
QC directory: "$QC_DIR_NAME"
Tracks directory: "$TRACKS_DIRNAME"
Read 1 fastq file: "$FASTQR1_FILE"
Read 2 fastq file: "$FASTQR2_FILE"
" > "$LOGFILE"
```

NGS Pipeline workflow: BaSH_seq main pipeline script

RNAseq_pipeline_v0.7.sh

```
# 0) PRE_FASTQC
        # run the stage
        STAGE_NAME="0-PRE_FASTQ"

        if [ $START_AT_STAGE -eq 0 ]
        then
                echo -e "${blue}Starting stage "$STAGE_NAME" for "$THIS_SAMPLE_NAME"${NC}
                "

                if [ ! -d "$QC_DIR_NAME" ]
                then
                        echo "making QC directory: "$QC_DIR_NAME"
                        "
                        mkdir --parents "$QC_DIR_NAME"
                fi

                #sh $PIPELINE_SCRIPTS_DIR/0_PRE_FASTQC.sh
                #Usage: 0_PRE_FASTQC.sh <SAMPLE_DIR> <SAMPLE_NAME> <FASTQR1_FILE/FASTQR2_FILE> <QC_DIR_NAME> <OUT_DIR_NAME> <THREADS>

                # Read 1
                JOB_NAME=stage-"$STAGE_NAME"_R1-"$PIPELINE_TYPE"-"$THIS_SAMPLE_NAME"
                STAGE_OUTPUT="$THIS_ANALYSIS_DIR"/Sample_"$THIS_SAMPLE_NAME"/std_out_err/"$JOB_NAME"
                STAGE_ERROR="$THIS_ANALYSIS_DIR"/Sample_"$THIS_SAMPLE_NAME"/std_out_err/"$JOB_NAME"
                echo -e "${blue}Running "$JOB_NAME"${NC}
                "
                sbatch -W \
                        --account="$THIS_USER_ACCOUNT" \
                        --job-name="$JOB_NAME" \
                        --output="$STAGE_OUTPUT".%j.%N.out \
                        --error="$STAGE_ERROR".%j.%N.err \
                        --partition=defq \
                        --time=10:00:00 \
                        --mem=32G \
                        --nodes=1 \
                        --cpus-per-task=8 \
                        --ntasks=1 \
                        --wrap="\
                                sh "$PIPELINE_SCRIPTS_DIR"/0_PRE_FASTQC.sh "$THIS_ANALYSIS_DIR"/Sample_"$THIS_SAMPLE_NAME"/ "$THIS_SAMPLE_NAME" "$FASTQR1
                                "

                # Catch output status
                OUTPUT_STATUS=$?
```

NGS Pipeline workflow: BaSH_seq main pipeline script

RNAseq_pipeline_v0.7.sh

```
                # Catch output status
                OUTPUT_STATUS=$?

                # Get Job ID
                getJobID "$JOB_NAME"

                # Check for SLURM error: <function> <$STAGE_ERROR> <$JOB_ID>
                slurmErrorCheck "$STAGE_ERROR" "$JOB_ID"

                # Check for exit code error: <function> <$JOB_ID>
                slurmExitCodeCheck "$JOB_ID"

                # Copy standard output and standard error to logfile: <function> <Title> <$STAGE_ERROR> <$JOB_ID>
                copyOutErrLog "$STAGE_NAME"_read2 "$STAGE_ERROR" "$JOB_ID"

                # check output status: <function> <stage label>
                checkOutputStatus "$STAGE_NAME"_read2

        fi

        echo -e "${green}Stage "$STAGE_NAME" complete for "$THIS_SAMPLE_NAME"${NC}
        "
        # update the stage
        START_AT_STAGE=$(($START_AT_STAGE+1))
        STAGE_NAME=""
        STAGE_OUTPUT=""
        STAGE_ERROR=""
fi

# check if we need to exit
if [ $END_AT_STAGE -eq $(($START_AT_STAGE-1)) ]
then
        echo -e "------\n${green}"$PIPELINE_TITLE" run ending at stage "$END_AT_STAGE"${NC}\n------\n"
        exit 0
fi
```

NGS Pipeline workflow: BaSH_seq stages

Stage	Tool	URL
Sequencing QC and filtering		
0_PRE_FASTQC	fastqc	https://www.bioinformatics.babraham.ac.uk/projects/fastqc/
1_FASTQ_MCF_BB	bbduk	https://jgi.doe.gov/data-and-tools/bbtools/bb-tools-user-guide/bbduk-guide/
	fastq-mcf	https://expressionanalysis.github.io/ea-utils/
2_POST_FASTQC	fastqc	https://www.bioinformatics.babraham.ac.uk/projects/fastqc/
3_FASTQ SCREEN	fastq_screen	https://www.bioinformatics.babraham.ac.uk/projects/fastq_screen/
Alignment to reference genome and QC		
4_HISAT2	hisat2	https://daehwankimlab.github.io/hisat2/
5_ADD_RGID	Picard	https://broadinstitute.github.io/picard/
6_MAPQ_FILTER	Samtools	http://www.htslib.org/
7_SORT_BAM	Samtools	http://www.htslib.org/
8_MARK_DUPLICATES	Picard	https://broadinstitute.github.io/picard/
9_PE_ALIGNMENT_METRICS	Samtools	http://www.htslib.org/
10_RSEQC	RSeQC	http://rseqc.sourceforge.net/
Gene-level read summarization		
11_HTSEQ_COUNT	Htseq-count	https://htseq.readthedocs.io/en/release_0.11.1/count.html
11_FEATURE_COUNTS	FeatureCounts	http://subread.sourceforge.net/
Genomic fragments/peak calling etc		
11_HOMER	HOMER	http://homer.ucsd.edu/homer/

NGS Pipeline workflow: Containers for each stage

https://github.com/mattgalbraith/fastqc-docker-singularity
https://github.com/mattgalbraith/bbtools-docker-singularity
https://github.com/mattgalbraith/eautils-docker-singularity
https://github.com/mattgalbraith/fastqscreen-docker-singularity
https://github.com/mattgalbraith/hisat2-docker-singularity
https://github.com/mattgalbraith/bwa-docker-singularity
https://github.com/mattgalbraith/bowtie2-docker-singularity
https://github.com/mattgalbraith/picard-docker-singularity
https://github.com/mattgalbraith/samtools-docker-singularity
https://github.com/mattgalbraith/rseqc-docker-singularity
https://github.com/mattgalbraith/htseq2-docker-singularity
https://github.com/mattgalbraith/subread-docker-singularity
https://github.com/mattgalbraith/homer-docker-singularity

NGS Pipeline workflow: Final directory structure

```
├── HCT116_Hypoxia_spikein
│   ├── analysis_05_06_2019
│   │   ├── QC
│   │   ├── Sample_HCT116_HIF1Ako_Hypox_2
│   │   │   ├── Alignments
│   │   │   ├── Counts
│   │   │   ├── Processed
│   │   │   ├── Raw
│   │   │   └── std_out_err
│   │   ├── Sample_HCT116_HIF1Ako_Hypox_3
│   │   ├── Sample_HCT116_HIF1Ako_Norm_2
│   │   ├── Sample_HCT116_HIF1Ako_Norm_3
│   │   ├── Sample_HCT116_WT_Hypox_2
│   │   ├── Sample_HCT116_WT_Hypox_3
│   │   ├── Sample_HCT116_WT_Norm_2
│   │   ├── Sample_HCT116_WT_Norm_3
│   │   ├── sample_locations.txt
│   │   └── scripts
│   ├── raw_05_06_2019
│   │   ├── H202SC19030981
│   │   └── Novogene_QC
```

Gene-level count file *per sample*

NGS Pipeline workflow: check pipeline run summary log

```
----------
Starting RNAseq_pipeline v0.6 stage(s) 4 to 4 for HCT116_WT_Norm_2
----------
FASTQR1 file found: /gpfs/jespinosa/shared/Projects2/RNAseq/HCT116_Hypoxia_spikein/analysis_05_06_2019//Sample_HCT116_WT_Norm_2/Raw/HCT116_WT_Norm_2_R1.fastq.gz

Running RNAseq pipeline in paired-end (PE) mode

FASTQR2 file found: /gpfs/jespinosa/shared/Projects2/RNAseq/HCT116_Hypoxia_spikein/analysis_05_06_2019//Sample_HCT116_WT_Norm_2/Raw/HCT116_WT_Norm_2_R2.fastq.gz

Running stage-4-TOPHAT2-ALIGN-RNAseq-HCT116_WT_Norm_2

Submitted batch job 373249
stage-4-TOPHAT2-ALIGN-RNAseq-HCT116_WT_Norm_2 complete.

------
RNAseq_pipeline run ending at stage 4
------


----------
Starting RNAseq_pipeline v0.6 stage(s) 4 to 4 for HCT116_WT_Hypox_2
----------
FASTQR1 file found: /gpfs/jespinosa/shared/Projects2/RNAseq/HCT116_Hypoxia_spikein/analysis_05_06_2019//Sample_HCT116_WT_Hypox_2/Raw/HCT116_WT_Hypox_2_R1.fastq.gz

Running RNAseq pipeline in paired-end (PE) mode

FASTQR2 file found: /gpfs/jespinosa/shared/Projects2/RNAseq/HCT116_Hypoxia_spikein/analysis_05_06_2019//Sample_HCT116_WT_Hypox_2/Raw/HCT116_WT_Hypox_2_R2.fastq.gz

Running stage-4-TOPHAT2-ALIGN-RNAseq-HCT116_WT_Hypox_2

Submitted batch job 373248
stage-4-TOPHAT2-ALIGN-RNAseq-HCT116_WT_Hypox_2 complete.

------
RNAseq_pipeline run ending at stage 4
------
```

NGS Pipeline workflow: check pipeline run sample logs

Standard out
Pipeline version
Global settings
Stage settings
Stage start/stop times
Auto detection messages

```
#####################################
RNAseq_pipeline v0.6
Stage(s): 4-4
Run: Tue May  7 15:28:36 MDT 2019
Sample: HCT116_WT_Norm_2
#####################################
COMMENTS:
This version is for HCT116 Hypoxia RNA-seq data with Drosophila and ERCC spike-ins.

GLOBAL SETTINGS:
From command line (via analysis_setup.sh):
Sample name: HCT116_WT_Norm_2
Analysis directory: /gpfs/jespinosa/shared/Projects2/RNAseq/HCT116_Hypoxia_spikein/analysis_05_06_2019/
User account: espinosalab-galbrama
Begin at stage: 4
Finish at stage: 4
Set within RNAseq_pipelinev0.6.sh
Sequencing type: PE
Strand type: strand-specific-rev
Read length: 150
Pipeline scripts directory: /gpfs/jespinosa/shared/Matt/PipeLinesScripts/RNAseq/stageScripts
QC directory: /gpfs/jespinosa/shared/Projects2/RNAseq/HCT116_Hypoxia_spikein/analysis_05_06_2019//QC
Tracks directory: /gpfs/jespinosa/shared/Projects2/RNAseq/HCT116_Hypoxia_spikein/analysis_05_06_2019//Tracks
Read 1 fastq file: /gpfs/jespinosa/shared/Projects2/RNAseq/HCT116_Hypoxia_spikein/analysis_05_06_2019//Sample_HCT116_WT_Norm_2/Raw/HCT116_WT_Norm_2_R1.fastq.gz
Read 2 fastq file: /gpfs/jespinosa/shared/Projects2/RNAseq/HCT116_Hypoxia_spikein/analysis_05_06_2019//Sample_HCT116_WT_Norm_2/Raw/HCT116_WT_Norm_2_R2.fastq.gz

--------------------
Stage 4-TOPHAT2-ALIGN
--------------------
OUTPUT FILE: /gpfs/jespinosa/shared/Projects2/RNAseq/HCT116_Hypoxia_spikein/analysis_05_06_2019//Sample_HCT116_WT_Norm_2/std_out_err/stage-4-TOPHAT2-ALIGN-RNAseq-HCT116_WT_Norm_2.373249.cubipmbigmem01.out
Script name: 4_TOPHAT2_ALIGN.sh
Script version: 0.2
Arguments for 4_TOPHAT2_ALIGN.sh:
(1) SEQ_TYPE: PE
(2) SAMPLE_DIR: /gpfs/jespinosa/shared/Projects2/RNAseq/HCT116_Hypoxia_spikein/analysis_05_06_2019//Sample_HCT116_WT_Norm_2/
(3) SAMPLE_NAME: HCT116_WT_Norm_2
(4) FASTQR1_FILE: /gpfs/jespinosa/shared/Projects2/RNAseq/HCT116_Hypoxia_spikein/analysis_05_06_2019//Sample_HCT116_WT_Norm_2/Processed/trimmed_HCT116_WT_Norm_2_R1.fastq.gz
(5) FASTQR2_FILE: /gpfs/jespinosa/shared/Projects2/RNAseq/HCT116_Hypoxia_spikein/analysis_05_06_2019//Sample_HCT116_WT_Norm_2/Processed/trimmed_HCT116_WT_Norm_2_R2.fastq.gz
(6) BOWTIE_INDEX: /gpfs/jespinosa/shared/Matt/Refs/Bowtie2_UCSC_hg19_dm6_ERCC/hg19_dm6_ERCC.genome
(7) GTF: /gpfs/jespinosa/shared/Matt/Refs/Bowtie2_UCSC_hg19_dm6_ERCC/hg19_dm6_genes.gtf
(8) TRANSCRIPTOME_INDEX: /gpfs/jespinosa/shared/Matt/Refs/Bowtie2_UCSC_hg19_dm6_ERCC/transcriptome_index_bt2/hg19_dm6_known
(9) STRAND_TYPE: strand-specific-rev
(10) MATE_INNER_DIST: 50
(11) MATE_STD_DEV: 20
(12) BAM_OUT_FILENAME: HCT116_WT_Norm_2.mapped.no-rgid.bam
(13) OUT_DIR_NAME: Alignments
(14) THREADS: 12
tophat2 version: echo v2.1.1
tophat2 options:
-p 12
--b2-sensitive
--keep-fasta-order
--no-coverage-search
--max-multihits 10
--library-type
--GTF /gpfs/jespinosa/shared/Matt/Refs/Bowtie2_UCSC_hg19_dm6_ERCC/hg19_dm6_genes.gtf
--transcriptome-index=/gpfs/jespinosa/shared/Matt/Refs/Bowtie2_UCSC_hg19_dm6_ERCC/transcriptome_index_bt2/hg19_dm6_known
samtools version: samtools 1.5 Using htslib 1.5 Copyright (C) 2017 Genome Research Ltd.

Creating directory: /gpfs/jespinosa/shared/Projects2/RNAseq/HCT116_Hypoxia_spikein/analysis_05_06_2019//Sample_HCT116_WT_Norm_2//Alignments
Setting --library-type to fr-firststrand
4_TOPHAT2_ALIGN.sh STARTED AT:  Tue May 7 15:28:39 MDT 2019 [JOB_ID: 373249 NODE_NAME: cubipmbigmem01]

Running tophat22 in paired-end mode for HCT116_WT_Norm_2...


