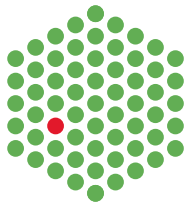
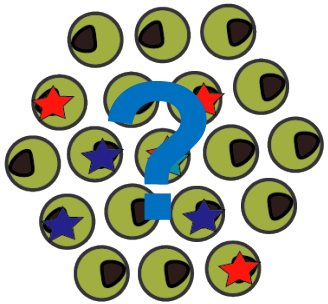
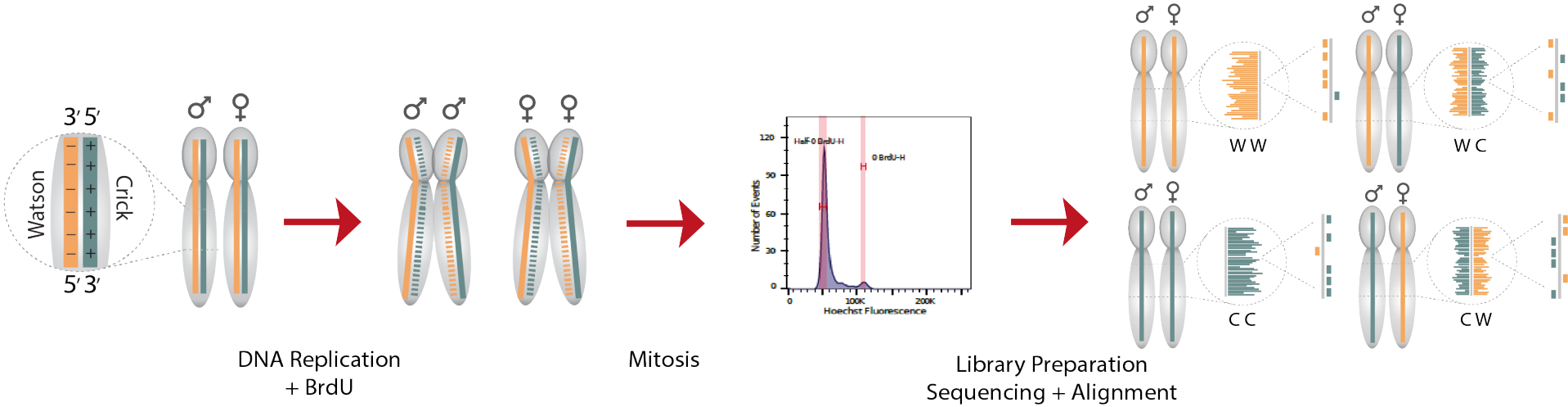
Detection of structural variations at the single-cell level using Strand-seq





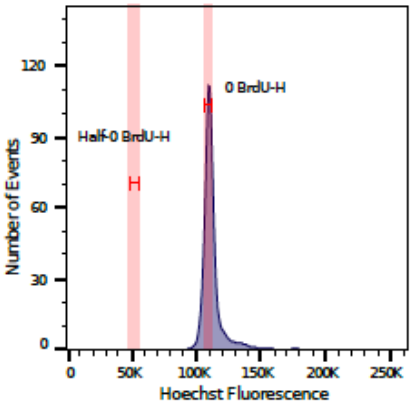
1. Structural variant detection
2. Single-cell resolution

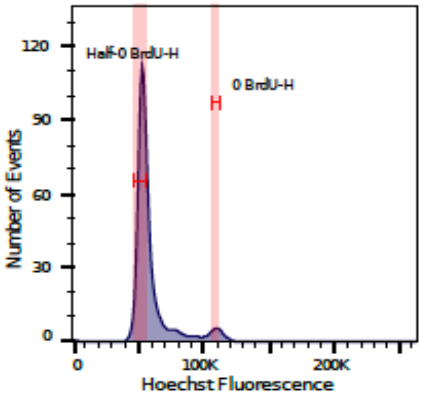
Strand-seq selectively sequences DNA template from single cells to preserve DNA directionality



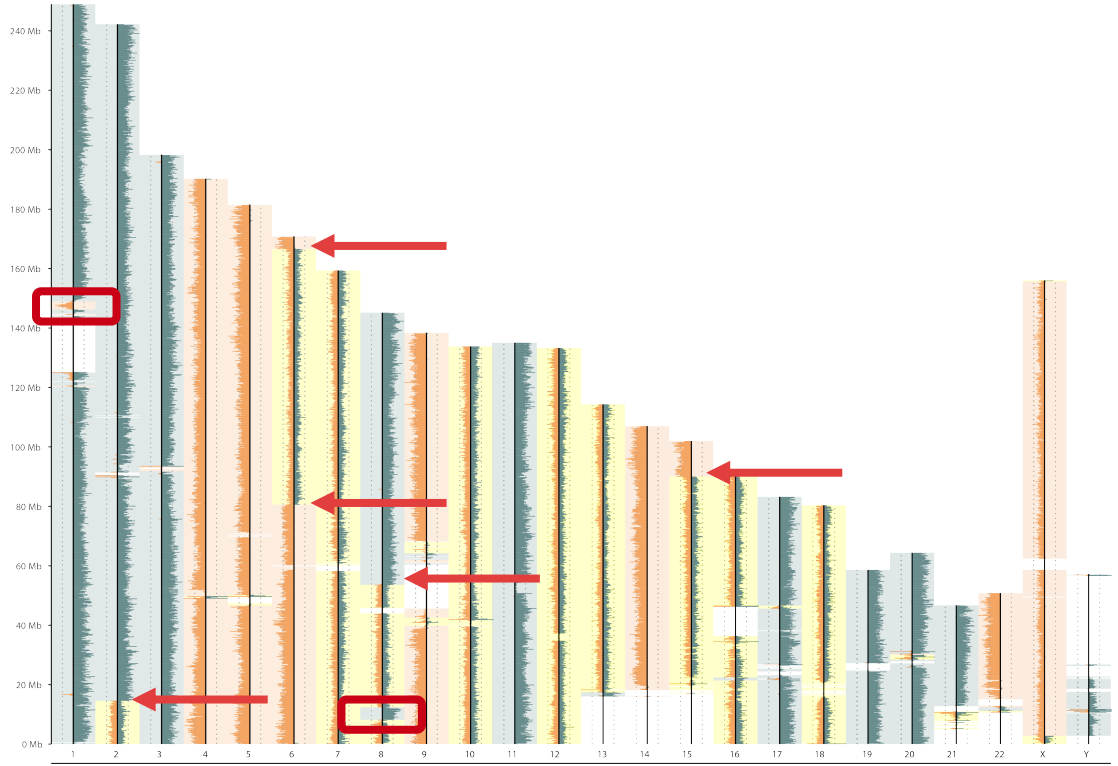
*(Adapted from Sanders et al., Nature Protocols **12**, (2017):1151–1176)*

Nuclei from cells hemi-substituted with BrdU can be identified based on the Hoechst profile





