

Constructing an open ecosystem for bioinformatics and genomics big data

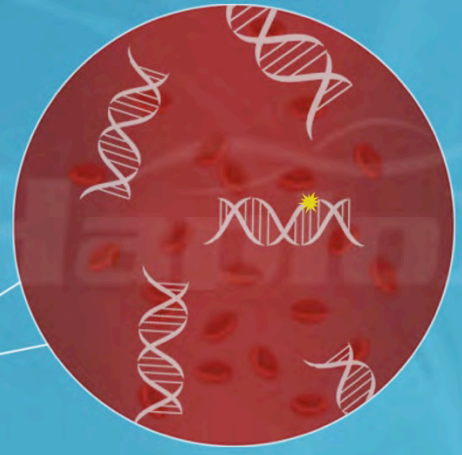
Shifu Chen Ph.D
Co-founder & CTO
HaploX Biotechnology
2018-10-13

肺癌
Lung Cancer

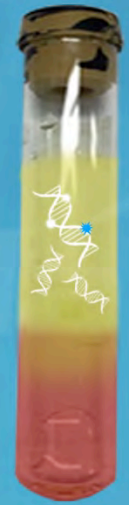
乳腺癌
Breast Cancer

胃癌
Gastric Cancer

结直肠癌
Colorectal Cancer



10mL blood



个性化靶向治疗
Personalized Targeted Therapy

耐药分析
Resistance Analysis

Stage I , II

Stage III , IV

早期癌症检测
Early Detection

预后监测
Prognostic Surveillance

HaploX Genomics Center: a large sequencing center

QX200 ddPCR



2* NextSeq 500/550



2* HiSeq X Ten



10* NovaSeq 6000



Able to sequence 60,000 whole genomes per year



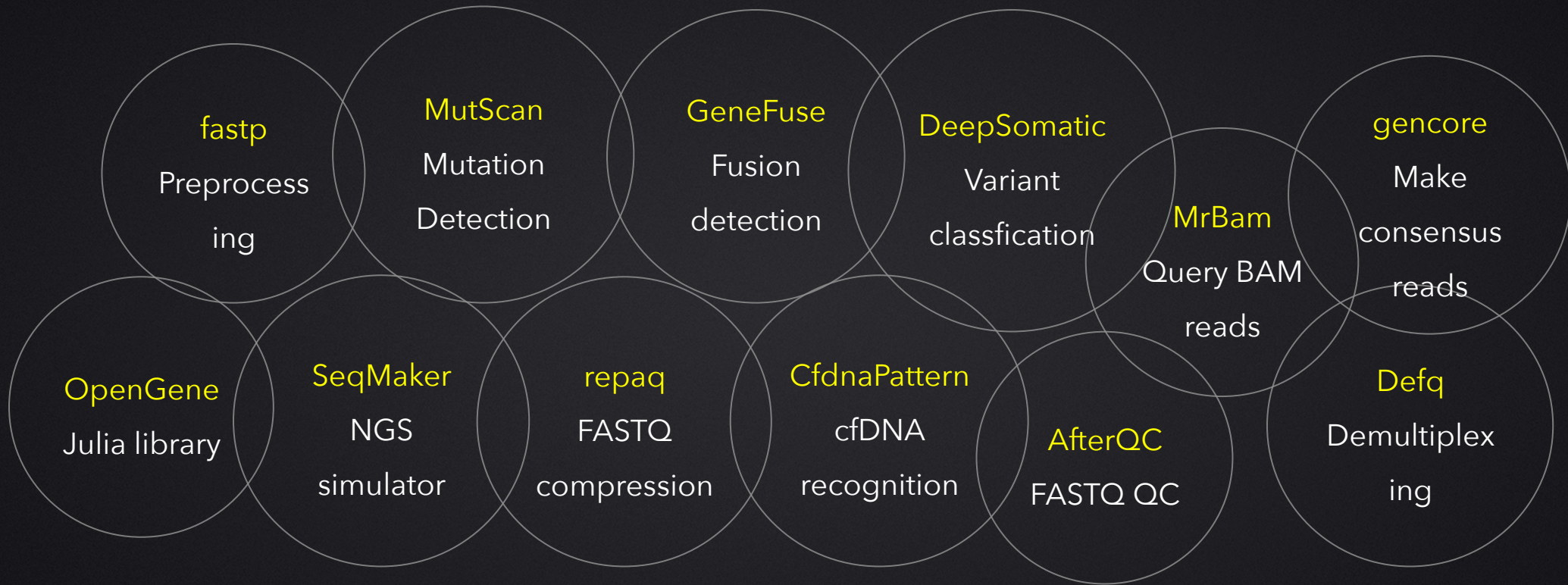
To address some common bioinformatics problems, I started