4_TOPHAT2_ALIGN.sh ENDED AT:  Tue May 7 19:45:28 MDT 2019 [JOB_ID: 373249 NODE_NAME: cubipmbigmem01]
```

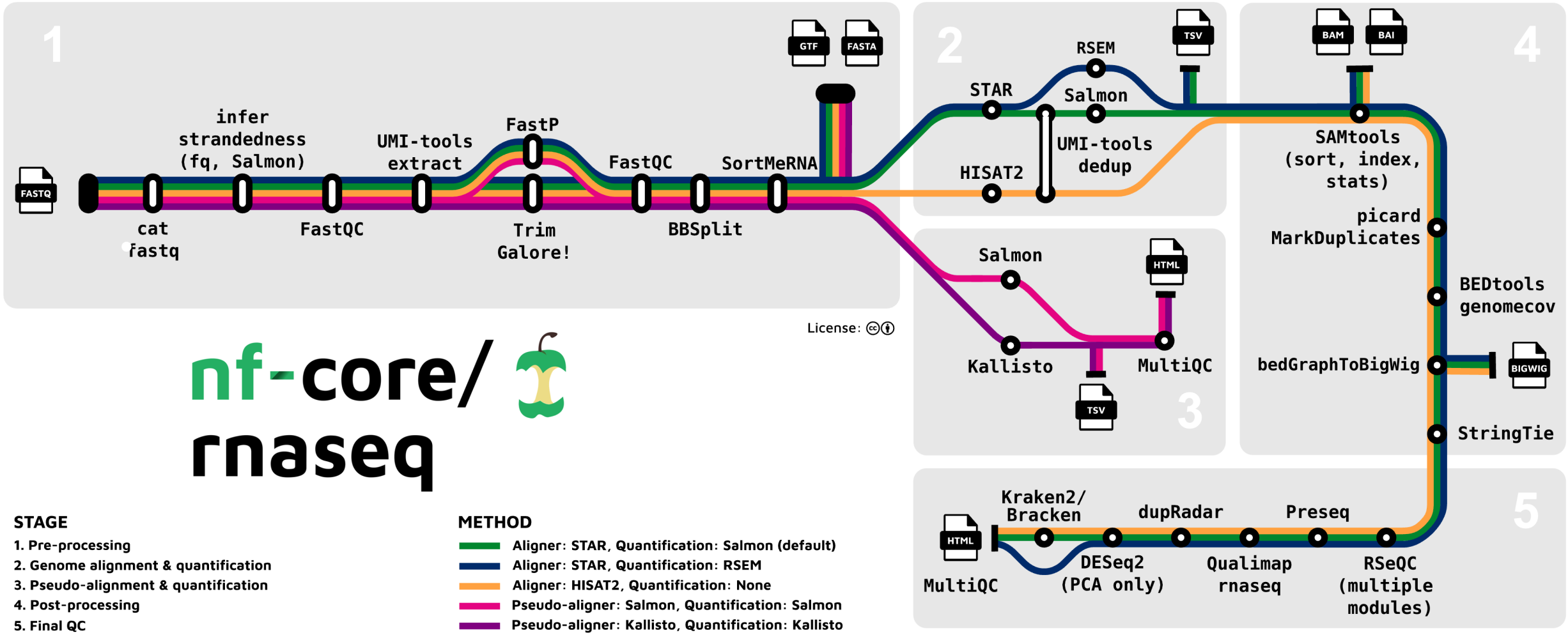
NGS Pipeline workflow: check pipeline run sample logs

Standard error: program messages and errors

```
ERROR FILE: /gpfs/jespinosa/shared/Projects2/RNAseq/HCT116_Hypoxia_spikein/analysis_05_06_2019//Sample_HCT116_WT_Norm_2/std_out_err/stage-4-TOPHAT2-ALIGN-RNAseq-HCT116_WT_Norm_2.373249.cubipmbigmem01.err