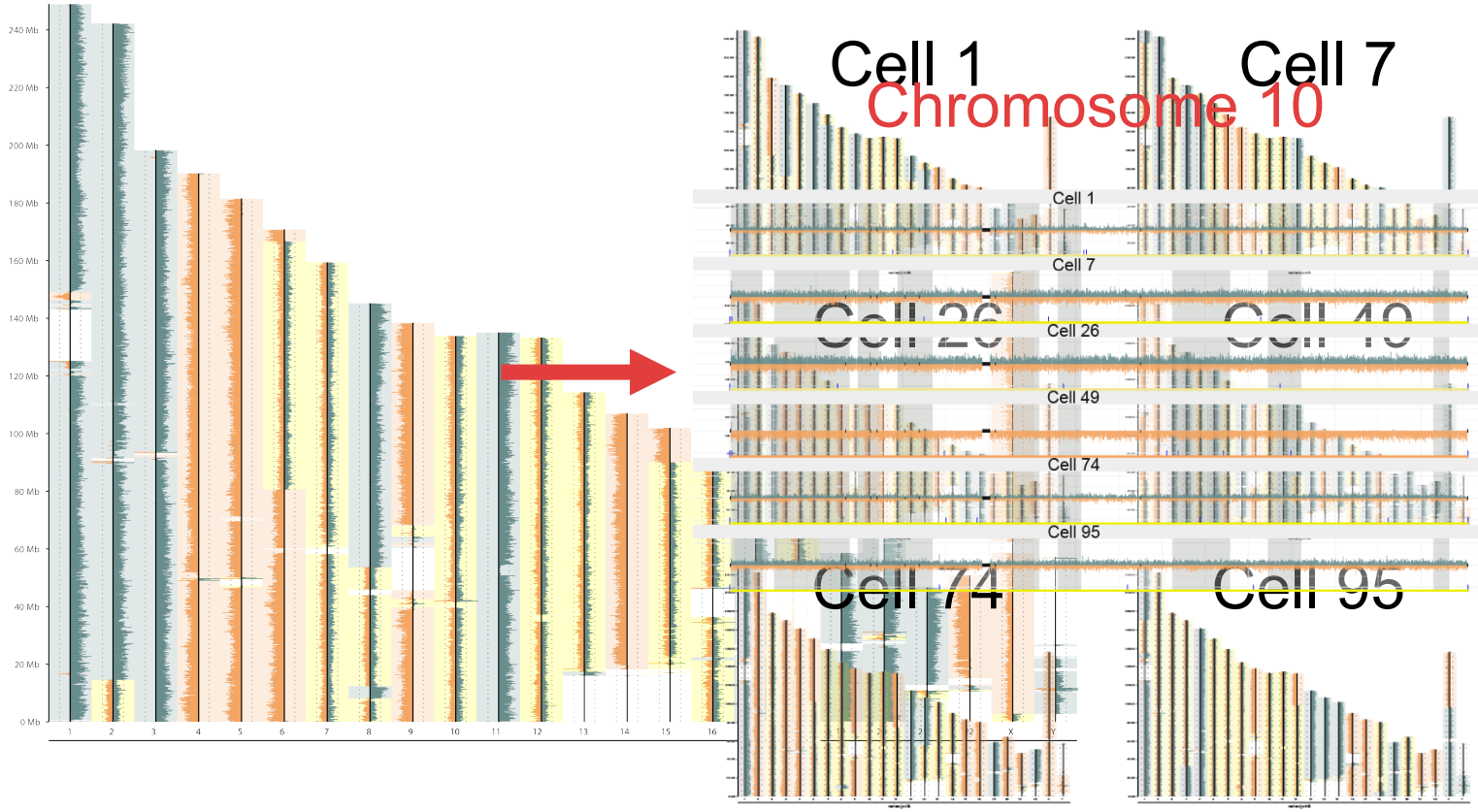
Strand-seq allows for structural variant detection in single cells at very low coverage



Median uniquely mapped reads: ~400,000

Median coverage: 0.03 X

Strand-seq allows for structural variant detection in single cells at very low coverage

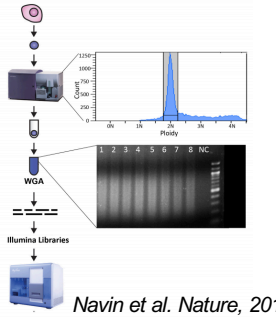


Overview of the single-cell genome data analysis – scWGS

Single nucleotide variation (SNVs) - scWGS

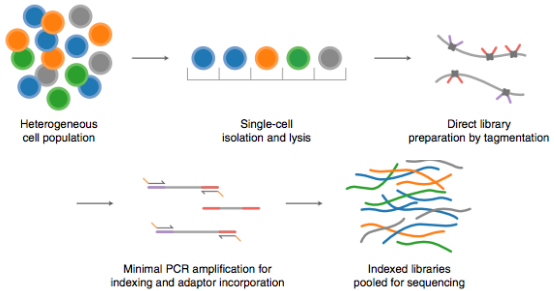
Structural Variation

Single-nucleus sequencing (SNS)



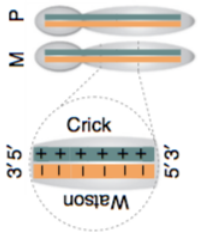
Navin et al. Nature, 2011

Direct library preparation (DLP)



Zahn et al. Nat Methods, 2017

Strand-seq



Sanders et al. Nat protocol, 2017

Step1. Alignment - Finding a correct position of reads: Bowtie2, BWA

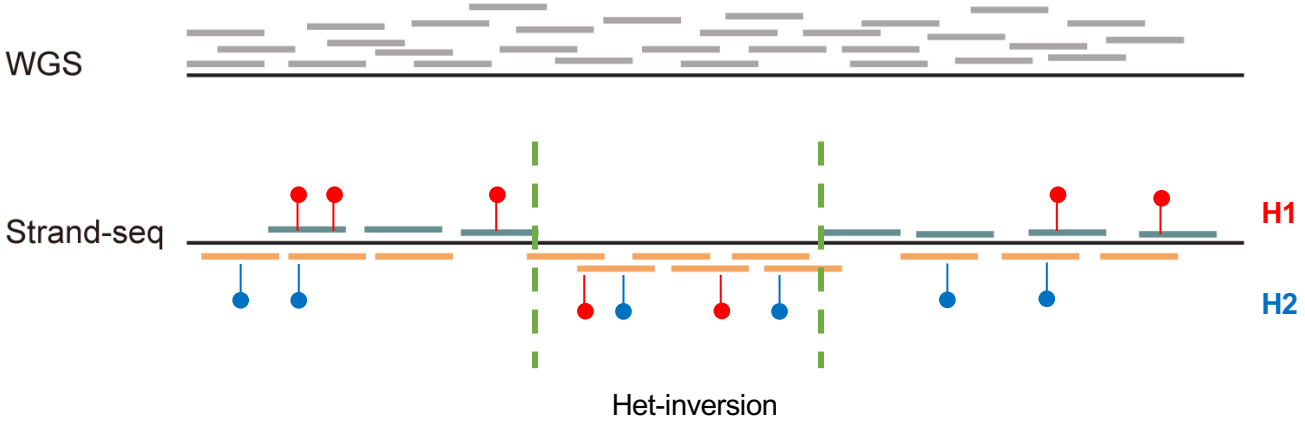
Step2. Remove PCR duplicate: Picard mark duplicate, Biobambam

Step3. Genotyping: Freebayes, GATK

Step4. Somatic mutation and CNV calling: SCcaller, Monovar, Aneufinder

Step5. Single-cell clustering and Phylogenetics: SCIPhi, TimeScape

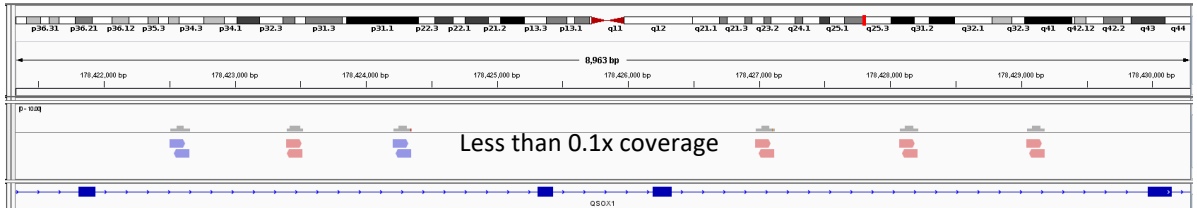
Specialties of the Strand-seq data analysis