OpenGene

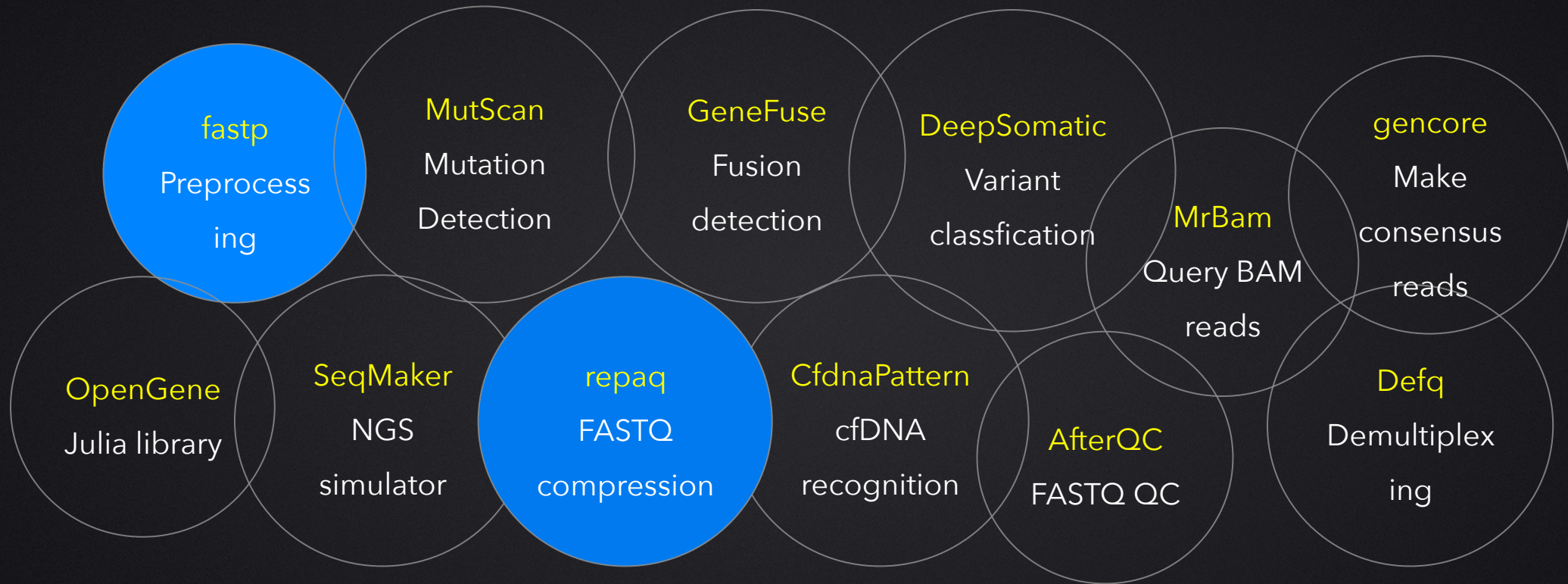
Since Apr. 2016

<https://github.com/OpenGene>

Developed 10+ open source software

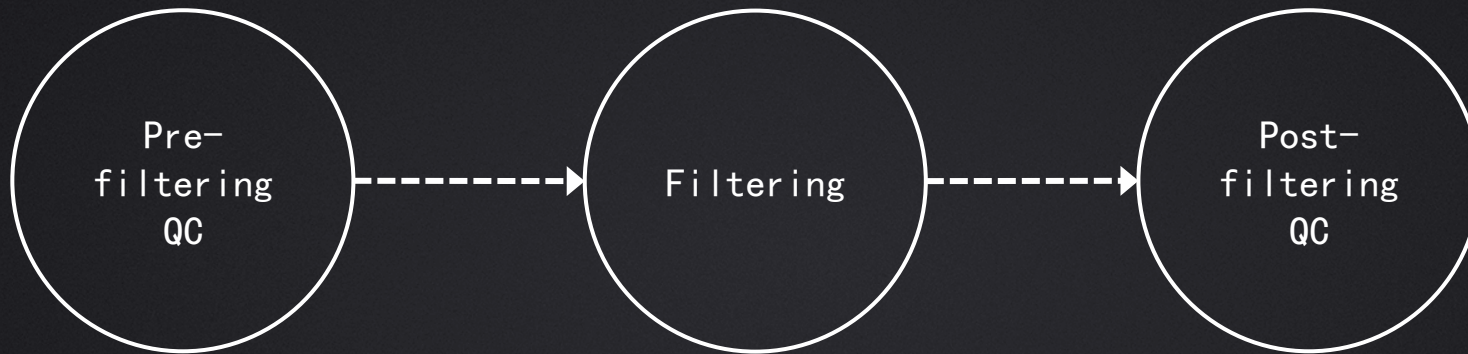


Developed 10+ open source software



Why do we need another FASTQ preprocessor?