[2019-05-07 15:28:40] Beginning TopHat run (v2.1.1)
-----------------------------------------------
[2019-05-07 15:28:40] Checking for Bowtie
                Bowtie version:        2.2.9.0
[2019-05-07 15:28:40] Checking for Bowtie index files (transcriptome)..
[2019-05-07 15:28:40] Checking for Bowtie index files (genome)..
[2019-05-07 15:28:40] Checking for reference FASTA file
[2019-05-07 15:28:40] Generating SAM header for /gpfs/jespinosa/shared/Matt/Refs/Bowtie2_UCSC_hg19_dm6_ERCC/hg19_dm6_ERCC.genome
[2019-05-07 15:28:42] Reading known junctions from GTF file
[2019-05-07 15:28:46] Preparing reads
        left reads: min. length=30, max. length=150, 49847648 kept reads (4 discarded)
       right reads: min. length=30, max. length=150, 49847652 kept reads (0 discarded)
[2019-05-07 15:58:46] Using pre-built transcriptome data..
[2019-05-07 15:58:47] Mapping left_kept_reads to transcriptome hg19_dm6_known with Bowtie2
[2019-05-07 16:51:27] Mapping right_kept_reads to transcriptome hg19_dm6_known with Bowtie2
[2019-05-07 17:42:02] Resuming TopHat pipeline with unmapped reads
[2019-05-07 17:42:02] Mapping left_kept_reads.m2g_um to genome hg19_dm6_ERCC.genome with Bowtie2
[2019-05-07 18:00:52] Mapping left_kept_reads.m2g_um_seg1 to genome hg19_dm6_ERCC.genome with Bowtie2 (1/6)
[2019-05-07 18:03:09] Mapping left_kept_reads.m2g_um_seg2 to genome hg19_dm6_ERCC.genome with Bowtie2 (2/6)
[2019-05-07 18:05:27] Mapping left_kept_reads.m2g_um_seg3 to genome hg19_dm6_ERCC.genome with Bowtie2 (3/6)
[2019-05-07 18:07:45] Mapping left_kept_reads.m2g_um_seg4 to genome hg19_dm6_ERCC.genome with Bowtie2 (4/6)
[2019-05-07 18:10:02] Mapping left_kept_reads.m2g_um_seg5 to genome hg19_dm6_ERCC.genome with Bowtie2 (5/6)
[2019-05-07 18:12:21] Mapping left_kept_reads.m2g_um_seg6 to genome hg19_dm6_ERCC.genome with Bowtie2 (6/6)
[2019-05-07 18:14:36] Mapping right_kept_reads.m2g_um to genome hg19_dm6_ERCC.genome with Bowtie2
[2019-05-07 18:32:37] Mapping right_kept_reads.m2g_um_seg1 to genome hg19_dm6_ERCC.genome with Bowtie2 (1/6)
[2019-05-07 18:34:34] Mapping right_kept_reads.m2g_um_seg2 to genome hg19_dm6_ERCC.genome with Bowtie2 (2/6)
[2019-05-07 18:36:42] Mapping right_kept_reads.m2g_um_seg3 to genome hg19_dm6_ERCC.genome with Bowtie2 (3/6)
[2019-05-07 18:38:53] Mapping right_kept_reads.m2g_um_seg4 to genome hg19_dm6_ERCC.genome with Bowtie2 (4/6)
[2019-05-07 18:41:14] Mapping right_kept_reads.m2g_um_seg5 to genome hg19_dm6_ERCC.genome with Bowtie2 (5/6)
[2019-05-07 18:43:32] Mapping right_kept_reads.m2g_um_seg6 to genome hg19_dm6_ERCC.genome with Bowtie2 (6/6)
[2019-05-07 18:45:48] Searching for junctions via segment mapping
[2019-05-07 18:51:54] Retrieving sequences for splices
[2019-05-07 18:53:32] Indexing splices
Building a SMALL index
[2019-05-07 18:53:54] Mapping left_kept_reads.m2g_um_seg1 to genome segment_juncs with Bowtie2 (1/6)
[2019-05-07 18:54:22] Mapping left_kept_reads.m2g_um_seg2 to genome segment_juncs with Bowtie2 (2/6)
[2019-05-07 18:54:46] Mapping left_kept_reads.m2g_um_seg3 to genome segment_juncs with Bowtie2 (3/6)