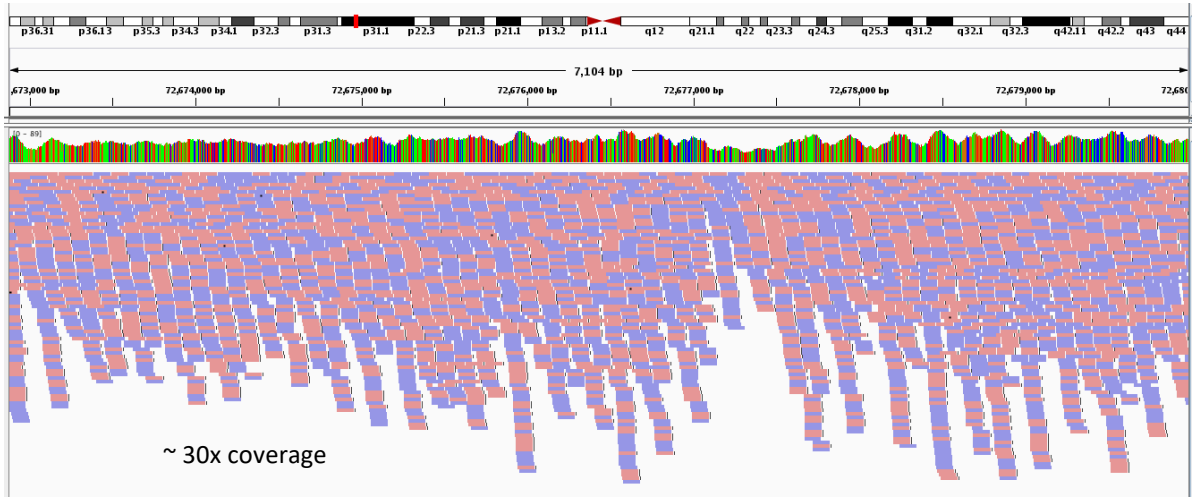
- **Sequence orientation is important (Crick or Watson)**
- **Breakpoint needs to be detected**
- **Strand state and haplotypes can be assigned**
- **Multiple types of structural variations need to be classified**

Challenges of the Strand-seq data analysis

Strand sequencing



Conventional sequencing (short-read)

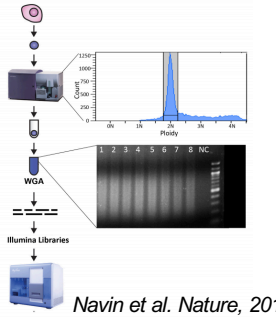


Overview of the single-cell genome data analysis – Strand-seq

Single nucleotide variation (SNVs) - scWGS

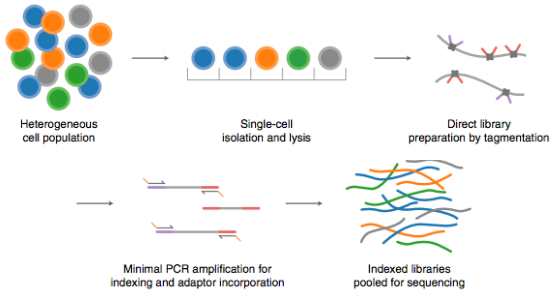
Structural Variation

Single-nucleus sequencing (SNS)



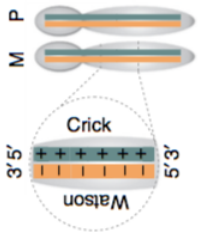
Navin et al. Nature, 2011

Direct library preparation (DLP)



Zahn et al. Nat Methods, 2017

Strand-seq



Sanders et al. Nat protocol, 2017

Step1. Alignment - Finding a correct position of reads: BWA, sequence orientation

Step2. Remove PCR duplicate: Biobambam

Step3. Genotyping, Haplotyping, Segmentation:, BreakpointR

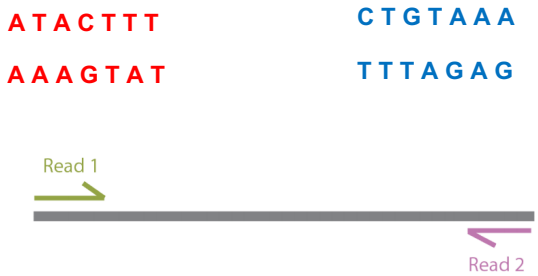
Step4. Structural variation calling: MosaiCatcher

Step5. Single-cell clustering and Phylogenetics

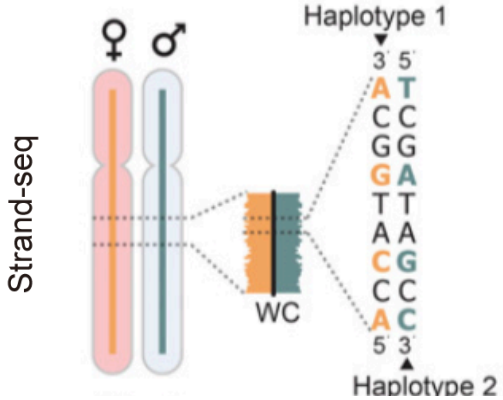
Quality checking!

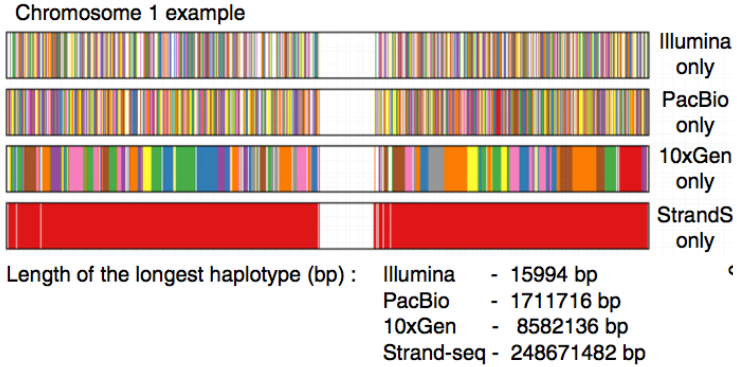
Fastq files are the starting point of Strand-seq data analysis

Paired-end
*_1_sequence.txt.gz *_2_sequence.txt.gz

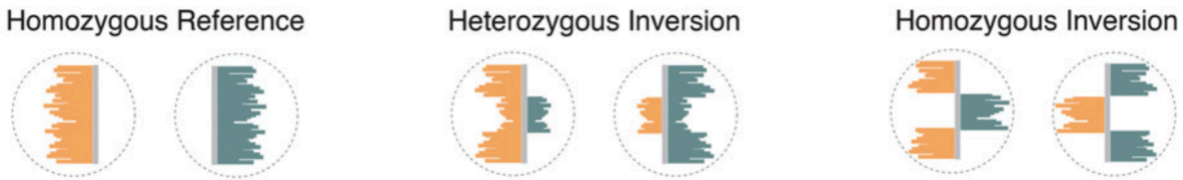


Why the orientation of the reads are important?





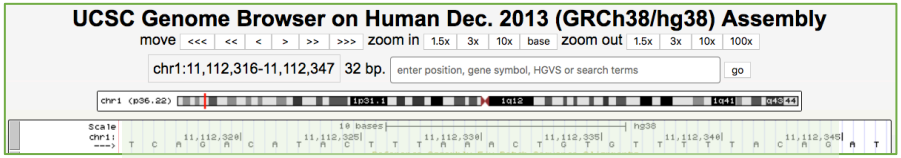
Porubsky et al. Nat comm, 2017



Sanders et al. Genome Res, 2016

How can we assign sequencing reads into Crick and Watson?