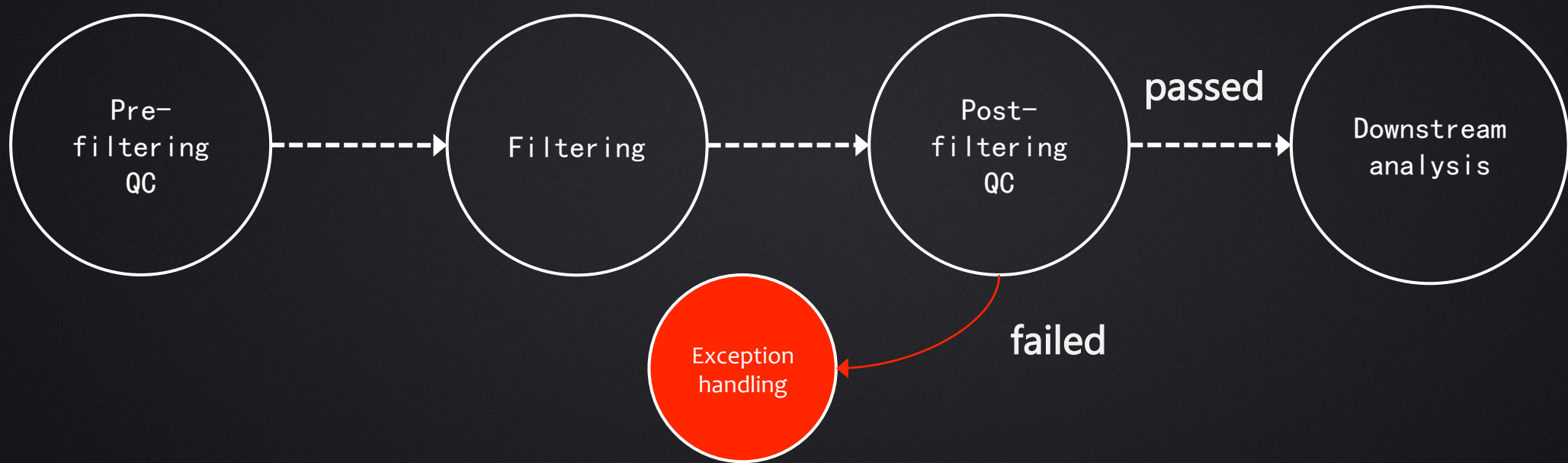
How did we do NGS data preprocessing?



1, QC + filtering can be combined to reduce extra computation and I/O

Why do we need another FASTQ preprocessor?

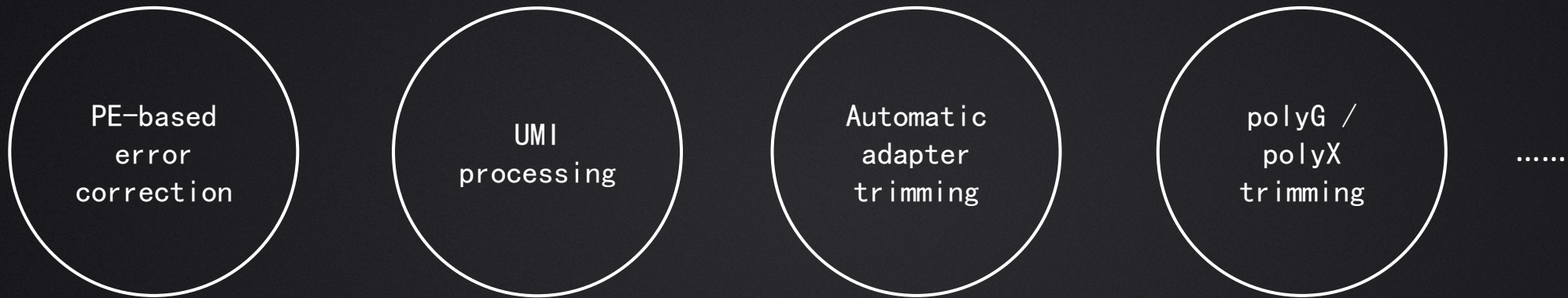
We hope that the QC process can be automatic, so figures are not enough



2, the preprocessor should provide QC report readable for both human and computer

Why do we need another FASTQ preprocessor?