[2019-05-07 18:55:13] Mapping left_kept_reads.m2g_um_seg4 to genome segment_juncs with Bowtie2 (4/6)
[2019-05-07 18:55:39] Mapping left_kept_reads.m2g_um_seg5 to genome segment_juncs with Bowtie2 (5/6)
[2019-05-07 18:56:04] Mapping left_kept_reads.m2g_um_seg6 to genome segment_juncs with Bowtie2 (6/6)
[2019-05-07 18:56:31] Joining segment hits
[2019-05-07 18:59:23] Mapping right_kept_reads.m2g_um_seg1 to genome segment_juncs with Bowtie2 (1/6)
[2019-05-07 18:59:54] Mapping right_kept_reads.m2g_um_seg2 to genome segment_juncs with Bowtie2 (2/6)
[2019-05-07 19:00:26] Mapping right_kept_reads.m2g_um_seg3 to genome segment_juncs with Bowtie2 (3/6)
[2019-05-07 19:00:56] Mapping right_kept_reads.m2g_um_seg4 to genome segment_juncs with Bowtie2 (4/6)
[2019-05-07 19:01:27] Mapping right_kept_reads.m2g_um_seg5 to genome segment_juncs with Bowtie2 (5/6)
[2019-05-07 19:01:59] Mapping right_kept_reads.m2g_um_seg6 to genome segment_juncs with Bowtie2 (6/6)
[2019-05-07 19:02:26] Joining segment hits
[2019-05-07 19:05:13] Reporting output tracks
-----------------------------------------------
[2019-05-07 19:38:39] A summary of the alignment counts can be found in ./tophat_out/align_summary.txt
[2019-05-07 19:38:39] Run complete: 04:09:59 elapsed
```

NGS Pipeline workflow example: NextFlow pipeline

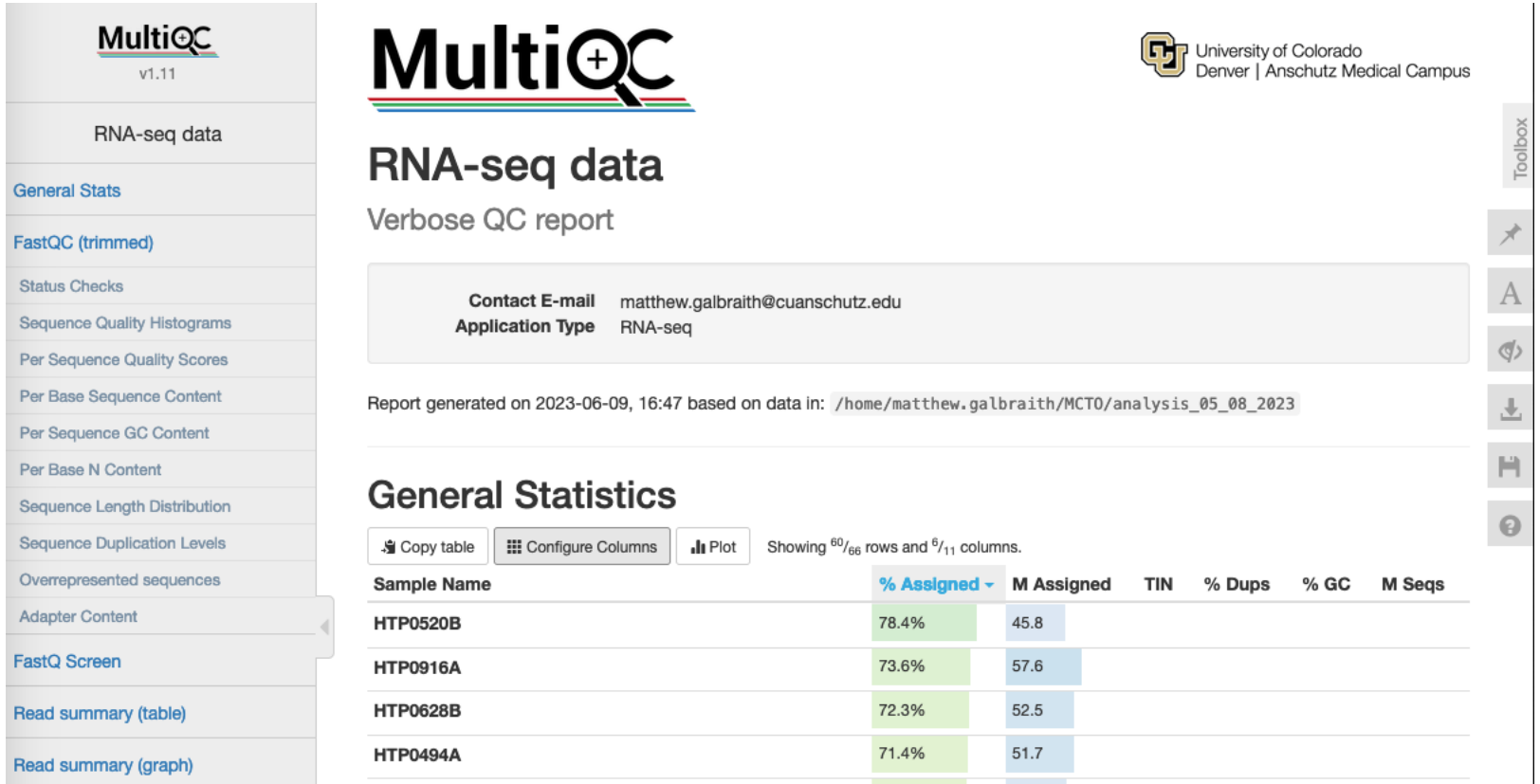


https://nf-co.re/rnaseq/3.19.0/

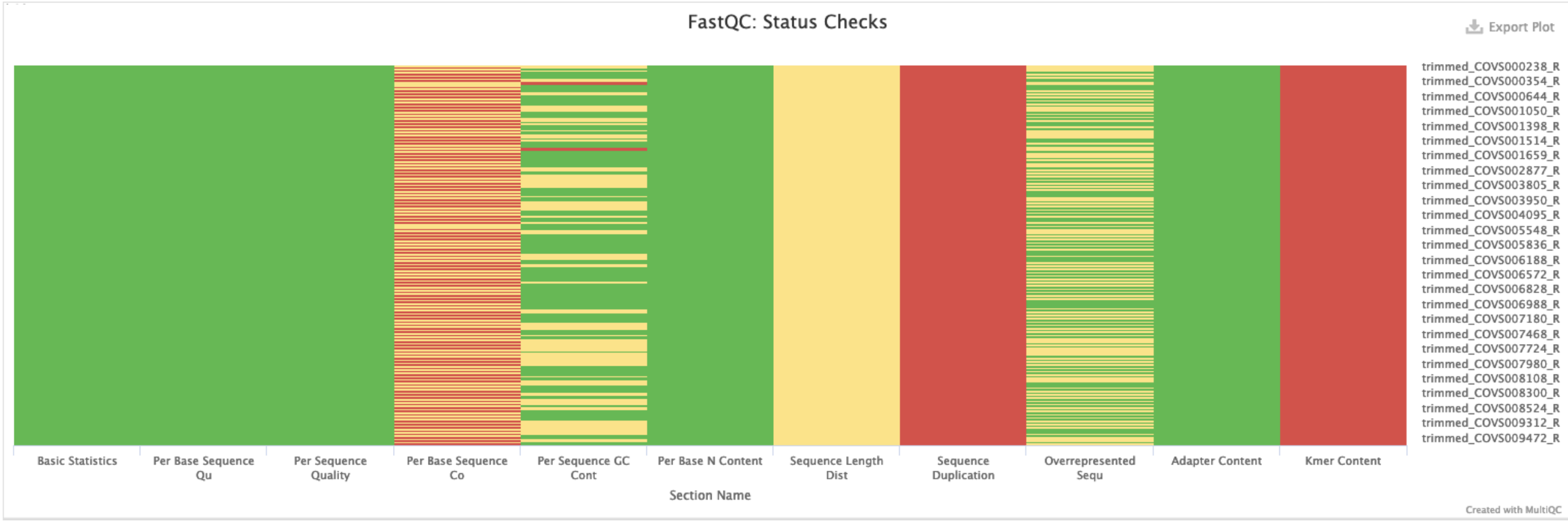
Sequencing quality control: MultiQC

MultiQC: Summarize analysis results for multiple tools and samples in a single report
Philip Ewels, Måns Magnusson, Sverker Lundin and Max Käller
Bioinformatics (2016) doi: 10.1093/bioinformatics/btw354 PMID: 27312411
https://multiqc.info/

Works with 100+ pre/post alignment QC tools



Sequencing quality control: MultiQC



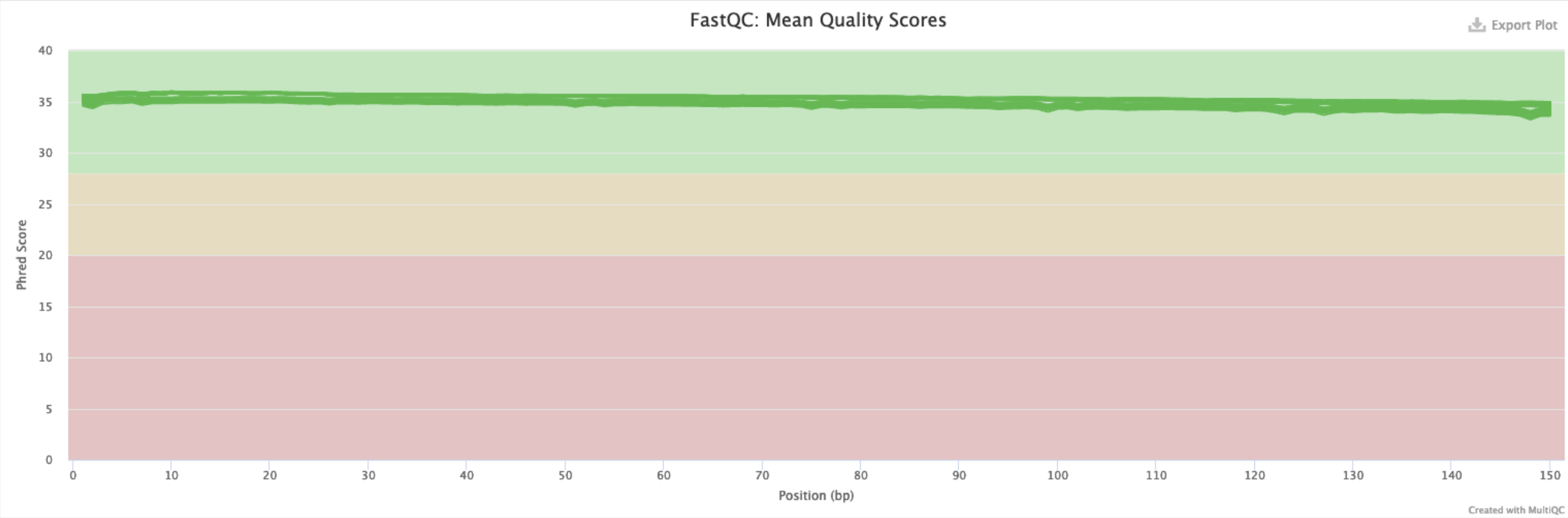
Sequencing quality control: MultiQC

Sequence Quality Histograms 230

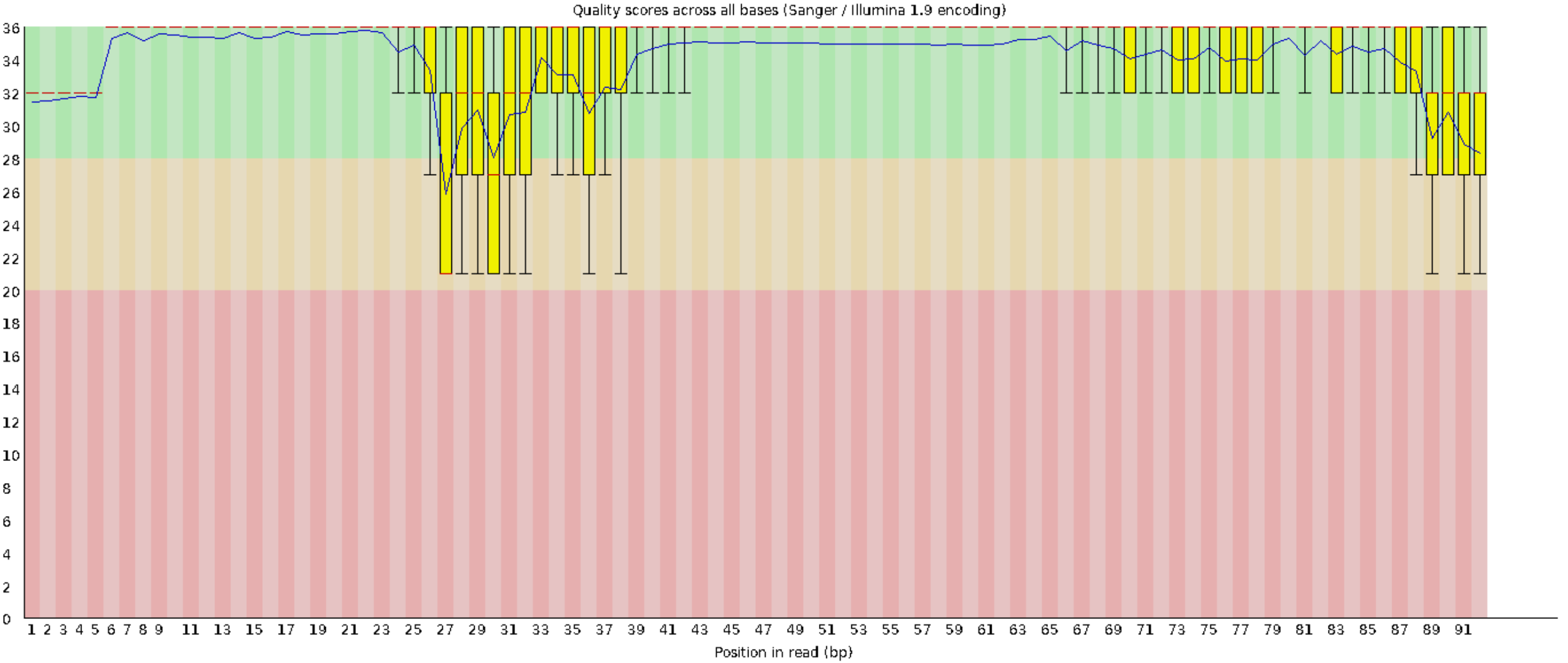
Help

The mean quality value across each base position in the read.

Y-Limits: on



CRISPR screen (amplicon sequencing)



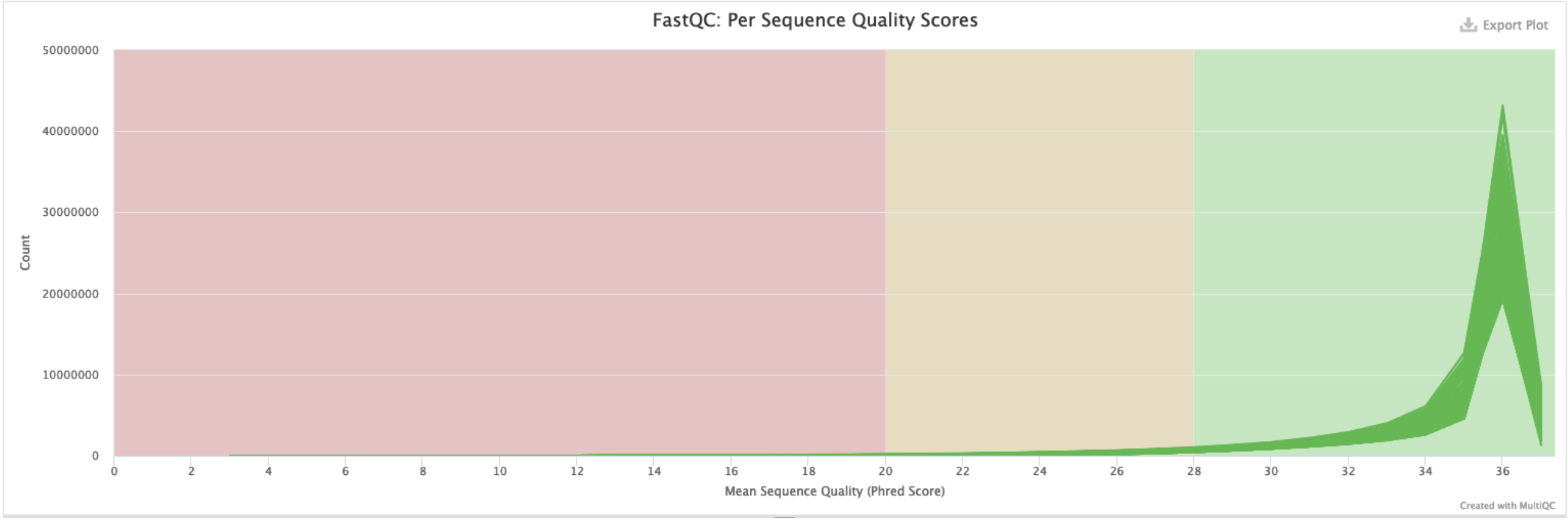
Sequencing quality control: MultiQC

Per Sequence Quality Scores 230

Help

The number of reads with average quality scores. Shows if a subset of reads has poor quality.

Y-Limits: on



Sequencing quality control: MultiQC

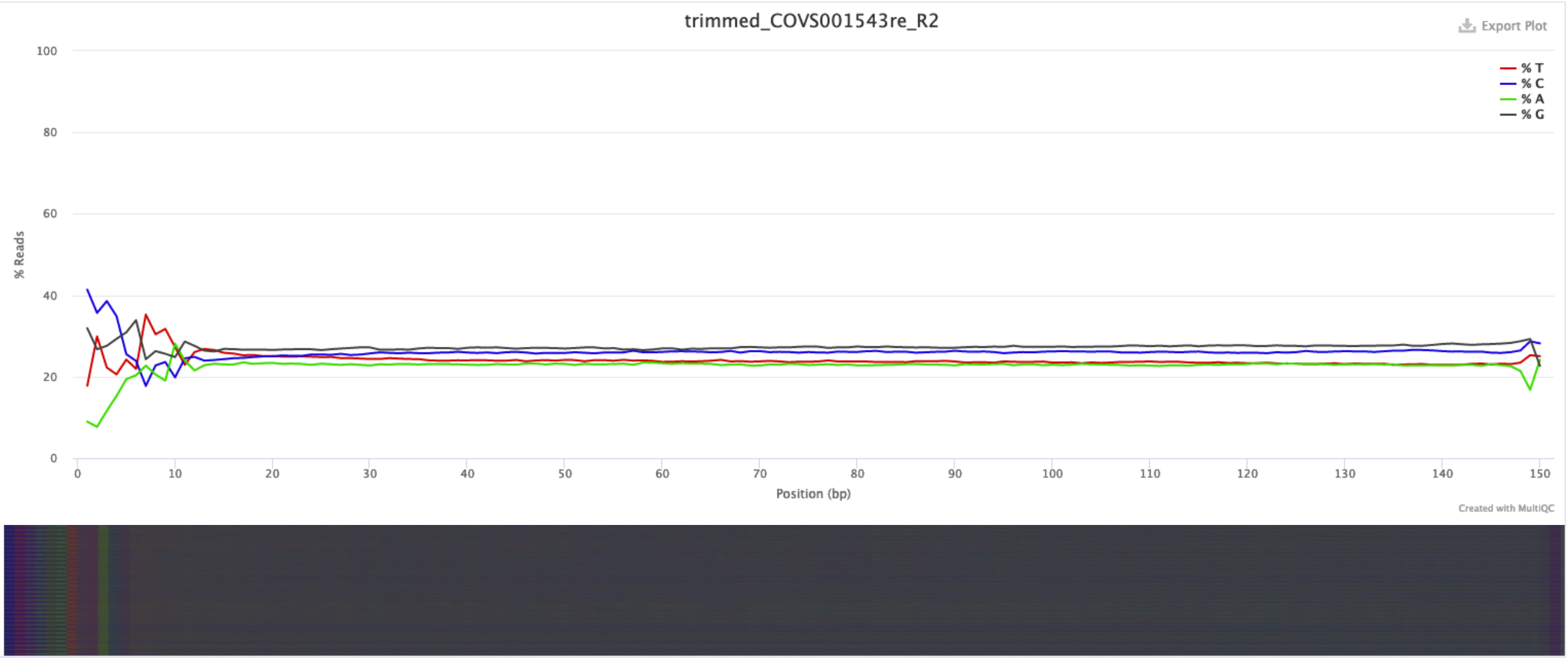
Per Base Sequence Content

140 134

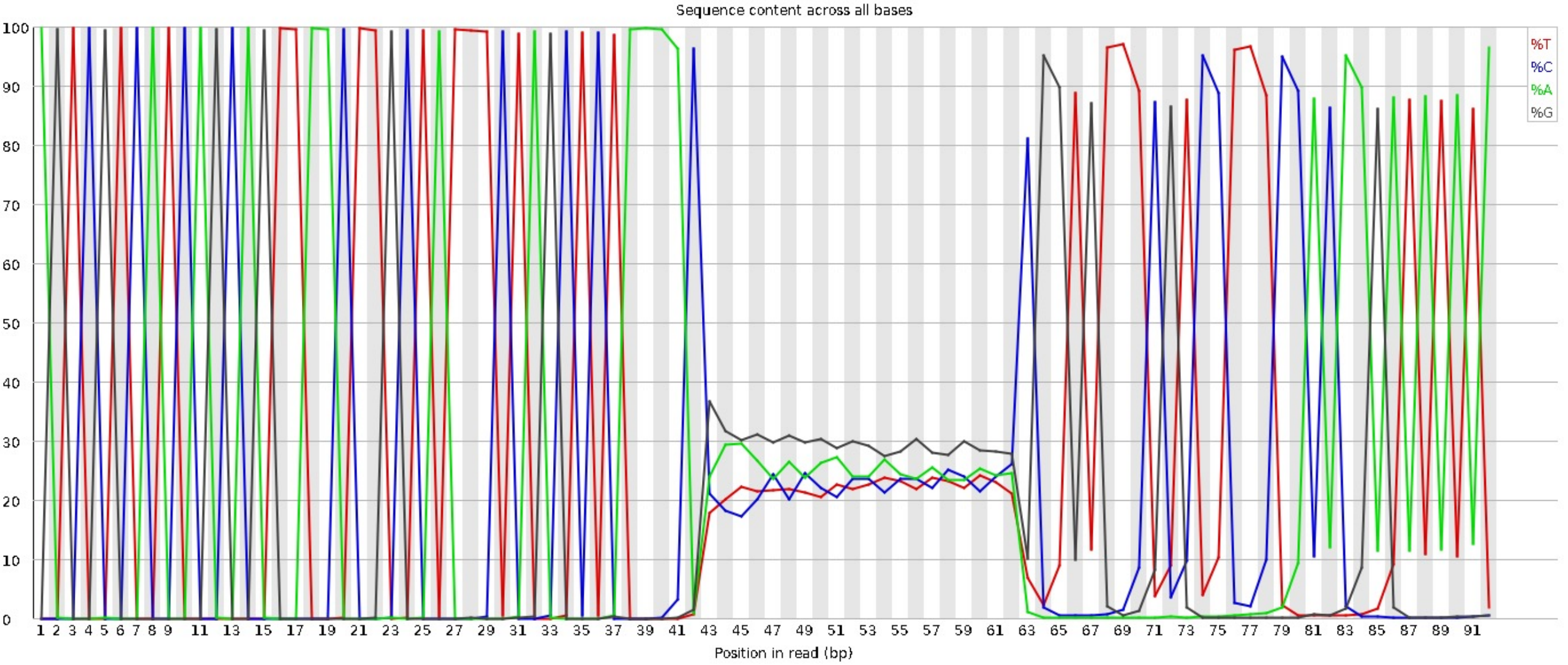
Help

The proportion of each base position for which each of the four normal DNA bases has been called.

Back to overview heatmap | « Prev | Next »

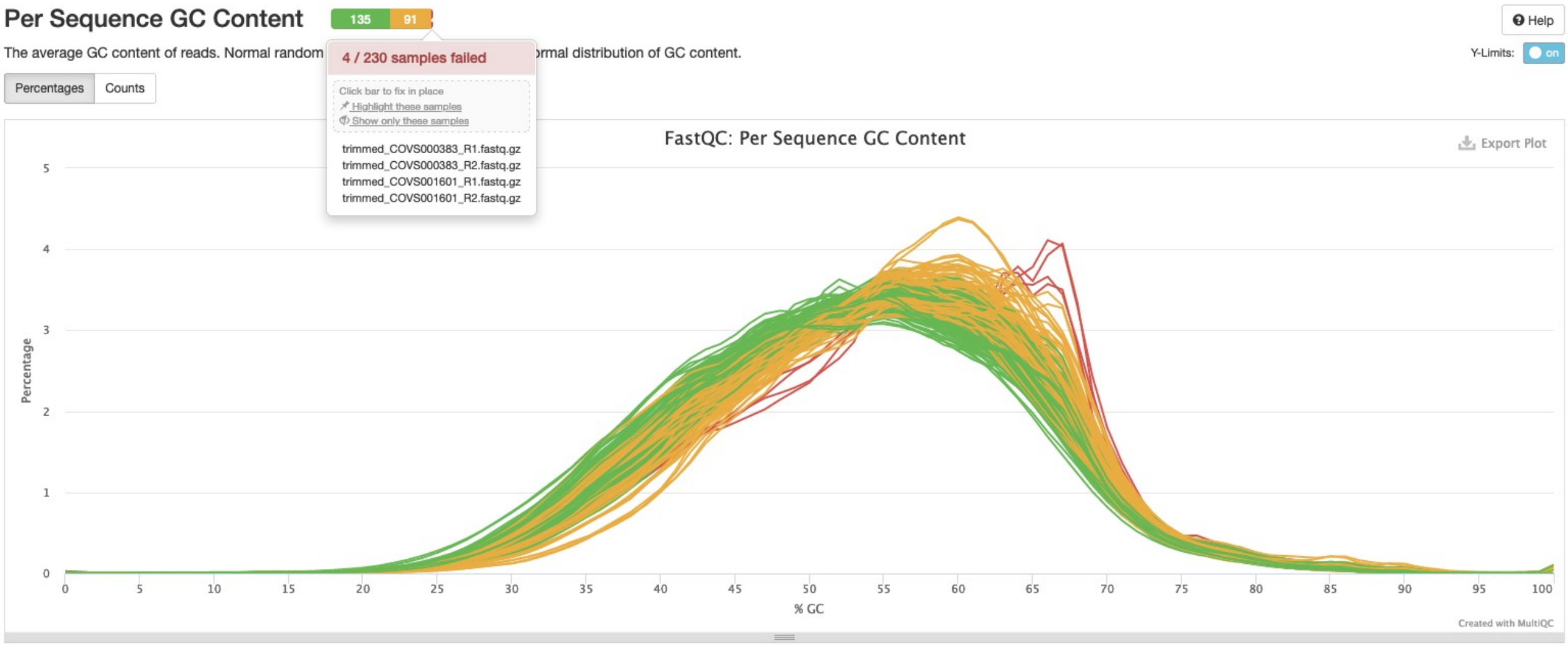


CRISPR screen (amplicon sequencing)



https://sequencing.qcfail.com/articles/biased-sequence-composition-can-lead-to-poor-quality-data-on-illumina-sequencers/

Sequencing quality control: MultiQC



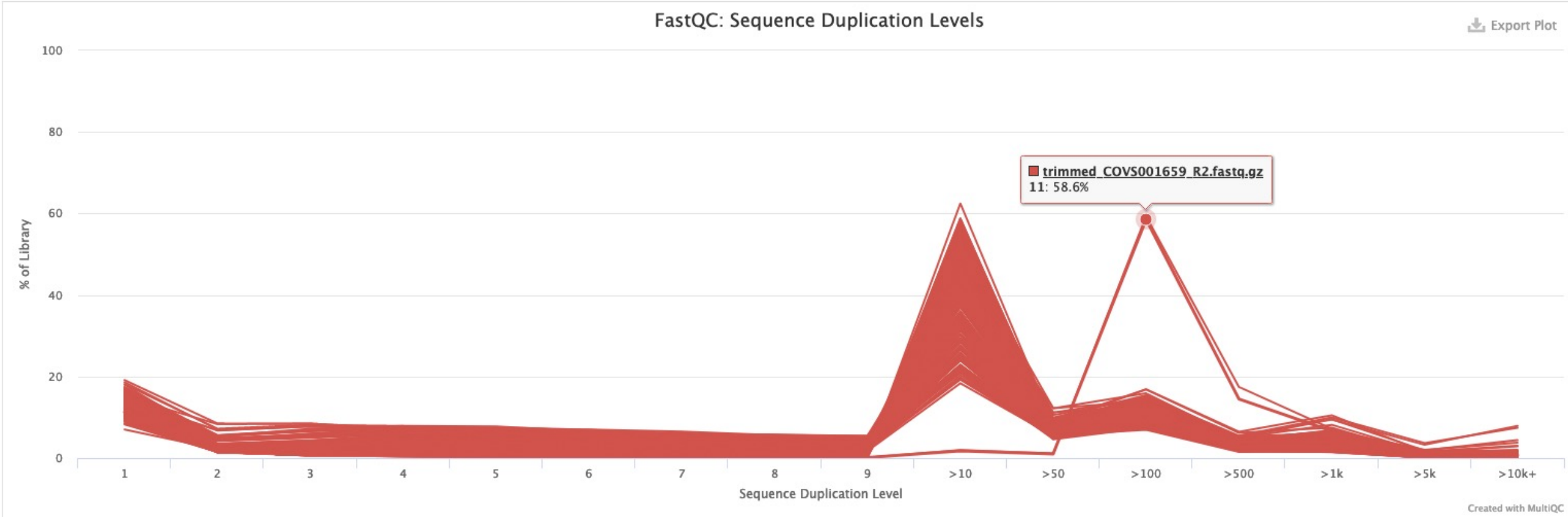
Sequencing quality control: MultiQC

Sequence Duplication Levels 230

Help

The relative level of duplication found for every sequence.