- Crick (C) aligns to the plus (forward) strand of the reference assembly
- Watson (W) aligns to the minus (reverse) strand



A T A C T T T

A A A G T A T

A T A C T T T

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

A A A G T A T

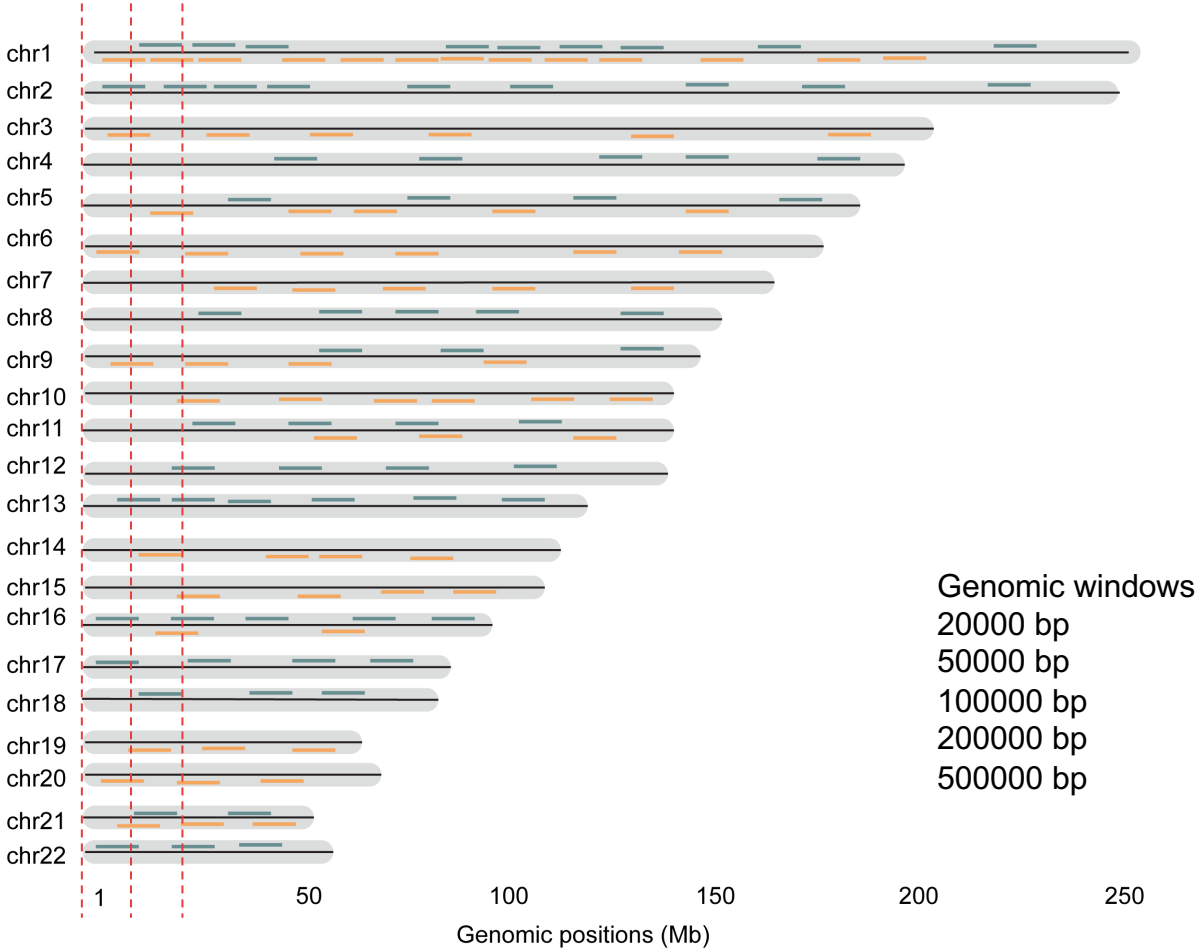
A T A C T T T Forward (+) → Crick (SAMFLAG 0)

A A A G T A T Reverse (-) → Watson (SAMFLAG 16)

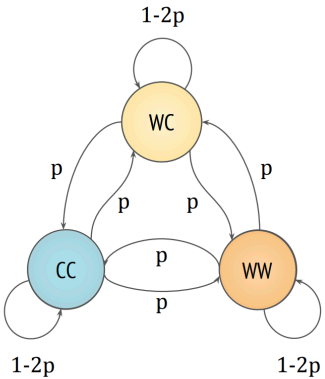
Crick = A1 read matches + or A2 read matches -

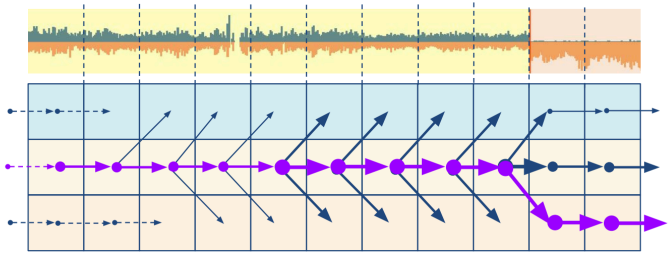
Watson = A1 read matches - or A2 read matches +

Count the Watson and Crick reads using genomic windows

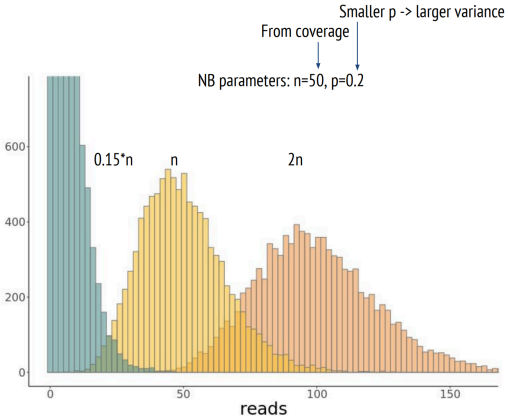


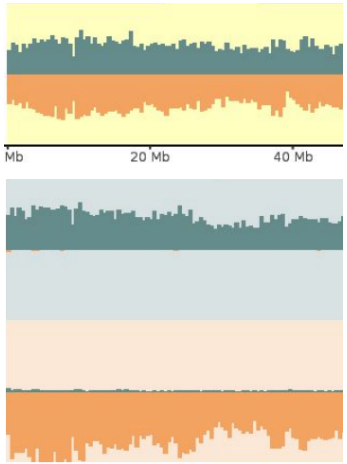
Strand states are called using a Hidden Markov Model





Arrows show the most probable sequence of state transitions
Thickness of line = probability of the path from start
Purple path is the most probable path in the end