Y-Limits: on

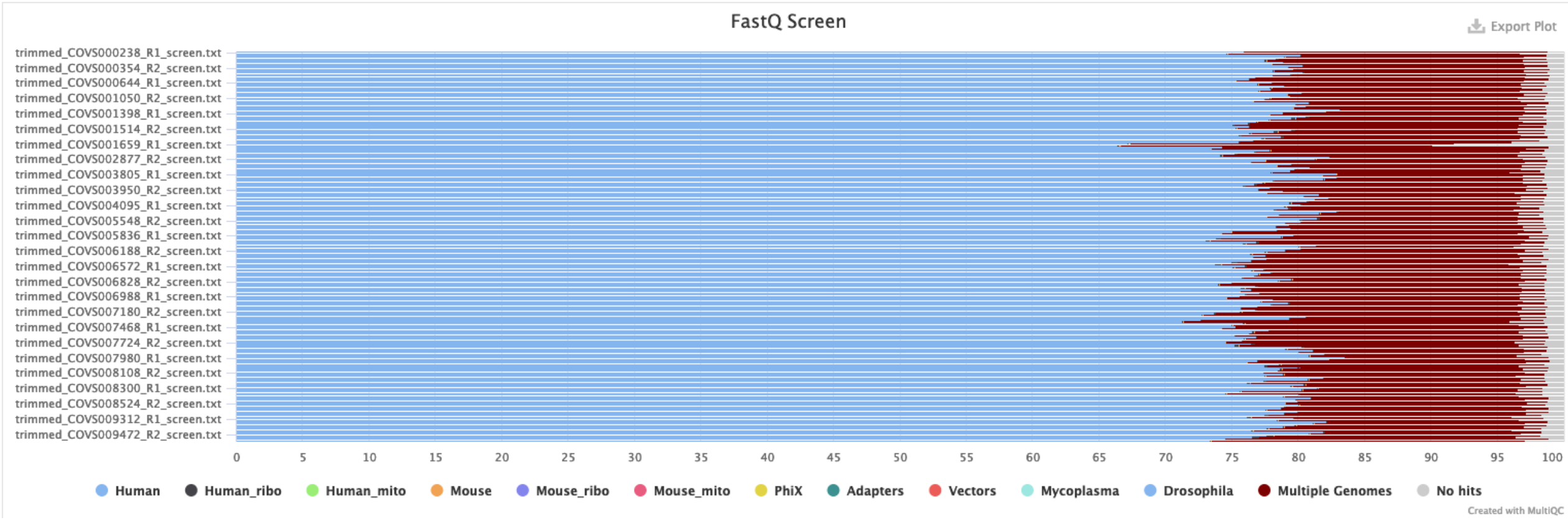


Alignment quality control: MultiQC

FastQ Screen

FastQ Screen allows you to screen a library of sequences in FastQ format against a set of sequence databases so you can see if the composition of the library matches with what you expect.

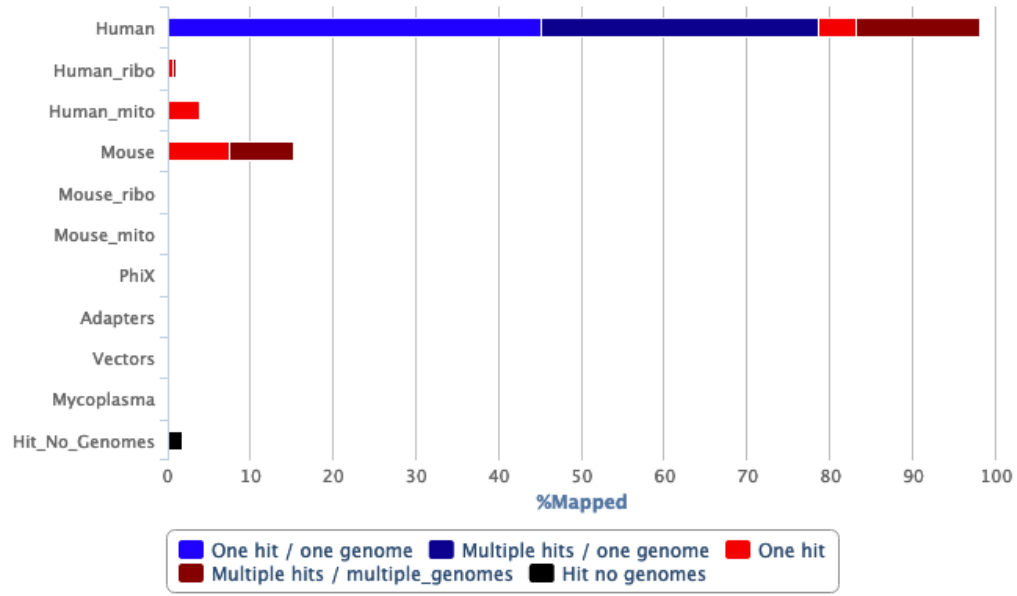
Counts | Percentages



FastQ Screen Processing Report

trimmed_COVS005516_R1.fastq.gz

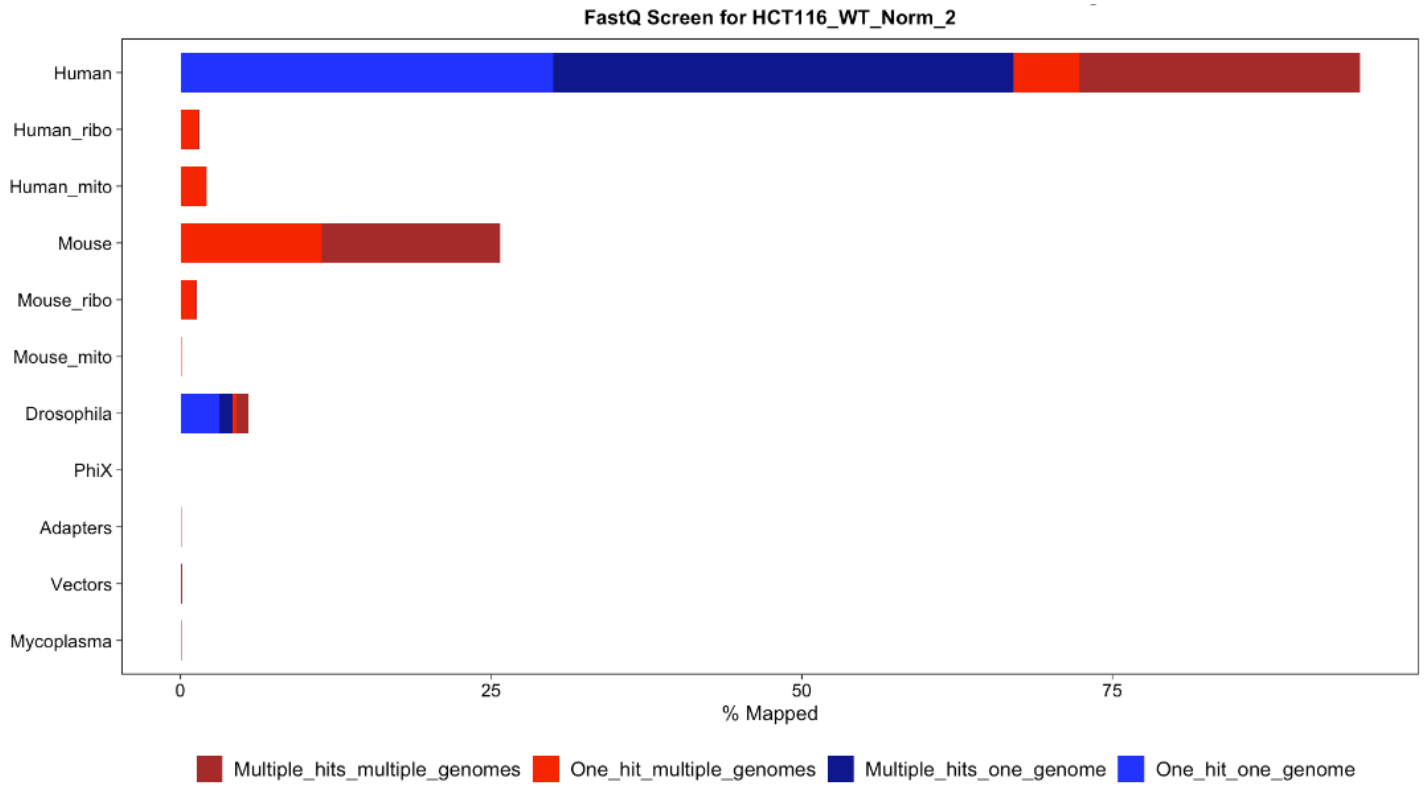
Mapping Results



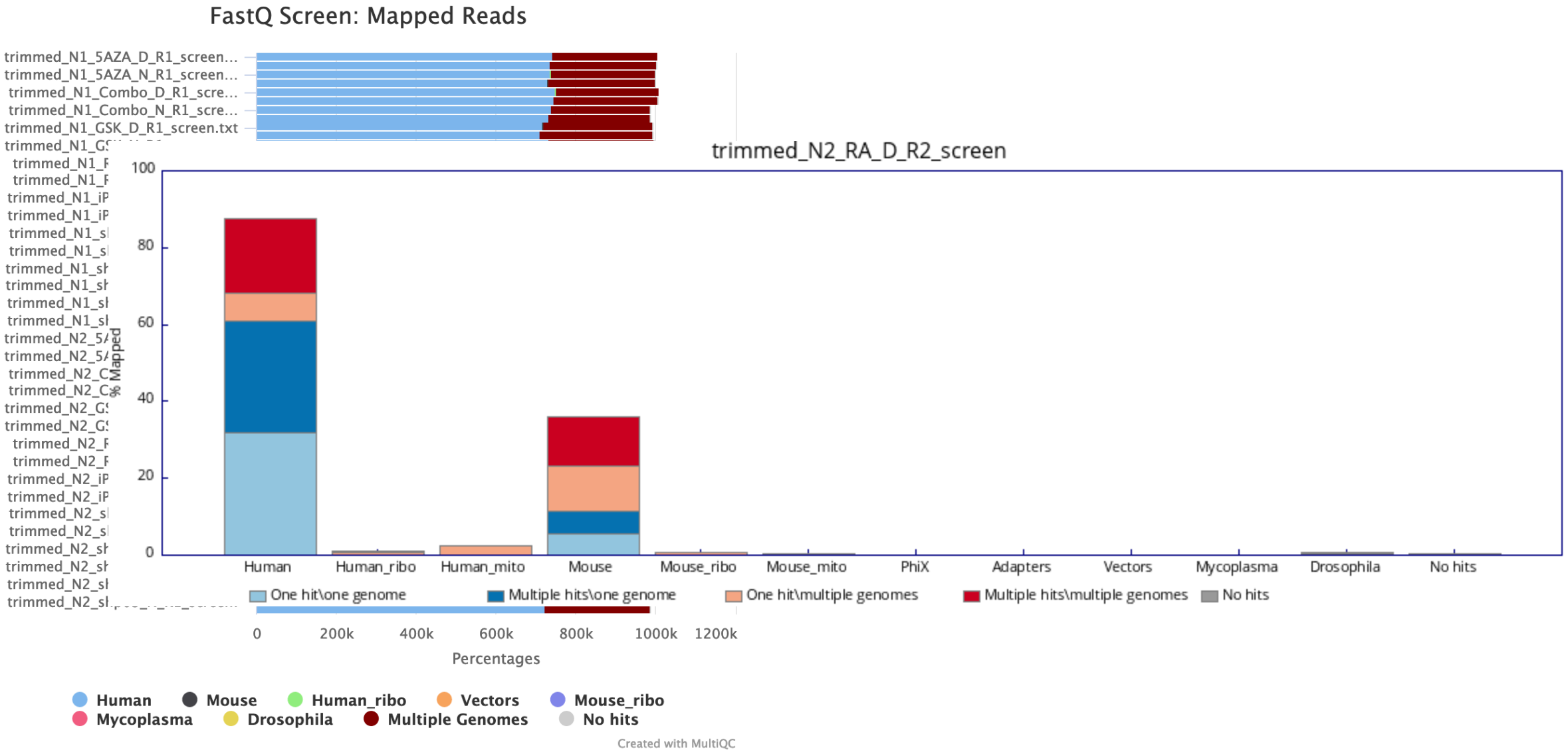
File	Reads processed	Unmapped	One hit / one genome	Multiple hits / one genome	One hit / multiple genomes	Multiple hits / multiple genomes
Human	1,010,168	19,478	455,406	338,776	44,766	151,742
Human_ribo	1,010,168	1,000,616	505	22	6,547	2,478
Human_mito	1,010,168	971,845	0	0	38,199	124
Mouse	1,010,168	856,320	110	71	74,989	78,678
Mouse_ribo	1,010,168	1,006,648	4	0	1,848	1,668
Mouse_mito	1,010,168	1,010,028	0	0	139	1
PhiX	1,010,168	1,010,168	0	0	0	0
Adapters	1,010,168	1,008,969	2	0	433	764
Vectors	1,010,168	1,007,307	8	6	477	2,370
Mycoplasma	1,010,168	1,009,281	1	0	25	861

FASTQ Screen: Drosophila spike-in

With spike-in



FASTQ Screen: mouse contamination?

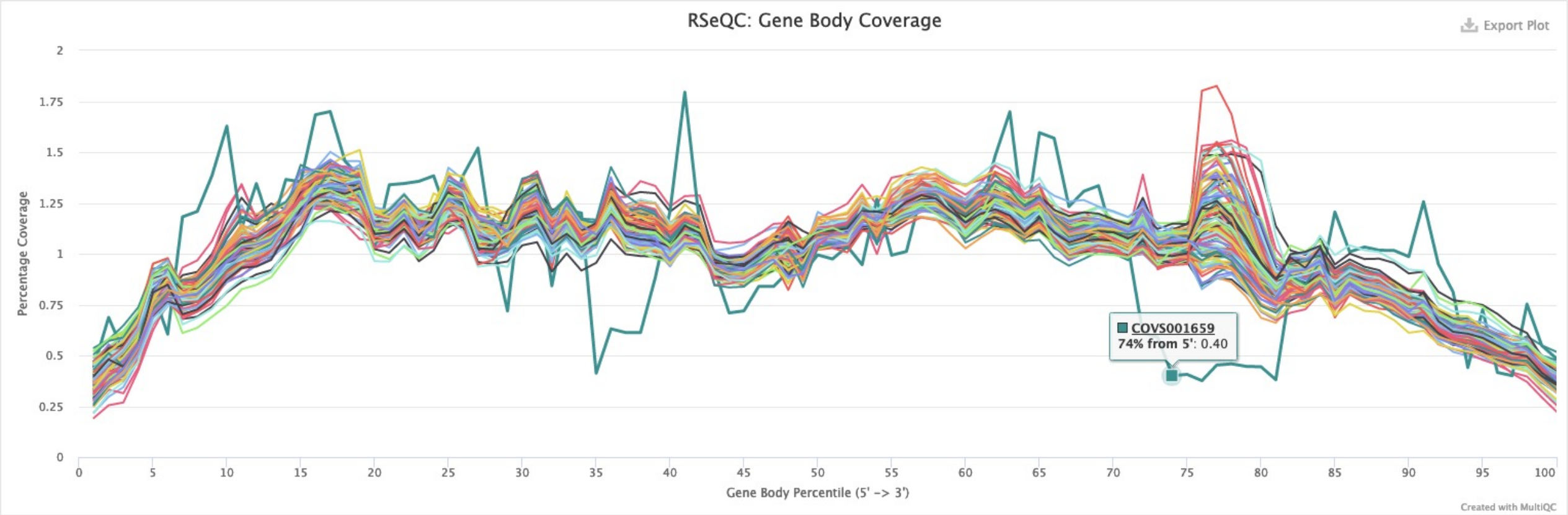


Alignment quality control: MultiQC

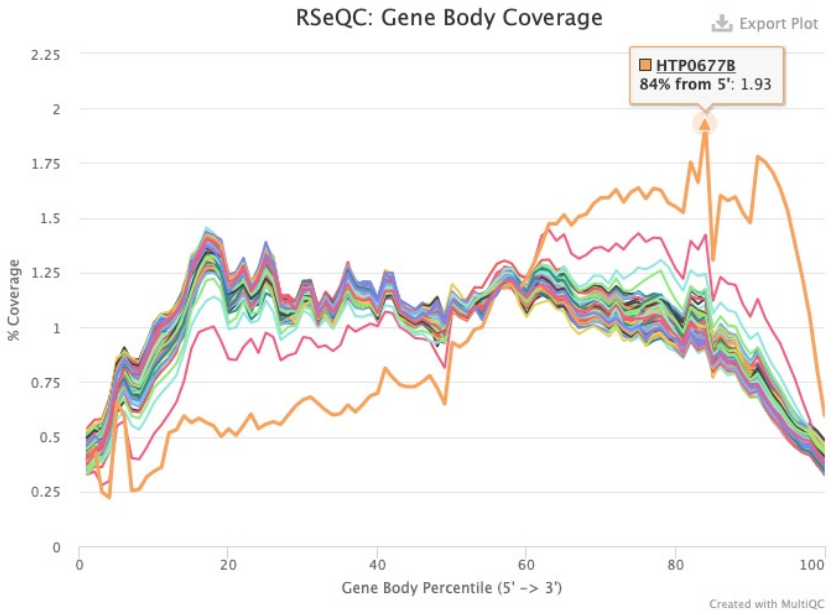
Gene Body Coverage

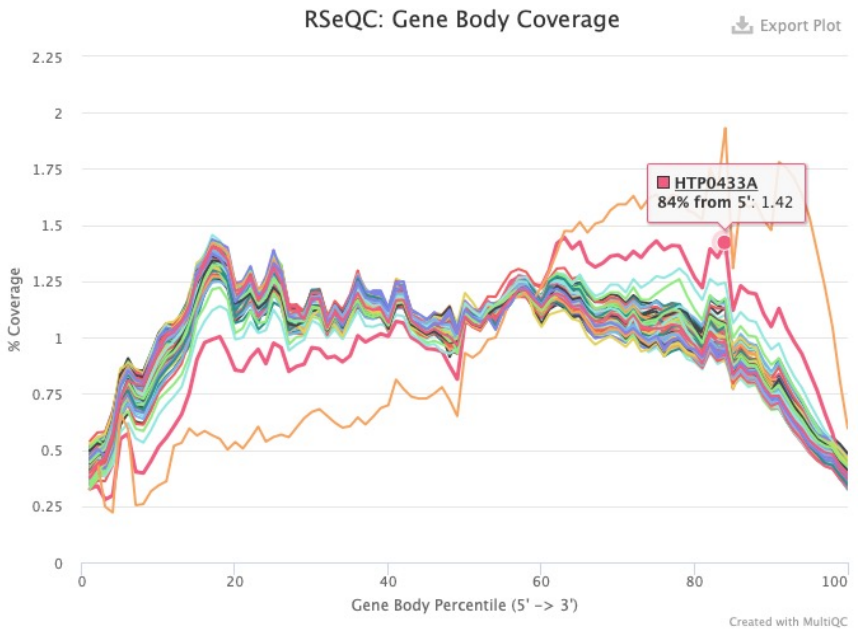
Gene Body Coverage calculates read coverage over gene bodies. This is used to check if reads coverage is uniform and if there is any 5' or 3' bias.

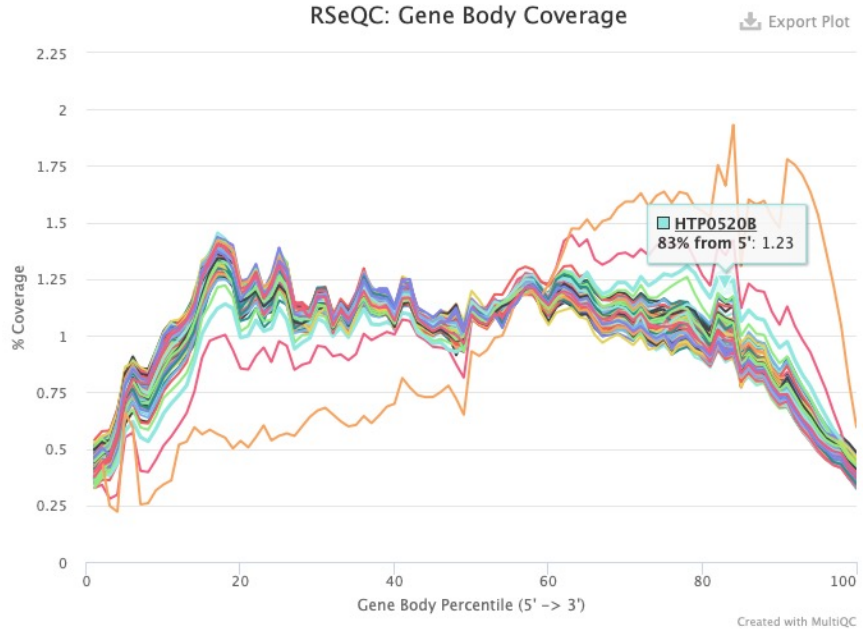
Percentages | Counts



Alignment quality control: MultiQC





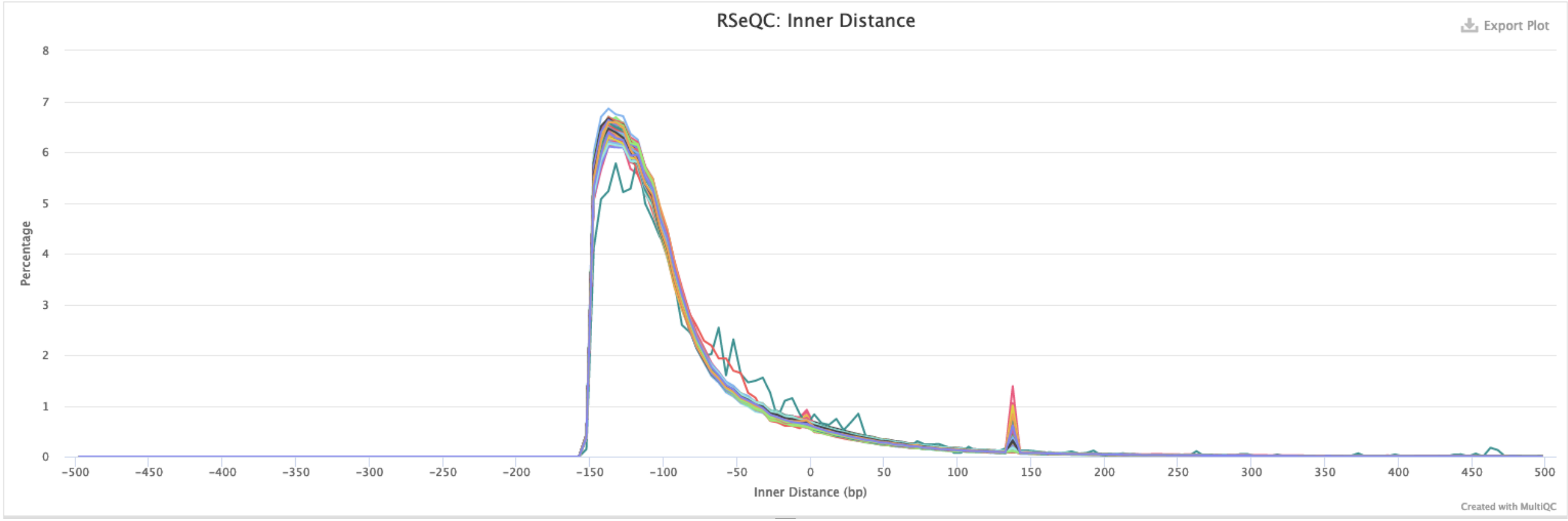


Alignment quality control: MultiQC

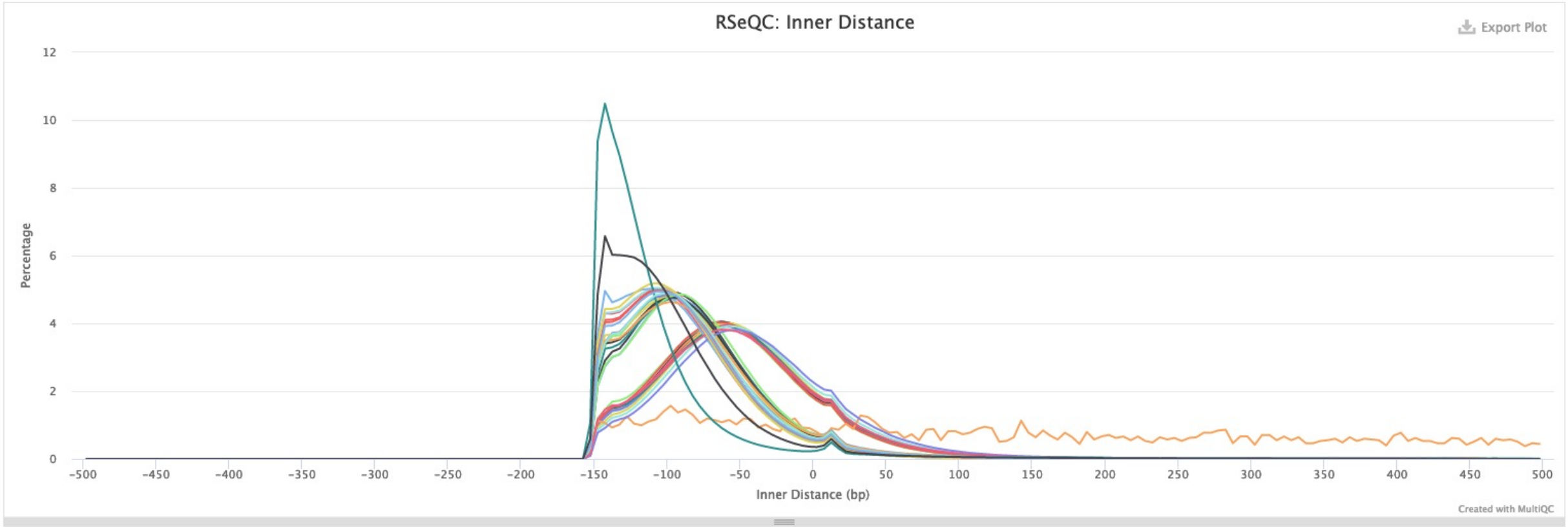
Inner Distance

Inner Distance calculates the inner distance (or insert size) between two paired RNA reads. Note that this can be negative if fragments overlap.

Counts | Percentages



RSeQC Inner Distance: two different libraries?



Alignment quality control: MultiQC

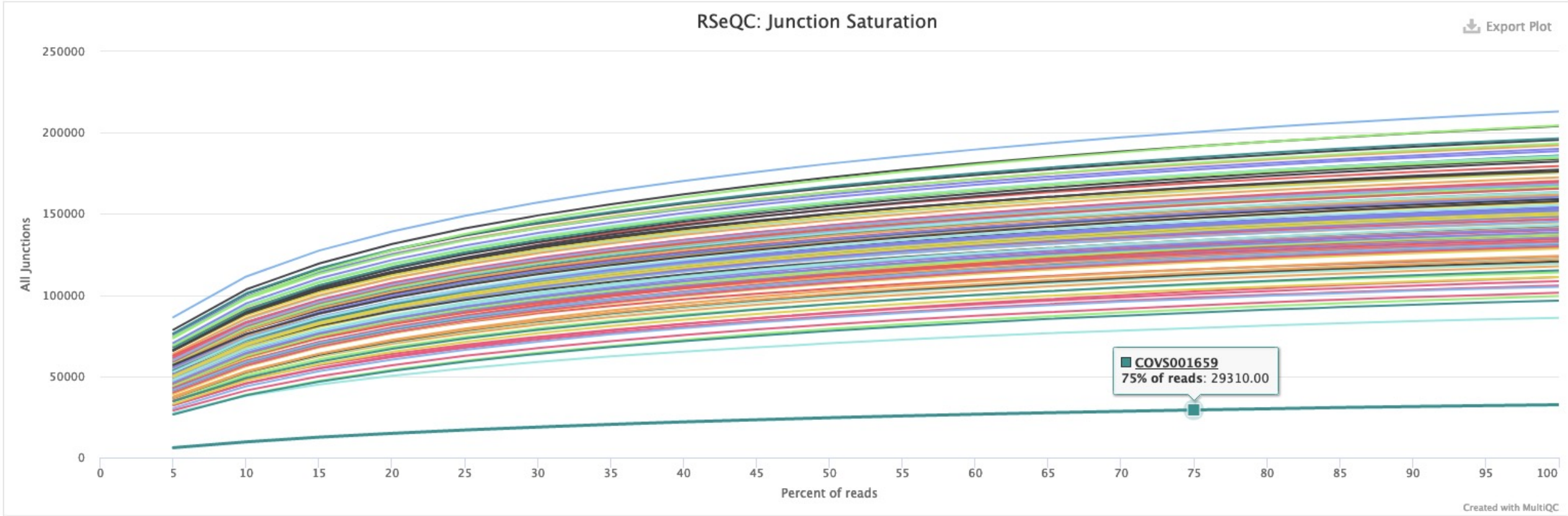
Junction Saturation

Junction Saturation counts the number of known splicing junctions that are observed in each dataset. If sequencing depth is sufficient, all (annotated) splice junctions should be rediscovered, resulting in a curve that reaches a plateau. Missing low abundance splice junctions can affect downstream analysis.

Click a line to see the data side by side (as in the original RSeQC plot).

Y-Limits: on

All Junctions | Known Junctions | Novel Junctions



Getting RNA-seq data into R

What do we need?

- Sample/Experiment **metadata** (should already have this)
- Per sample gene-level → **counts matrix** (R script)
- RPKM/FPKM/TPM data (R script run from command line)
- **R project directory + standardized R script**

Workflow

1. Read in counts data for all samples
2. Filter by minimum cpm
3. Groups and/or covariates setup
4. Generate DESeqDataSet(s)
5. Run DESeq2 analysis
6. Get summary data for all samples
7. Get results for comparisons of interest
8. Plots

```
├── DifferentialExpression_MG.Rproj
├── GSEA
├── GSEA_functions.R
├── NucleicAcidTrans_Fibro_RPE_DESeq2_v2.12.R
├── NucleicAcidTrans_Fibro_RPE_DESeq2_v2.12_FilteredPerCellLine.R
├── data
├── helper_functions_DESeq.R
├── plots
├── rdata
└── results
```

```
#' ### Summary:
#' Testing for differential expression in D21/T21 Fibroblasts and RPEs
#' transfected with various nucleic acids.
#' PolyA+ libraries generated and sequenced by CU Anschutz sequencing core.
#'
#'
#' **Data type(s):**
#'
#' A. RNAseq data
#'
#' **Workflow:**
#'
#' 1. Read in counts data for all samples
#' 2. Filter by minimum cpm
#' 3. Groups and/or cov setup
#' 4. Generate DESeqDataSet(s)
#' 5. Run DESeq2 analysis
#' 6. Get summary data for all samples
#' 7. Get results for comparisons of interest
#' 8. Per-comparison plots
#' 9. Extra plots
#'
#' **Comments:**
#'
#'
```

Getting RNA-seq data into R: packages

```
#+ general_setup, include=FALSE, message=FALSE, warning=FALSE
# Setup ------------------------------------------------------------------
# Clear workspace
if (exists("params")) { rm(list=ls()[! grepl("params", ls())]) } else { rm(list=ls()) } # This excludes "params" from the list for removal
# Load required libraries
library("tools")
library("edgeR") # required for cpm function
library("DESeq2")
library("RColorBrewer")
library("gplots")
library("knitr") # used for simple tables
library("DT") # used for fancy tables
library("kableExtra") # used for horizontal scrolling with very wide kables
library("readxl") # reading Excel files
library("openxlsx") # required for exporting results as Excel workbooks
library("genefilter") # used for function rowVars
library("lattice") # used for PCA plots?
library("graphics") # used for dendrograms
library("dendextend") # used for coloring dendograms
library("ggforce") # used for sina plots
library("reshape2") # melt function
library("tidyverse") # required for ggplot2, dplyr etc
library("ggrepel") # required for using geom_text and geom_text_repel() to make sample labels for PCA plot
library("BiocParallel") # enables mutli-cpu for some of DEseq2 functions
register(MulticoreParam(workers=7)) # enables mutli-cpu for some of DEseq2 functions; can set number of workers - default is all cores;
library("htmltools") # required for outputting multiple datatables from a loop
library("factoextra") # extraction and visualization for PCA
library("conflicted")
conflict_prefer("filter", "dplyr")
conflict_prefer("select", "dplyr")
conflict_prefer("count", "dplyr")
conflict_prefer("paste", "base")
conflict_prefer("rowVars", "matrixStats")
library("here")
```

Getting RNA-seq data into R: input files + variables

```
############################
### EDITABLE PARAMETERS ###
gtf_file <- here("data", "gencode.v33.basic.annotation.gtf.gz")
gene_anno_file <- here("data", "gene_annotation_gencode.v33.basic.txt")
counts_file <- here("data", "NucleicAcidTrans_Fibro_RPE_Gencodev33_counts.txt.gz")
counts_format <- "custom" # featureCounts or HTSeq or custom; custom requires manual editing
rpkms_file <- here("data", "NucleicAcidTrans_Fibro_RPE_Gencodev33_RPKMs.txt.gz")
tpms_file <- here("data", "NucleicAcidTrans_Fibro_RPE_Gencodev33_TPMs.txt.gz")
# htp_meta_data_file <- here("data", "HTP_CLEANED_02_2021_v0.5_MASTER_RecordID_vs_LabID.Labels.tsv") # contains comparison groups info +
expt_meta_data_file <- here("data", "SampleInfo_NucleicAcidTrans_Fibro_RPE.xlsx")
groups_auto <- FALSE # true/false; if true, groups will be automatically extracted from Sampleid which must be in correct format; if fals
min_cpm <- 0.5 # default is 0.5
min_samples <- "auto" # for cpm filter; use number, "all", or "auto" (sets to half number of samples)
predictor <- "Treatment" # not actually used for the comparisons in this version
#
# standard_colors <- c("FALSE" = "#333333", "TRUE" = "#009b4e") # these should be named
# standard_colors <- c("WT" = "grey30", "Dp16" = "#009b4e") # these should be named
#
out_file_prefix <- "NucleicAcidTrans_Fibro_RPE_DESeq2_v2.12_" # should match this script title
### END EDITABLE PARAMETERS ###
############################
source(here("helper_functions_DESeq.R")) # load helper functions
```

Getting RNA-seq data into R: combining counts files

```r
counts_files <- list.files(path = here("data"), pattern = "counts_RPKM_TPM.txt", full.names = TRUE) %>%
  enframe(name = NULL) %>%
  mutate(name = basename(value) %>% str_remove("_HTSeq_counts_RPKM_TPM\\.txt$")) %>%
  select(name, value)
# counts_data <- counts_files[1,2] %>% paste() %>% read_tsv() %>% select(1) # CUSTOM: Get gene list
all_data_long <- NULL # CUSTOM for binding individual samples with Samplid variable
for(i in (seq(1:nrow(counts_files)))) {
  # name <- paste0("counts_", counts_files[i,2]) # changed, KK
  name <- paste0(counts_files[i,1]) # CUSTOM MG
  # count_data <- read_tsv(paste(counts_files[i,2]), col_names=c("EnsemblID", paste(counts_files[i,1])))
  count_data <- read_tsv(paste(counts_files[i,2])) %>% mutate(Sampleid = name)
  assign(name, count_data, envir=.GlobalEnv)
  # counts_data <- counts_data %>% full_join(get(name))
  all_data_long <- bind_rows(all_data_long, count_data) # better way to combine counts + RPKMs + TPMs with Sampleid variable
}
# Clean up column names / custom fixes to Sampleids etc (IF NEEDED)
# NOTHING HERE
#
# NEW / CUSTOM: Split out Counts, RPKMs, TPMs
counts_data <- all_data_long %>%
  select(Geneid, Sampleid, raw_count) %>%
  pivot_wider(names_from = Sampleid, values_from = raw_count)
rpkms_data <- all_data_long %>%
  select(Geneid, Sampleid, RPKM) %>%
  pivot_wider(names_from = Sampleid, values_from = RPKM)
tpms_data <- all_data_long %>%
  select(Geneid, Sampleid, TPM) %>%
  pivot_wider(names_from = Sampleid, values_from = TPM)
#
counts_data %>% write_tsv(file = counts_file)
rpkms_data %>% write_tsv(file = rpkms_file)
tpms_data %>% write_tsv(file = tpms_file)
```

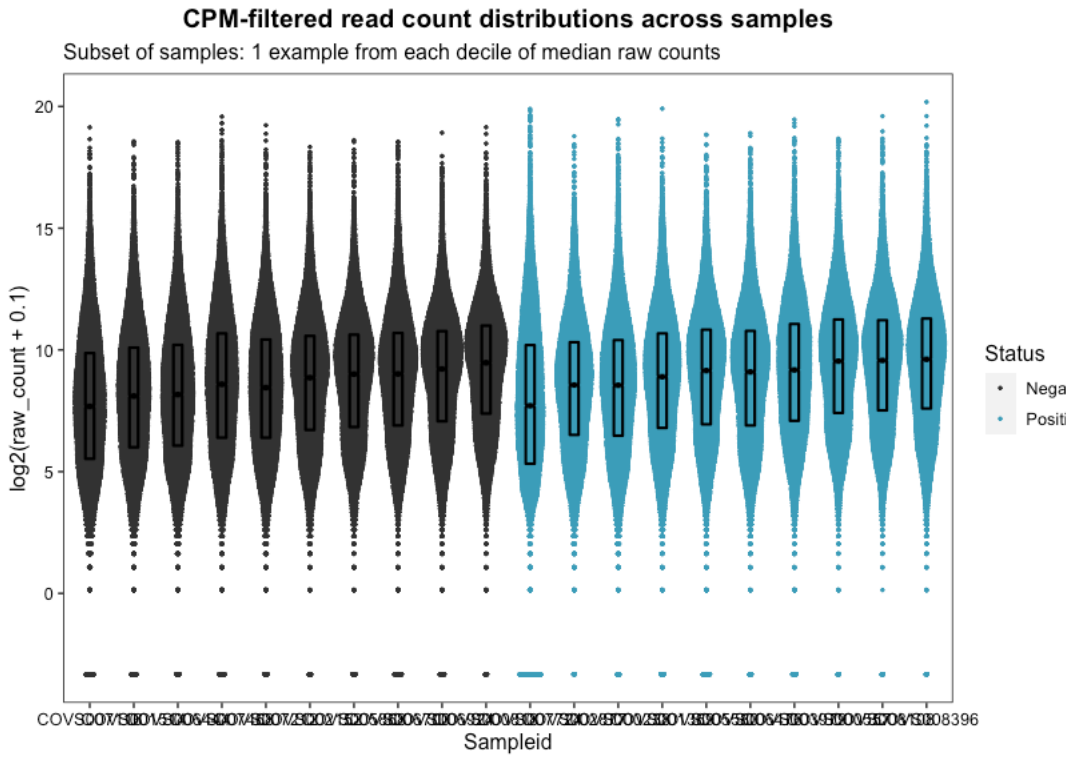
Getting RNA-seq data into R: combined counts

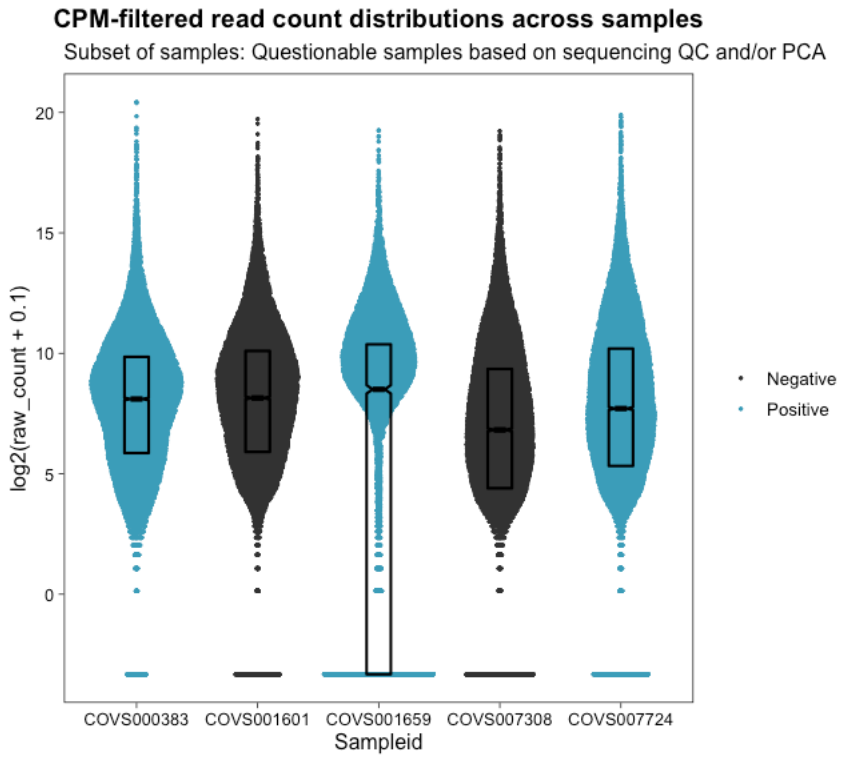
```
# A tibble: 60,662 × 41
   Geneid           `2036_D21_Ctrl…` `2036_D21_Ctrl…` `2036_D21_ssRN…` `2036_D21_ssRN…` `2036_D21_pIC_1` `2036_D21_pIC_2`
   <chr>                       <dbl>            <dbl>            <dbl>            <dbl>            <dbl>            <dbl>
 1 ENSG00000000003…              335              434              185              220              234              399
 2 ENSG00000000005…                0                0                0                0                0                0
 3 ENSG00000000419…              955             1205              881             1178              966             1522
 4 ENSG00000000457…              171              242              223              313              180              452
 5 ENSG00000000460…              561              577              607              499              674              773
 6 ENSG00000000938…                0                0                9                6                3                2
 7 ENSG00000000971…              593              839             2517             3820             2575             6480
 8 ENSG00000001036…             5165             5758             5295             6931             5451            10235
 9 ENSG00000001084…              461              449              569              548              568              750
10 ENSG00000001167…             1001             1190              674              952              686             1226
# … with 60,652 more rows, and 34 more variables: `2036_D21_dsDNA_1` <dbl>, `2036_D21_dsDNA_2` <dbl>,
#   `2036_D21_ssDNA_1` <dbl>, `2036_D21_ssDNA_2` <dbl>, `2767_T21_Ctrl_1` <dbl>, `2767_T21_Ctrl_2` <dbl>,
#   `2767_T21_ssRNA_1` <dbl>, `2767_T21_ssRNA_2` <dbl>, `2767_T21_pIC_1` <dbl>, `2767_T21_pIC_2` <dbl>,
#   `2767_T21_dsDNA_1` <dbl>, `2767_T21_dsDNA_2` <dbl>, `2767_T21_ssDNA_1` <dbl>, `2767_T21_ssDNA_2` <dbl>,
#   RPE_D21_Ctrl_1 <dbl>, RPE_D21_Ctrl_2 <dbl>, RPE_D21_ssRNA_1 <dbl>, RPE_D21_ssRNA_2 <dbl>, RPE_D21_pIC_1 <dbl>,
#   RPE_D21_pIC_2 <dbl>, RPE_D21_dsDNA_1 <dbl>, RPE_D21_dsDNA_2 <dbl>, RPE_D21_ssDNA_1 <dbl>, RPE_D21_ssDNA_2 <dbl>,
#   RPE_T21_Ctrl_1 <dbl>, RPE_T21_Ctrl_2 <dbl>, RPE_T21_ssRNA_1 <dbl>, RPE_T21_ssRNA_2 <dbl>, RPE_T21_pIC_1 <dbl>, …
```

Getting RNA-seq data into R: combined RPKMs

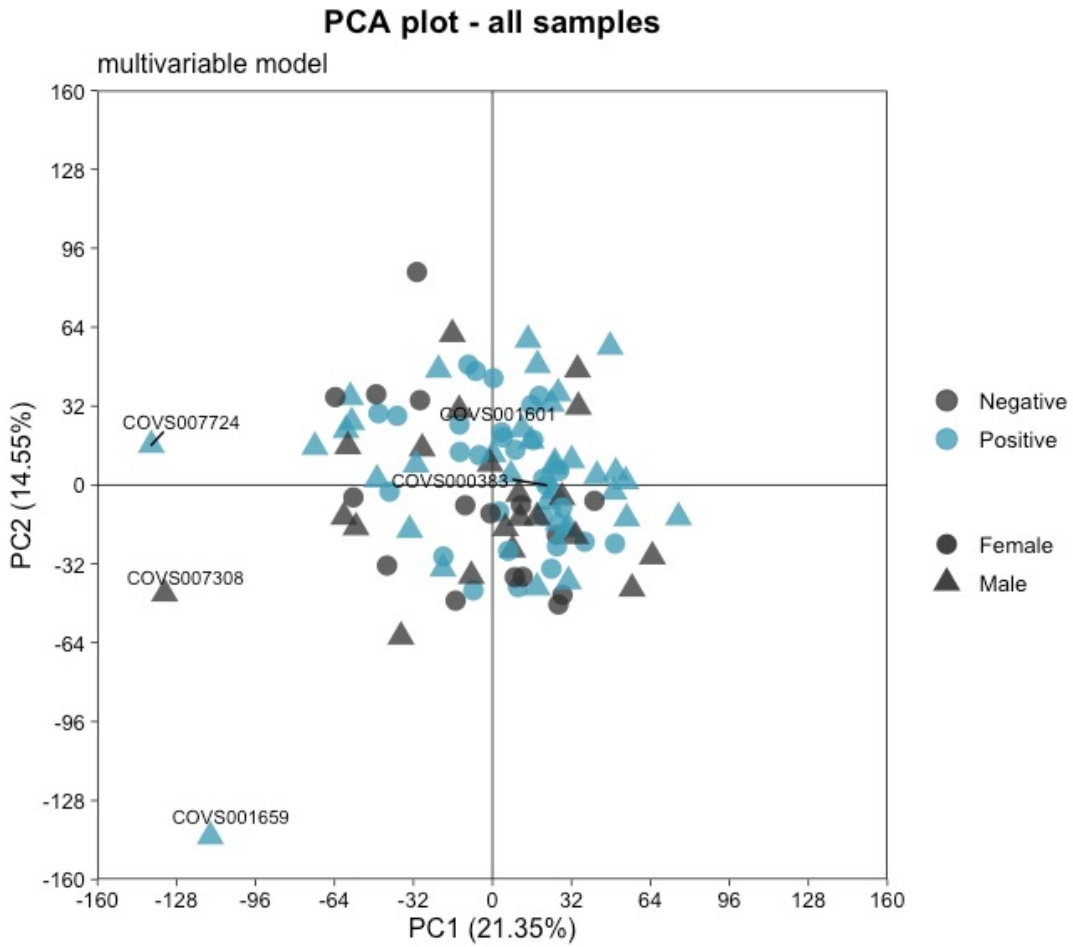
```
> rpkms_data
# A tibble: 60,662 × 41
   Geneid          `2036_D21_Ctrl…` `2036_D21_Ctrl…` `2036_D21_ssRN…` `2036_D21_ssRN…` `2036_D21_pIC_1` `2036_D21_pIC_2`
   <chr>                      <dbl>            <dbl>            <dbl>            <dbl>            <dbl>            <dbl>
 1 ENSG00000000003…           3.09             3.84             1.88             1.81             2.28             2.18
 2 ENSG00000000005…           0                0                0                0                0                0
 3 ENSG00000000419…          30.9             37.4             31.5             34.0             33.0             29.2
 4 ENSG00000000457…           0.983            1.34             1.42             1.61             1.09             1.54
 5 ENSG00000000460…           4.34             4.29             5.19             3.45             5.51             3.55
 6 ENSG00000000938…           0                0                0.126            0.0679           0.0401           0.0150
 7 ENSG00000000971…           5.29             7.19            24.8             30.4             24.2             34.3
 8 ENSG00000001036…          81.4             87.1             92.0             97.5             90.6             95.5
 9 ENSG00000001084…           2.73             2.55             3.72             2.90             3.54             2.63
10 ENSG00000001167…           9.87            11.3              7.33             8.38             7.13             7.16
# … with 60,652 more rows, and 34 more variables: `2036_D21_dsDNA_1` <dbl>, `2036_D21_dsDNA_2` <dbl>,
#   `2036_D21_ssDNA_1` <dbl>, `2036_D21_ssDNA_2` <dbl>, `2767_T21_Ctrl_1` <dbl>, `2767_T21_Ctrl_2` <dbl>,
#   `2767_T21_ssRNA_1` <dbl>, `2767_T21_ssRNA_2` <dbl>, `2767_T21_pIC_1` <dbl>, `2767_T21_pIC_2` <dbl>,
#   `2767_T21_dsDNA_1` <dbl>, `2767_T21_dsDNA_2` <dbl>, `2767_T21_ssDNA_1` <dbl>, `2767_T21_ssDNA_2` <dbl>,
#   RPE_D21_Ctrl_1 <dbl>, RPE_D21_Ctrl_2 <dbl>, RPE_D21_ssRNA_1 <dbl>, RPE_D21_ssRNA_2 <dbl>, RPE_D21_pIC_1 <dbl>,
#   RPE_D21_pIC_2 <dbl>, RPE_D21_dsDNA_1 <dbl>, RPE_D21_dsDNA_2 <dbl>, RPE_D21_ssDNA_1 <dbl>, RPE_D21_ssDNA_2 <dbl>,
#   RPE_T21_Ctrl_1 <dbl>, RPE_T21_Ctrl_2 <dbl>, RPE_T21_ssRNA_1 <dbl>, RPE_T21_ssRNA_2 <dbl>, RPE_T21_pIC_1 <dbl>, …
```

Quantitative quality control in R





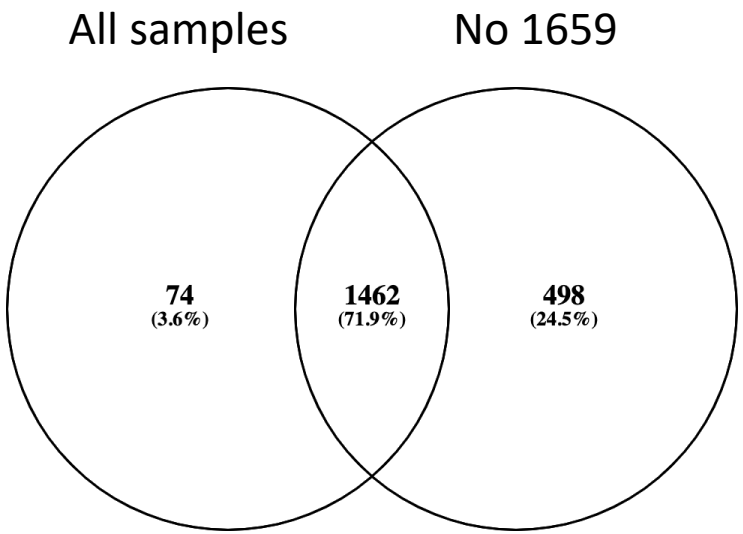
Quantitative quality control in R



2 other possible outliers:
COVS007724
COVS007308

Quantitative quality control in R

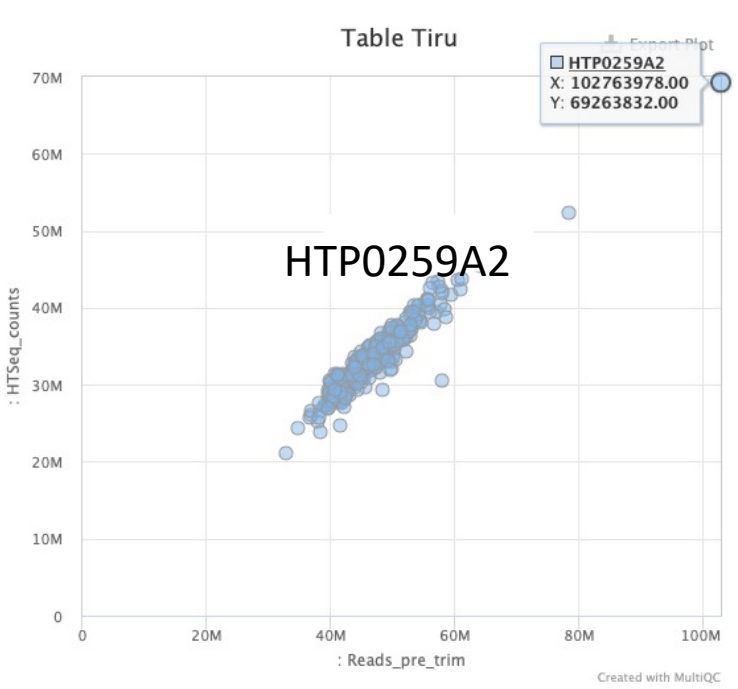
```
## ------------
## Results summary for contrast: Status_Positive_vs_Negative
## ------------
## independentFiltering ON, Cooks cutoff ON
## out of 15395 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 1028, 6.7%
## LFC < 0 (down)     : 508, 3.3%
## outliers [1]       : 17, 0.11%
## low counts [2]     : 299, 1.9%
## (mean count < 25)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