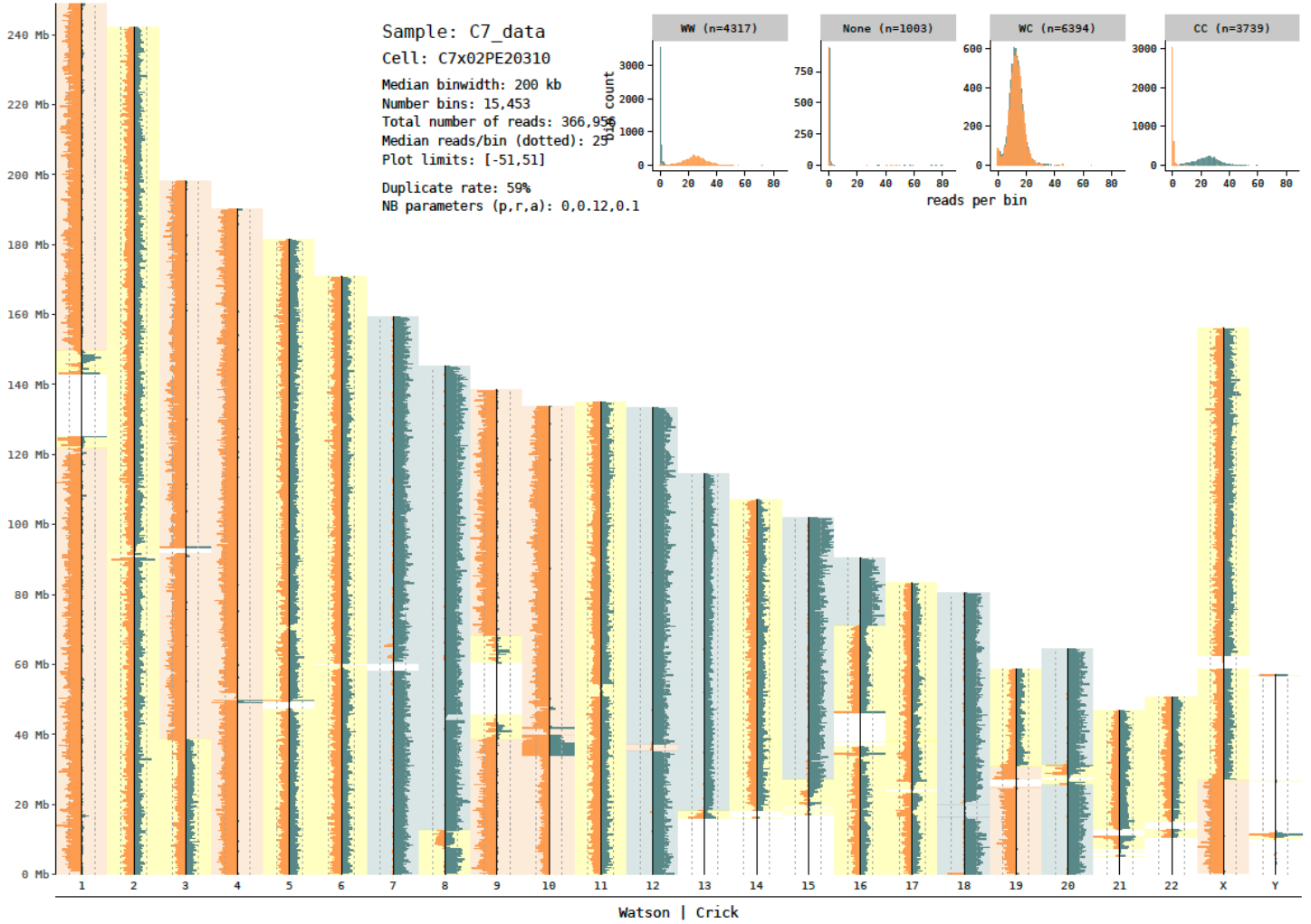




WC

CC

WW


Sample: C7_data
Cell: C7x02PE20310
Median binwidth: 200 kb
Number bins: 15,453
Total number of reads: 366,956
Median reads/bin (dotted): 25
Plot limits: [-51,51]
Duplicate rate: 59%
NB parameters (p,r,a): 0,0.12,0.1
WW (n=4317)
None (n=1003)
WC (n=6394)
CC (n=3739)
bin count
reads per bin
240 Mb
220 Mb
200 Mb
180 Mb
160 Mb
140 Mb
120 Mb
100 Mb
80 Mb
60 Mb
40 Mb
20 Mb
0 Mb
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 X Y
Watson | Crick

Processing the Strand-seq library after sequencing

Input: Paired-end fastq files : *_1_sequence.txt.gz *_2_sequence.txt.gz

Step1. Align reads, mark duplicates, add read groups

```
bwa mem -t {threads} \ -R '@RG\tID:{params.name}\tSM:{SAMPLE_NAME}\tLB:{wildcards.cell}' \
{params.genome} \ {input} \ | samtools view -bT {params.genome} - \ > {output}
bammarkduplicates markthreads={threads} I={input} O={output} index=1 rmdup=0
```

Step2. Filter & count bins in fixed windows

```
Rscript {params.qc_plot} {input.counts} {input.info} {output}
```

Step3. Plots and statistics

```
Rscript {params.sv_plot} {input} plot_chromosomes/{SAMPLE_NAME}/window_{wildcards.window}
```

Don’t worry, we can do it with one touch using Snakemake!

Sascha Meiers

Workflow management with Snakemake

- Snakemake is a Python-based workflow management tool

- It can run workflows consisting of multiple inter-dependent tasks. These can be computed either locally or on the cluster.
- Intermediate results are stored, so the pipeline only executes what is left to do.
- See http://snakemake.readthedocs.io or https://bitbucket.org/snakemake/snakemake/overview

```
rule bwa_map:
    input:
        "data/genome.fa",
        "data/samples/{sample}.fastq"
    output:
        "mapped_reads/{sample}.bam"
    shell:
        "bwa mem {input} | samtools view -Sb - > {output}"
```

```
rule samtools_sort:
    input:
        "mapped_reads/{sample}.bam"
    output:
        "sorted_reads/{sample}.bam"
    shell:
        "samtools sort -T sorted_reads/{wildcards.sample} "
        "-O bam {input} > {output}"
```

What will be happened in the pipeline?

- Align to GRCh38 (recommended version `GCA_000001405.15_GRCh38_no_alt_analysis_set.fna`)
- During that step, add read groups (1 `SM` tag per sample, but different (possibly multiple) `ID`s per cell)
- Mark duplicates
- Count reads of all cells in bins (20,50,100, and 200kb)
- Plot overviews of all cells
- Plot chromosome by chromosome

Job overview

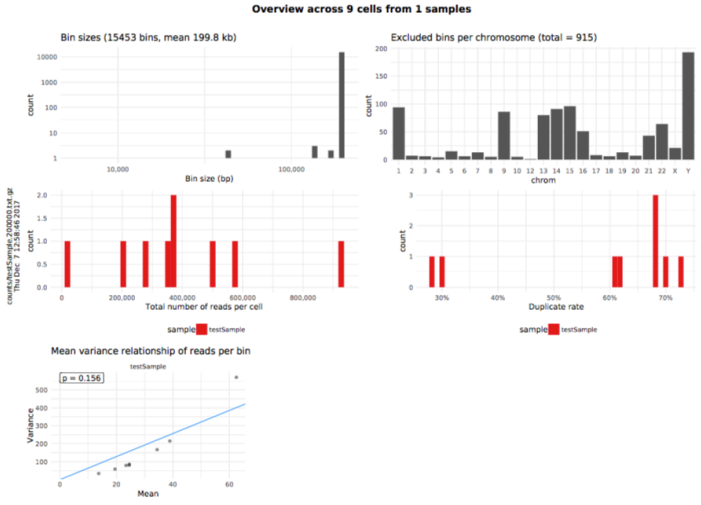
```
count
1   all
9   index
9   map_reads
9   markdups
5   mosaic_count
5   plot_chromosomes
5   plot_mosaic_counts
9   sort_bam
52
```

Job dependecies

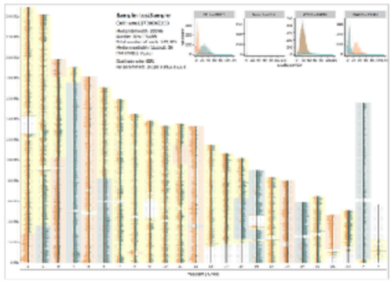
Output of the pipeline

- Three different plots for quality checking and first look for the SVs
- Count matrix for read depth and ploidy diagnosis
- Bam files from this pipeline will be used as an input of downstream advanced analysis

Overview plot



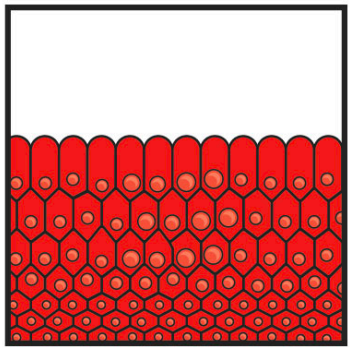
Each single cell



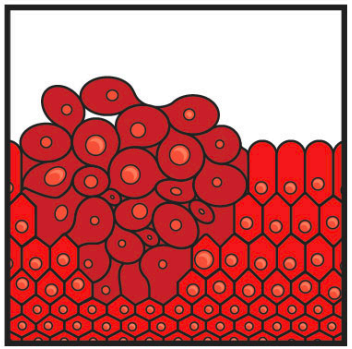
Single chromosomes

Which data do we analyze today?

Real world



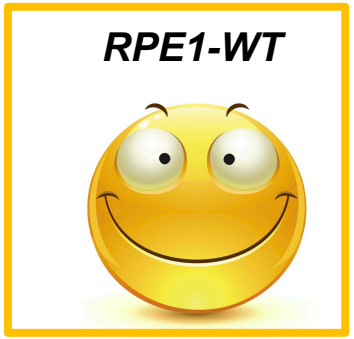
Normal cells



Cells forming a tumour

Model system

RPE1-WT

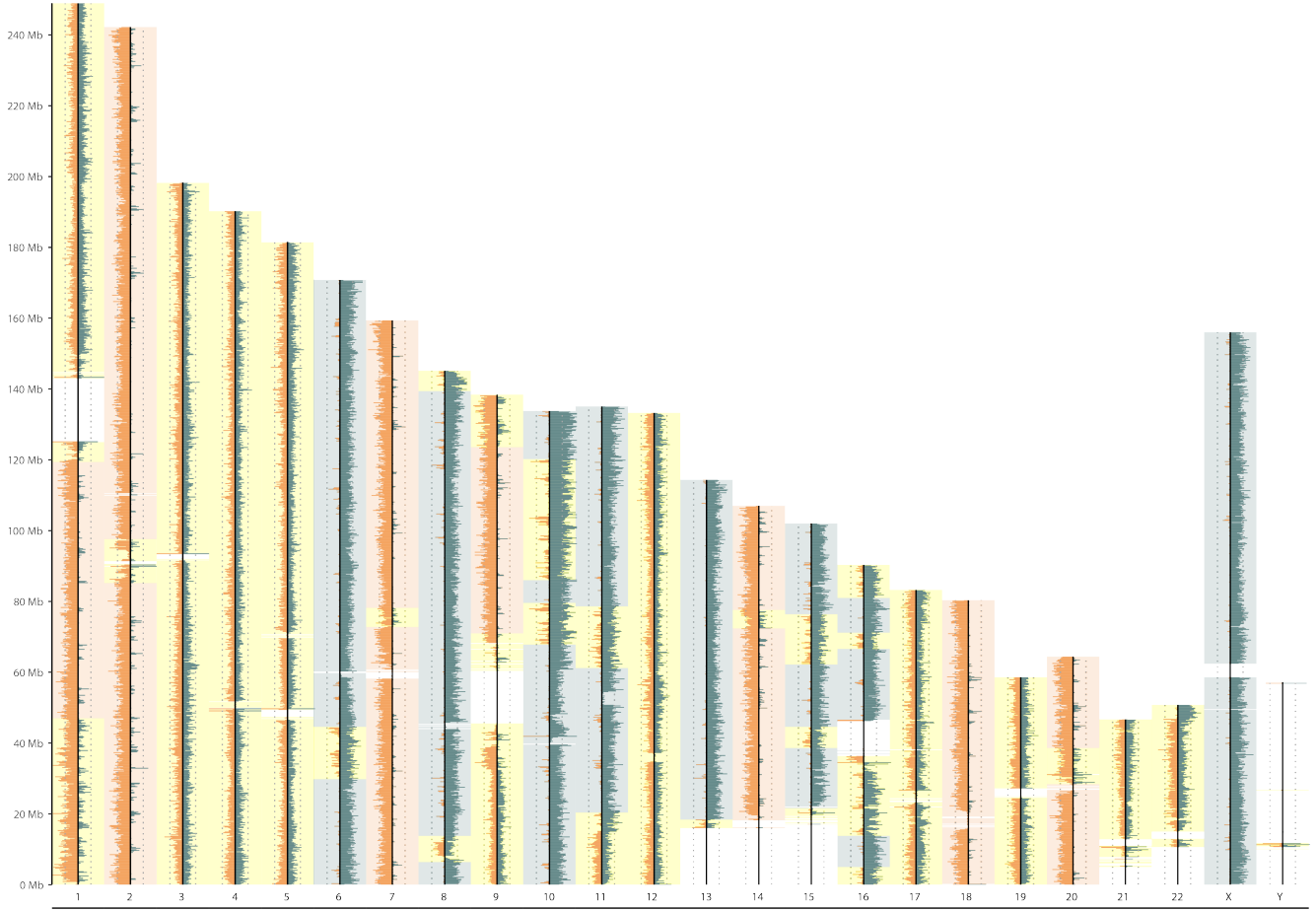


vs

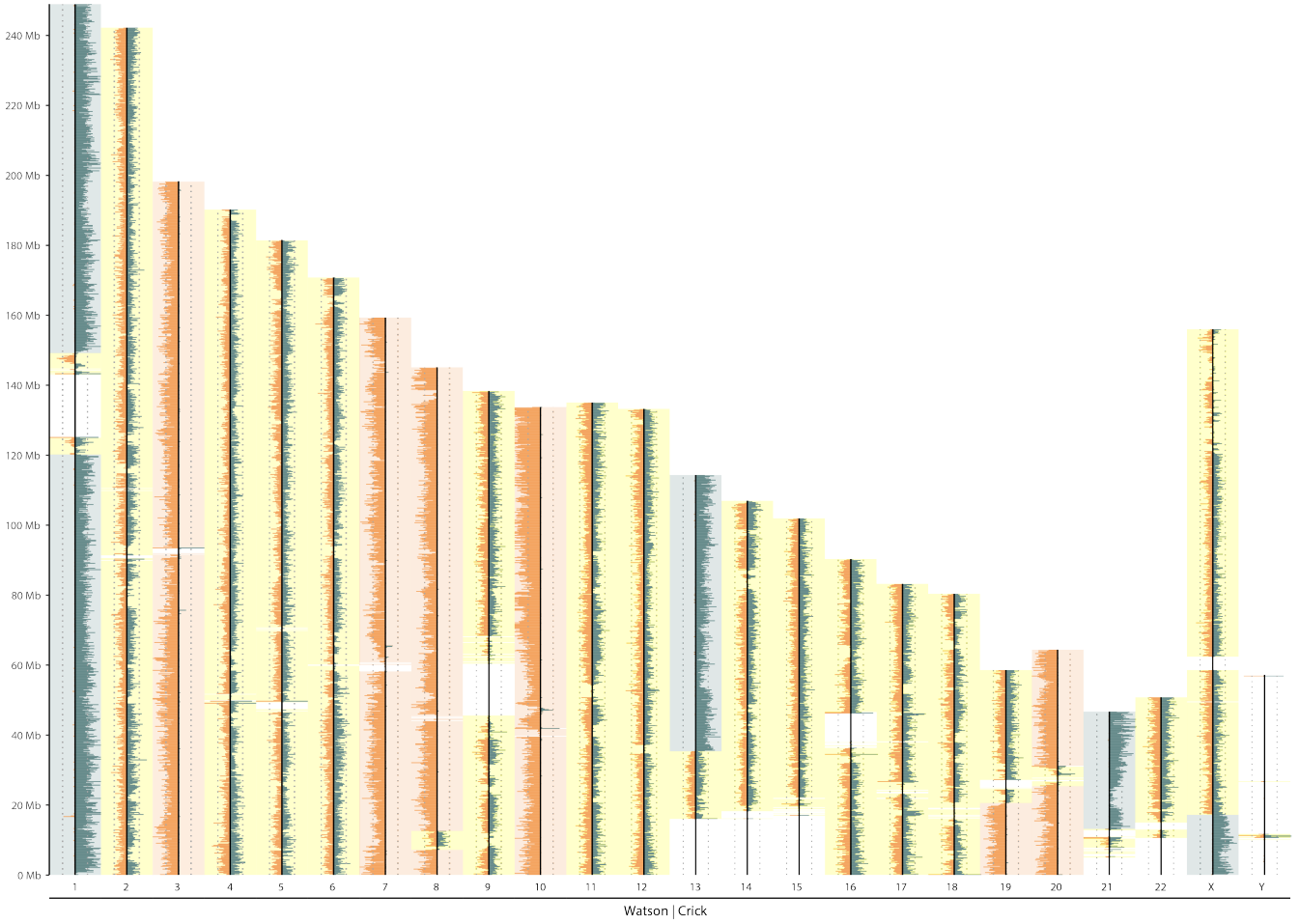
RPE1-BM510