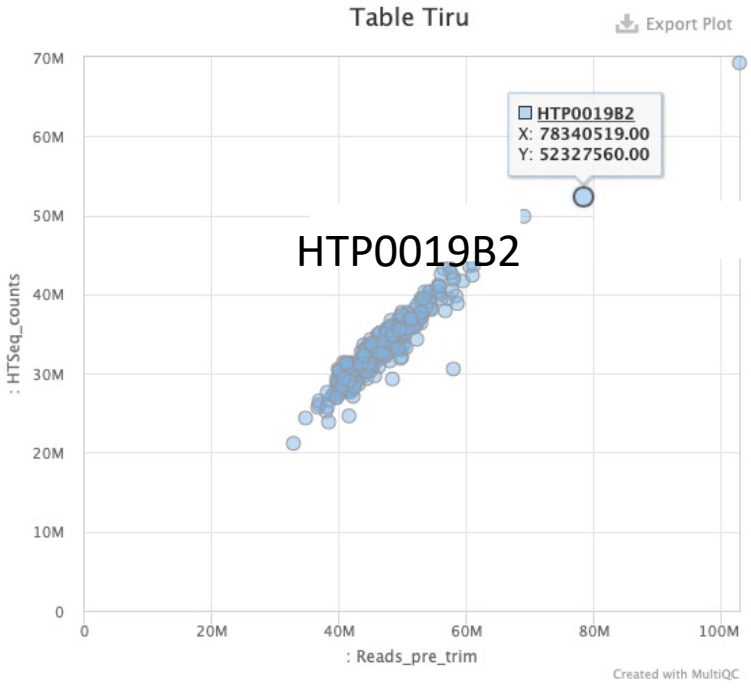


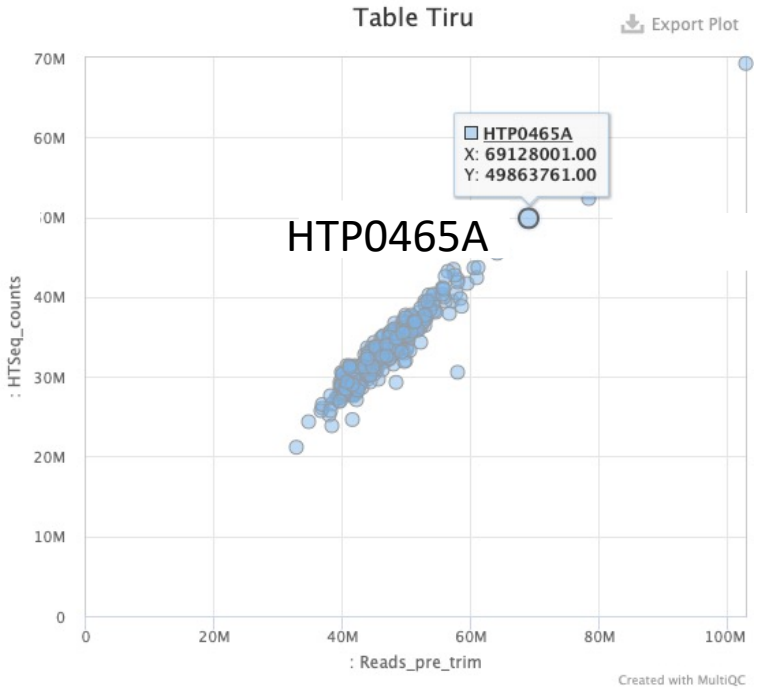

```
## ------------
## Results summary for contrast: Status_Positive_vs_Negative
## ------------
## independentFiltering ON, Cooks cutoff ON
## out of 15387 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 1094, 7.1%
## LFC < 0 (down)     : 866, 5.6%
## outliers [1]       : 17, 0.11%
## low counts [2]     : 299, 1.9%
## (mean count < 25)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

Quantitative quality control in R

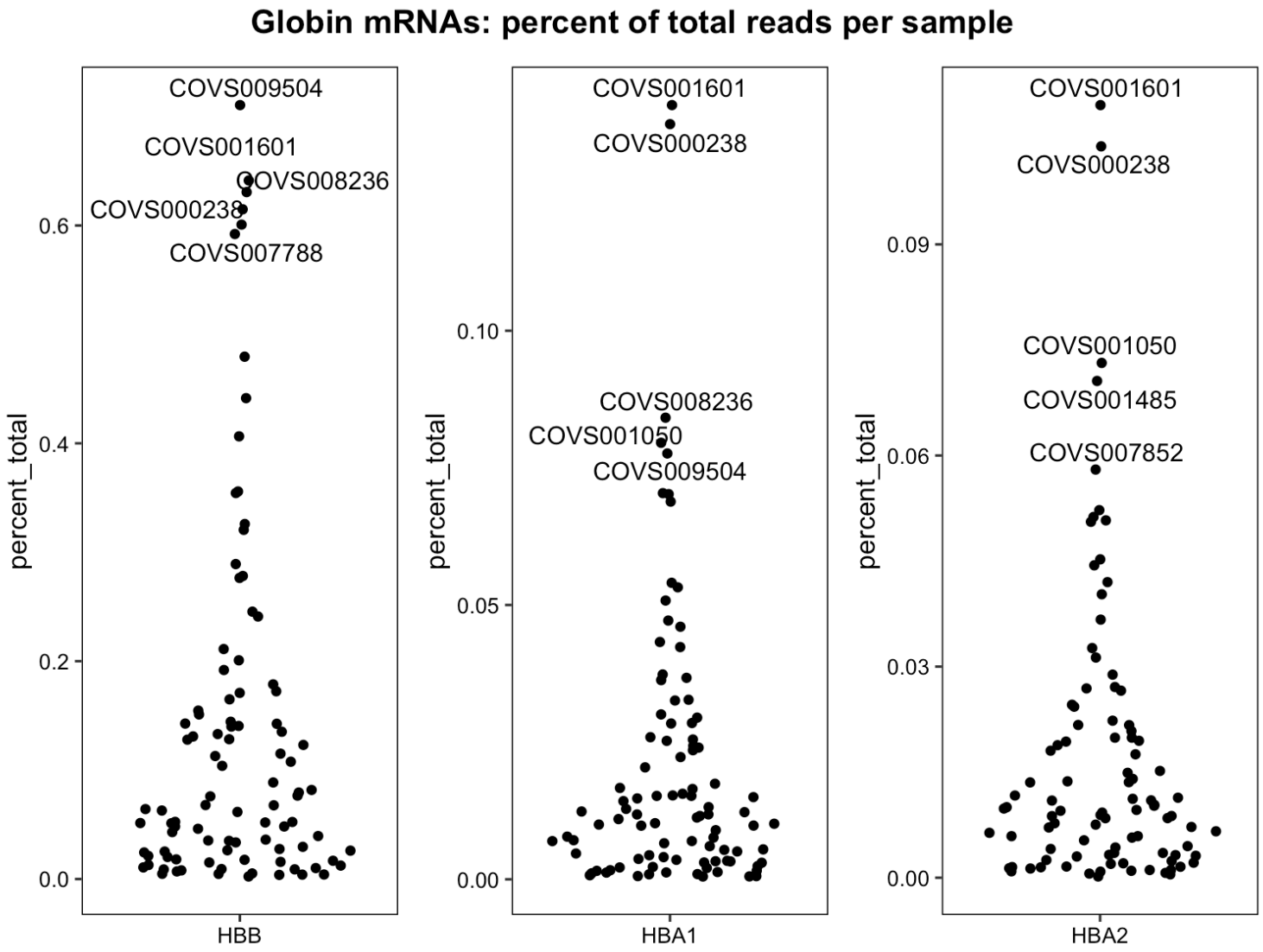
High-count outlier samples → down sample?



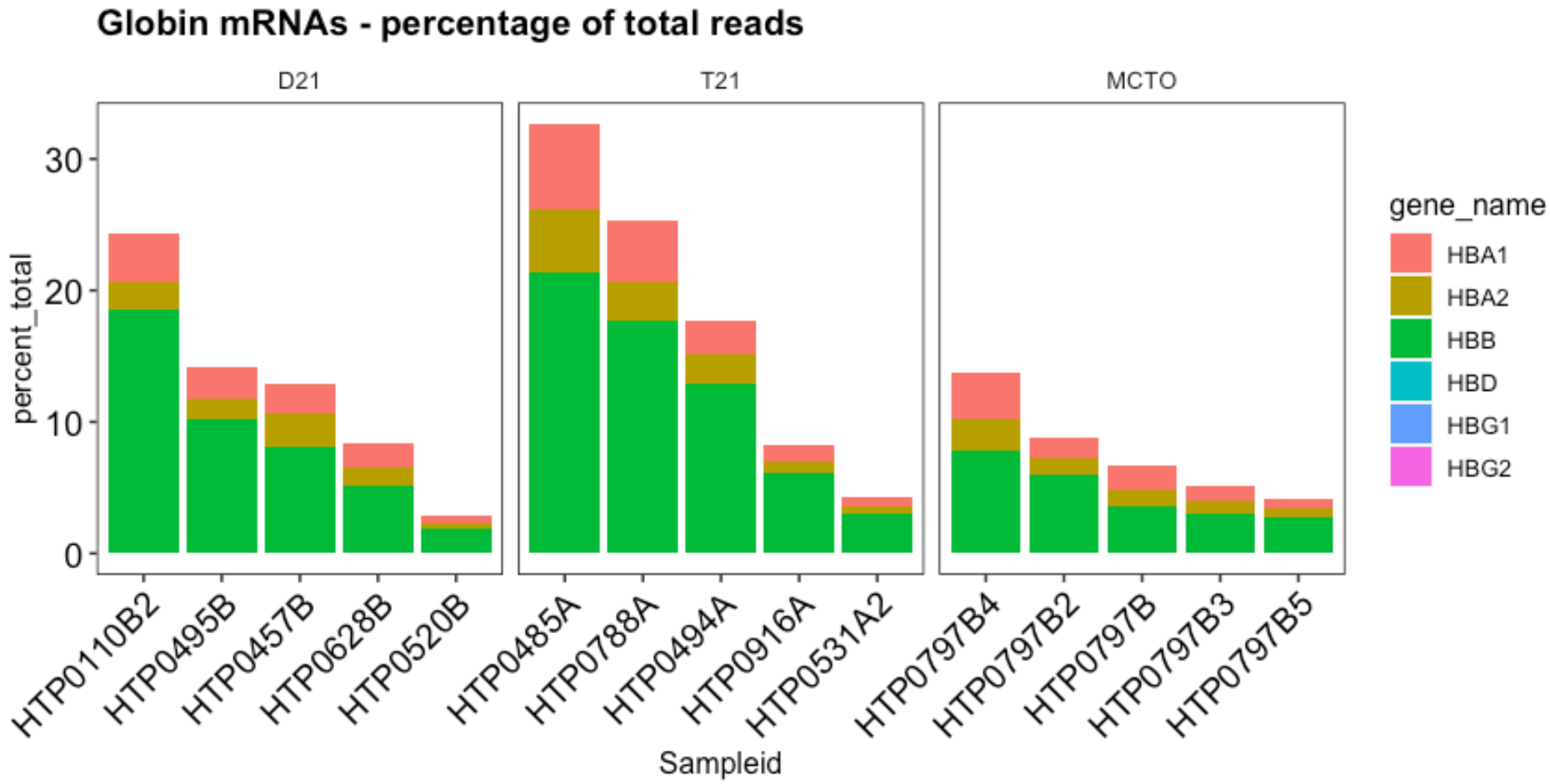




Quantitative quality control: globin depletion for whole blood RNA



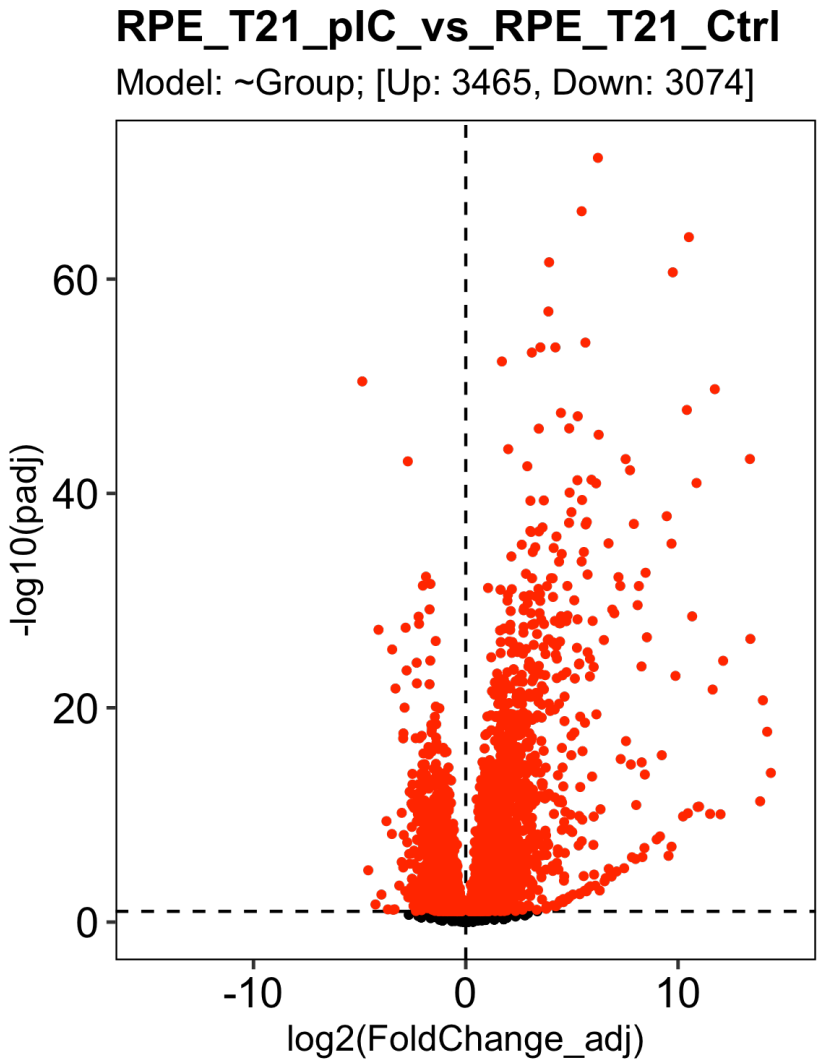
Quantitative quality control: globin depletion for whole blood RNA



= potential problem with globin depletion

RNA-seq differential expression analysis: DESeq2