https://github.com/jeongdo801/SV_practical_computational

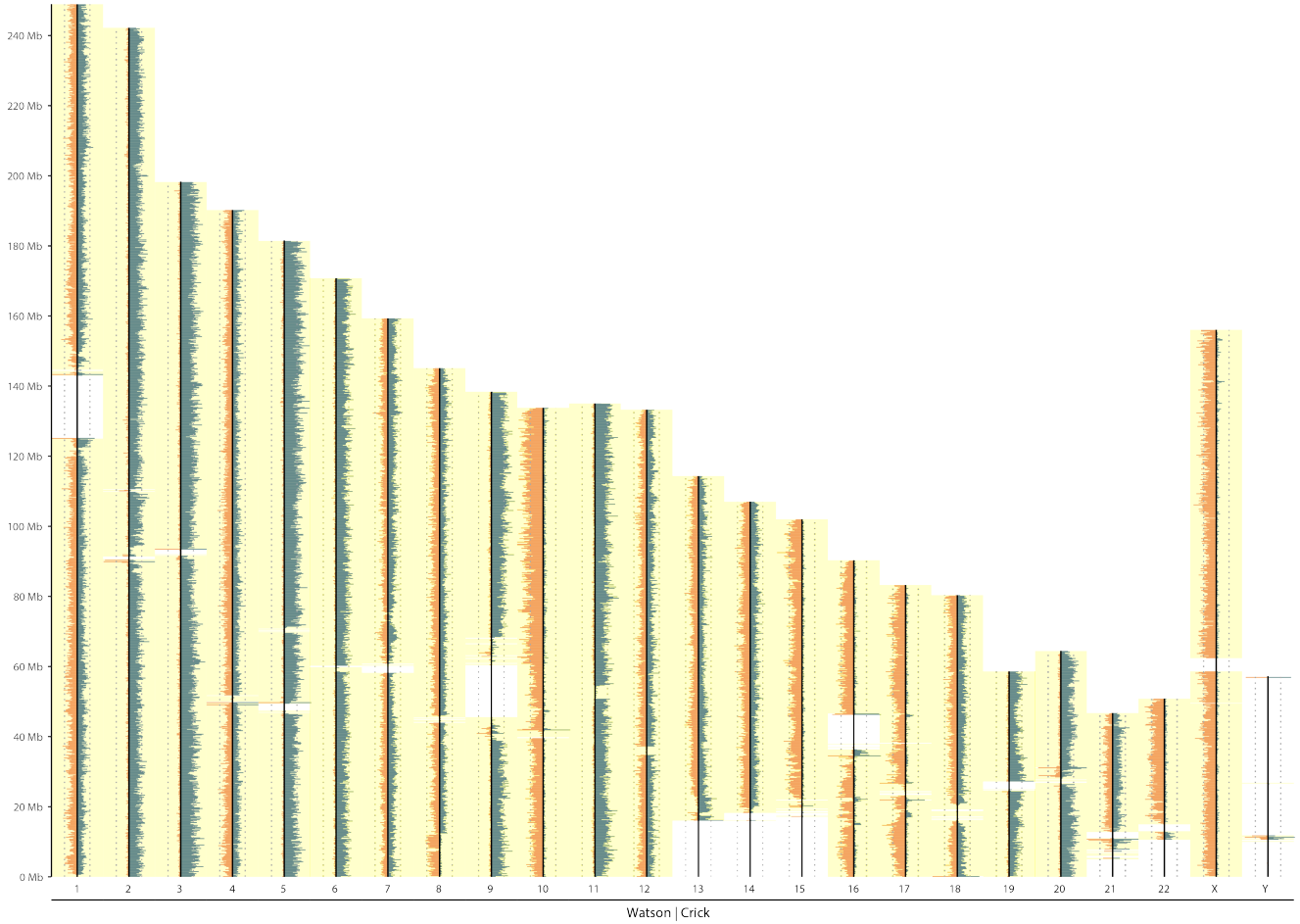
Filtering Strand-seq Libraries: Under-BrdU incorporation



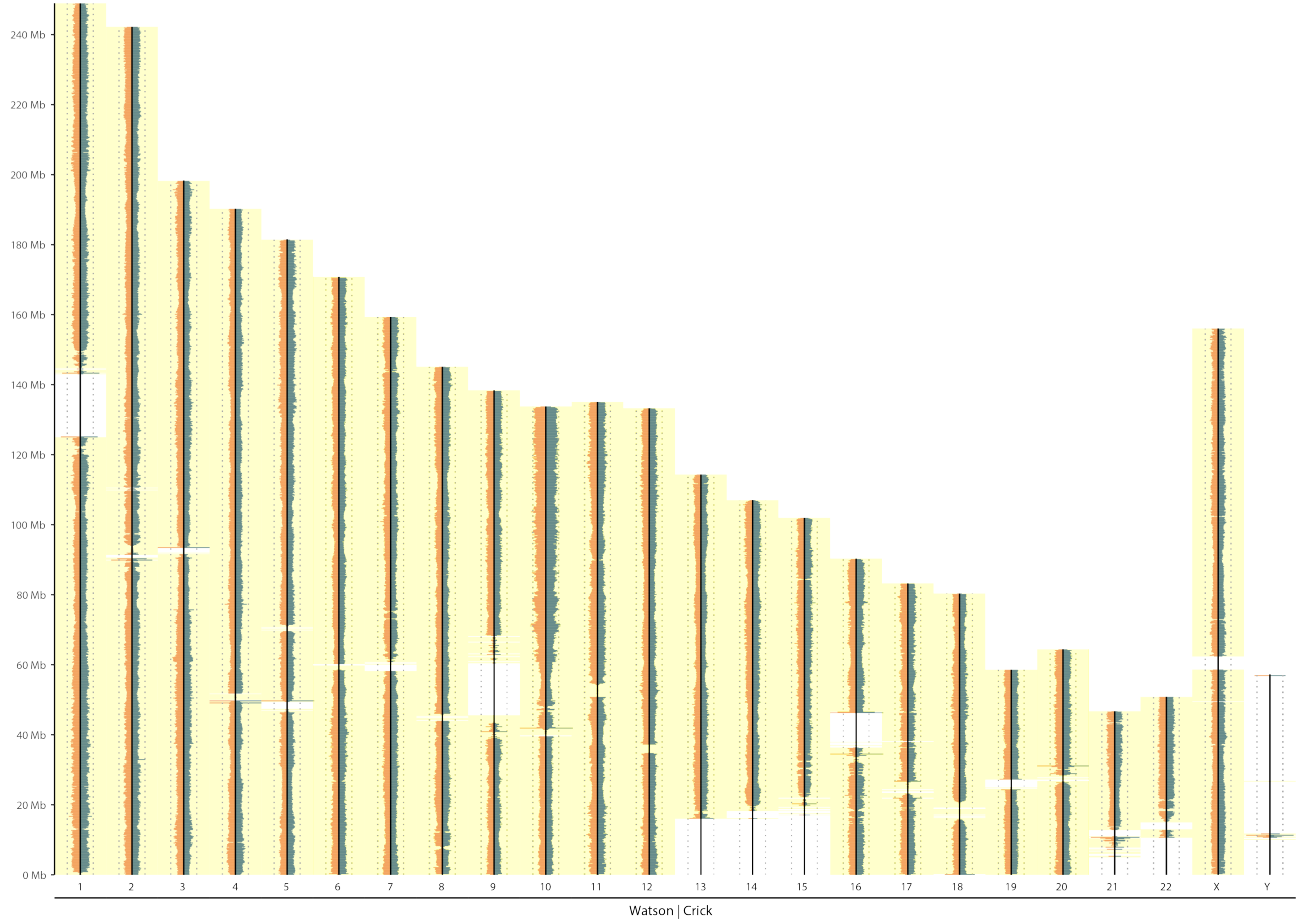
Filtering Strand-seq Libraries: Over-BrdU incorporation