```
------------
Results summary for contrast: RPE_T21_pIC vs. RPE_T21_Ctrl
------------
independentFiltering ON, Cooks cutoff ON
out of 13945 with nonzero total read count
adjusted p-value < 0.1
LFC > 0 (up)       : 3465, 25%
LFC < 0 (down)     : 3074, 22%
outliers [1]       : 0, 0%
low counts [2]     : 0, 0%
(mean count < 12)
[1] see 'cooksCutoff' argument of ?results
[2] see 'independentFiltering' argument of ?results
```



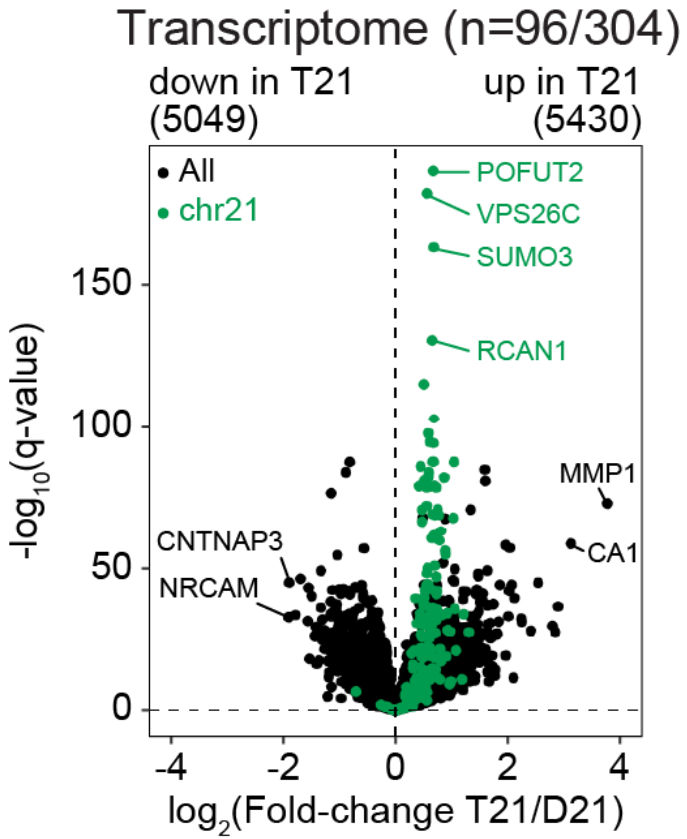
RNA-seq differential expression analysis: DESeq2

Assemble results and export (.txt + .xlsx)

```
> res_batch_RPE_T21_pIC_vs_RPE_T21_Ctrl
# A tibble: 13,945 x 16
   Gene_name chr   Geneid             baseMean baseMean_RPE_T2… baseMean_RPE_T2… FoldChange log2FoldChange FoldChange_adj
   <chr>     <chr> <chr>                 <dbl>            <dbl>            <dbl>      <dbl>          <dbl>          <dbl>
 1 PLEKHA4   chr19 ENSG00000105559.12    6061.             160.            11990.      74.8            6.22           72.5
 2 C1R       chr12 ENSG00000159403.18    6572.             188.             8241.      43.9            5.46           42.8
 3 BTC       chr4  ENSG00000174808.12     182.              55.8             845.      15.2            3.92           15.0
 4 SLC41A2   chr12 ENSG00000136052.9      459.             176.              1521.       8.66           3.11            8.54
 5 TENT5A    chr6  ENSG00000112773.16    2278.             256.              3805.      14.8            3.89           14.5
 6 LGMN      chr14 ENSG00000100600.15    3636.             232.              8953.      38.7            5.27           37.5
 7 CTSS      chr1  ENSG00000163131.11    1942.              35.8             5102.     143.             7.16          137.
 8 VEGFC     chr4  ENSG00000150630.4     8801.            1987.             17147.       8.63           3.11            8.49
 9 PARP10    chr8  ENSG00000178685.14    4327.             141.              7044.      50.0            5.64           48.2
10 ERAP2     chr5  ENSG00000164308.17    2861.              80.5             1506.      18.7            4.22           18.2
# … with 13,935 more rows, and 7 more variables: log2FoldChange_adj <dbl>, pvalue <dbl>, padj <dbl>, start <dbl>,
#   end <dbl>, gene_type <chr>, hgnc_id <chr>
```

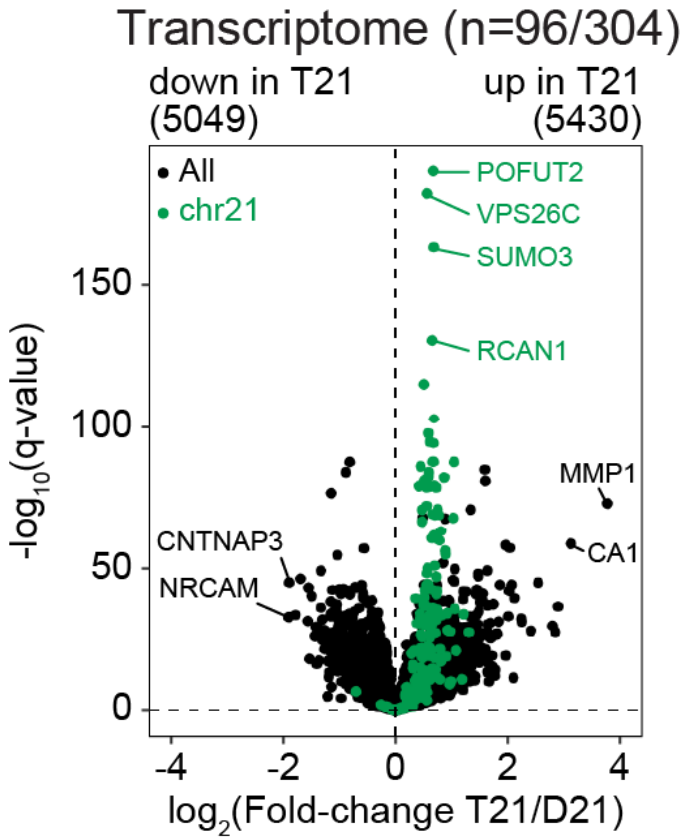
Note for excel users: watch out for conversion of gene names to dates!!

HTP RNA-seq: How to make the plots

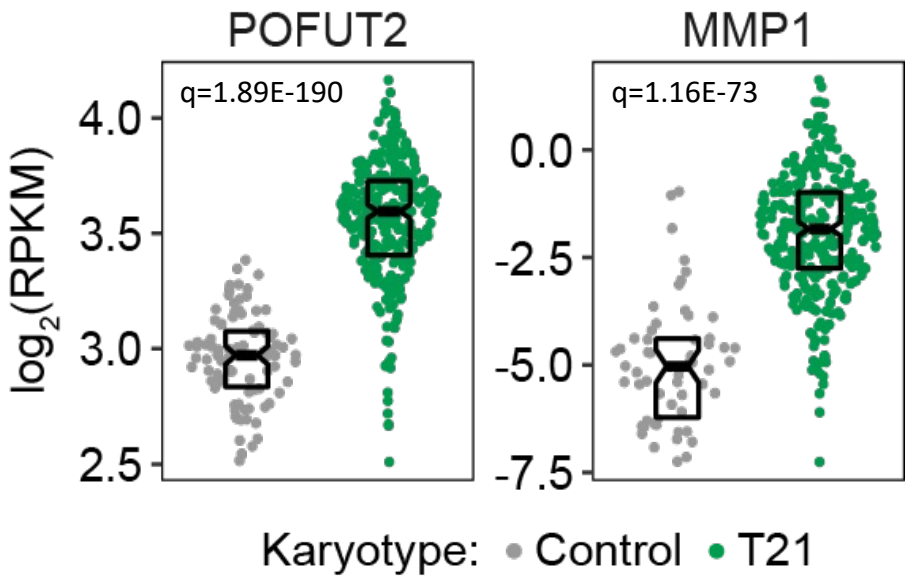



```
res_multivariable_Karyotype_T21_vs_Control %>%
  volcano_plot_chr21(
    title = "Differential expression: T21 vs. D21",
    subtitle = paste0("Model: ", c(multivar_formula), "\n[Down: ",(.) %>% filter(padj < 0.1 & FoldChange_adj <1) %>% nrow(), "; Up: ",(.) %>%
  )
ggsave(filename = here("plots", paste0(out_file_prefix, "chr21_volcano_multi.png")), width = 5, height = 5, units = "in")
ggsave(filename = here("plots", paste0(out_file_prefix, "chr21_volcano_multi.pdf")), device = cairo_pdf, width = 5, height = 5, units = "in")
```

HTP RNA-seq: How to make the plots




```r
# Highlight chr21 genes Volcano plot function
volcano_plot_chr21 <- function(res, title = "",
                               subtitle = "down in T21                                                                up in T21",
                               y_lim = c(0, NA)){
  res <- res %>%
    mutate(
      color = if_else(chr == "chr21", "chr21", "All")
    ) %>%
    arrange(color)
  # get max for x-axis
  x_lim <- res %>%
    summarize(max = max(log2(FoldChange_adj), na.rm = TRUE), min = min(log2(FoldChange_adj), na.rm = TRUE)) %>%
    abs() %>%
    max() %>%
    ceiling()
  res %>%
    ggplot(aes(log2(FoldChange_adj), -log10(padj), color = color)) +
    geom_hline(yintercept = -log10(0.1), linetype = 2) +
    geom_vline(xintercept = 0, linetype = 2) +
    geom_point(data = . %>% filter(chr != "chr21")) +
    geom_point(data = . %>% filter(chr == "chr21")) +
    scale_color_manual(values = c("chr21" = "#009b4e", "All" = "black")) +
    xlim(-x_lim, x_lim) +
    ylim(y_lim) +
    geom_text_repel(data = res %>% filter(!is.na(padj)) %>% slice_max(order_by = FoldChange_adj, n = 4), aes(label =
Gene_name), min.segment.length = 0, show.legend = FALSE, nudge_x = 0.5, nudge_y = 10) +
    geom_text_repel(data = res %>% filter(!is.na(padj)) %>% slice_min(order_by = FoldChange_adj, n = 4), aes(label =
Gene_name), min.segment.length = 0, show.legend = FALSE, nudge_x = -0.5, nudge_y = 10) +
    geom_text_repel(data = res %>% filter(!is.na(padj)) %>% slice_min(order_by = padj, n = 4), aes(label = Gene_name
), min.segment.length = 0, show.legend = FALSE, nudge_x = 0.5, nudge_y = -10) +
    theme(aspect.ratio=1.2) +
    labs(
      title = title,
      subtitle = subtitle
    )
} # end of function
```




```r
# example sina plots (RPKM adjusted) for Fig 1 ----
rpkm_adj %>%
  filter(Gene_name %in% c("POFUT2", "MMP1", "RCAN1", "CA1", "VPS26C")) %>%
  pivot_longer(cols = matches("HTP"), names_to = "Sampleid", values_to = "RPKM_adj") %>%
  inner_join(meta_data) %>%
  mutate(gene_name = fct_relevel(Gene_name, c("POFUT2", "MMP1", "RCAN1", "CA1", "VPS26C"))) %>%
  group_by(EnsemblID, Karyotype) %>%
  mutate(extreme = rstatix::is_extreme(log2(RPKM_adj))) %>%
  ungroup() %>%
  filter(extreme == FALSE) %>%
  ggplot(aes(Karyotype, log2(RPKM_adj), color = Karyotype)) +
  geom_sina(size = 0.75) +
  geom_boxplot(notch = TRUE, varwidth = FALSE, outlier.shape = NA, coef = FALSE, width = 0.3, color = "black", fill = "transparent", size = 0.75) +
  facet_wrap(~ gene_name, scales = "free_y", nrow = 1) +
  scale_color_manual(values = standard_colors) +
  labs(
    title = "RPKM (SexAgeSource adjusted, no extreme): selected",
    x = NULL
  ) +
  theme(
    aspect.ratio = 1.3,
    axis.text.x = element_blank(),
    legend.position = "bottom"
  )
ggsave(filename = here("plots", paste0(out_file_prefix, "sina_fig1_T21vsD21_RPKM_adj.png")), width = 15, height = 3, units = "in")
ggsave(filename = here("plots", paste0(out_file_prefix, "sina_fig1_T21vsD21_RPKM_adj.pdf")), device = cairo_pdf, width = 15, height = 3, units = "in")
```