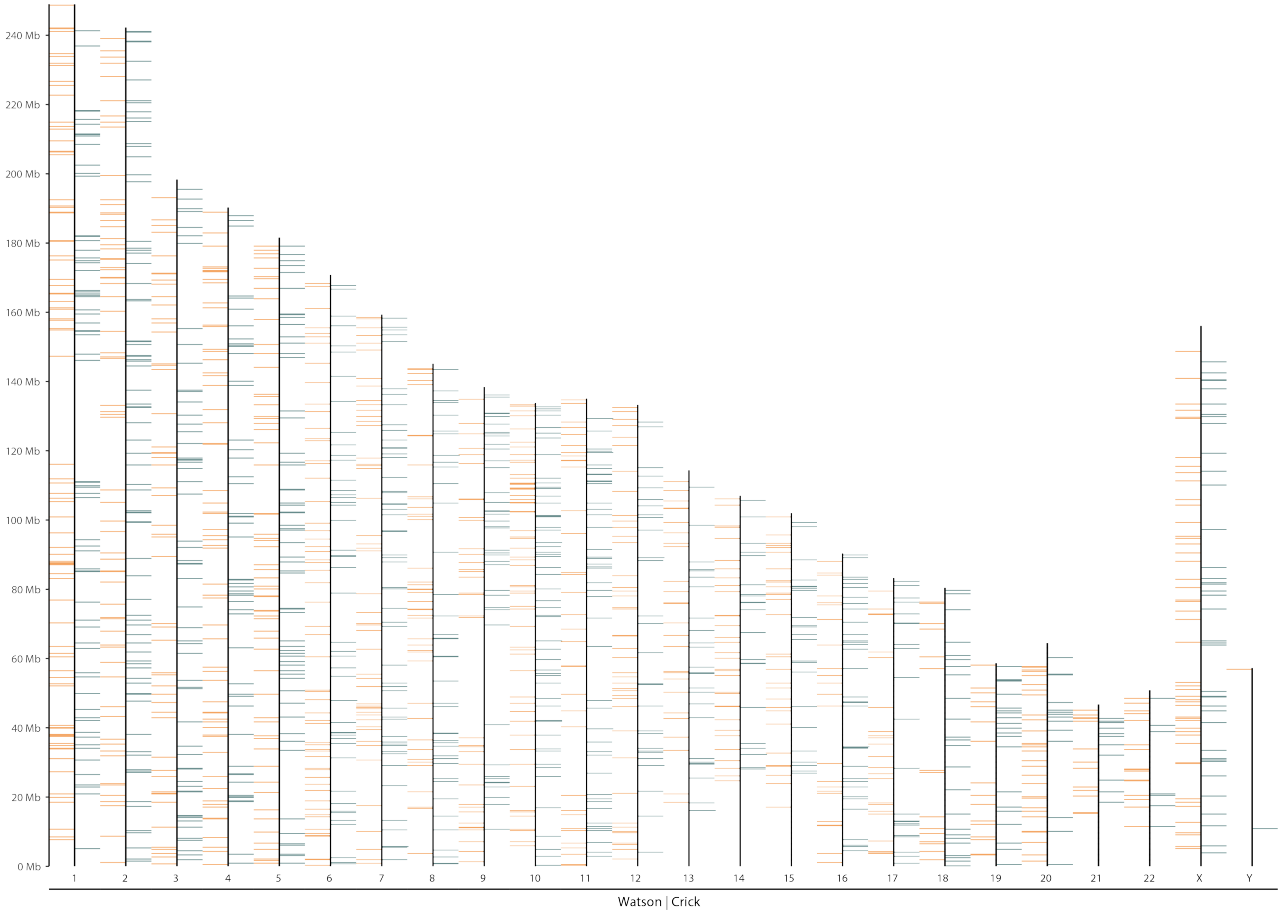
Filtering Strand-seq Libraries: Background Contamination



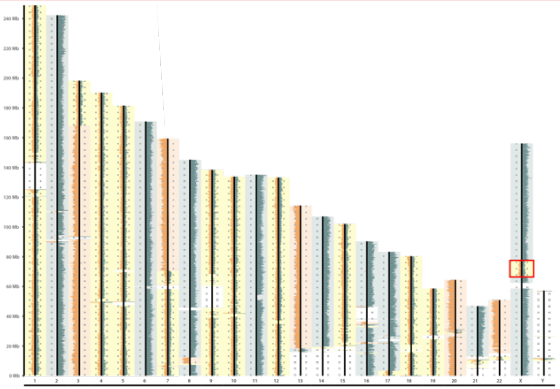
Filtering Strand-seq Libraries: Multi-cell (Positive Control)



Filtering Strand-seq Libraries: 0-cell (Negative Control)

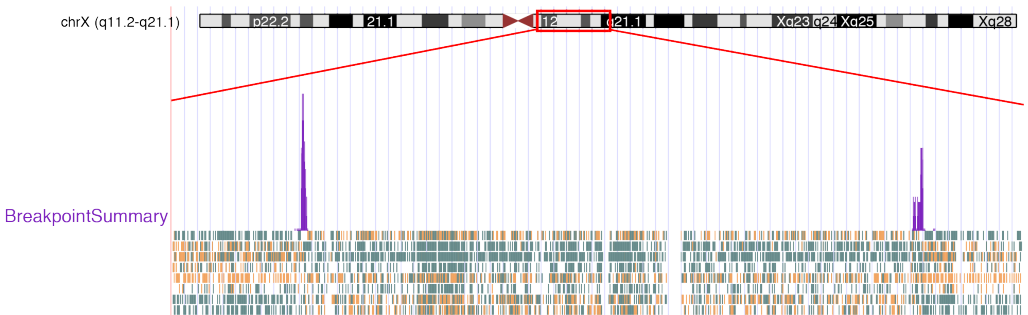


BreakPointR analysis allows refinement of breakpoints and alignment of Strand-seq data to public datasets on UCSC



(github.com/daewoooo/BreakPointR)

Input:
- Curated Strand-seq bam files



Output:
- Bed files
- Breakpoint summaries

(genome.ucsc.edu)

Thank you for listening

Day3

BREAK POINT