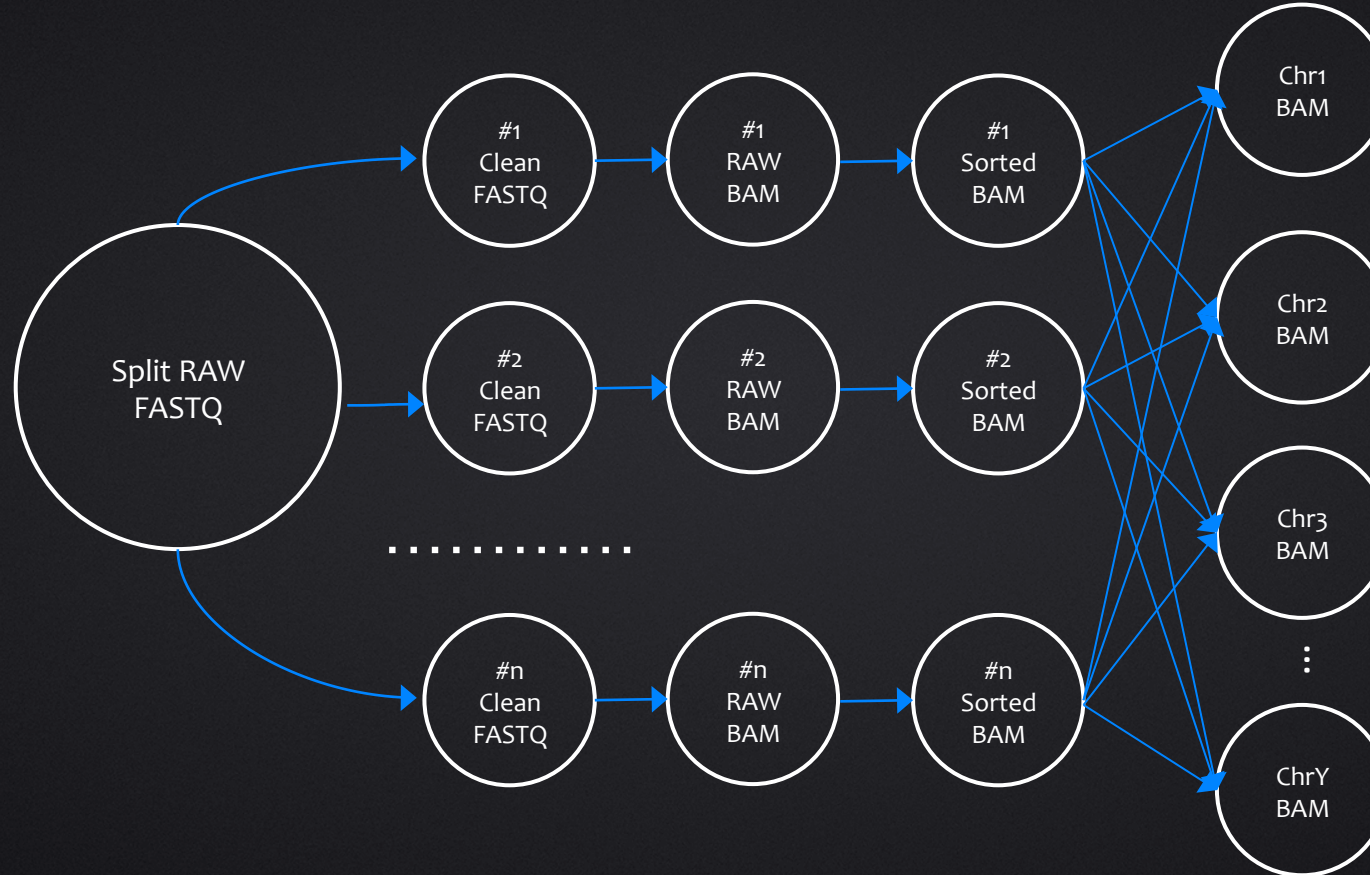
Some functions we were expecting...



3, some functions must be integrated into the preprocessing step

Why do we need another FASTQ preprocessor?

We want to parallelize and stream our pipeline



4, the preprocessor should support STDIN, STDOUT and splitting the data

| Why do we need another FASTQ preprocessor?

Wait for 10 hours to QC and filter WGS data?

Speed

5, the preprocessor should be ultra-fast



fastp: an ultra-fast all-in-one FASTQ preprocessor

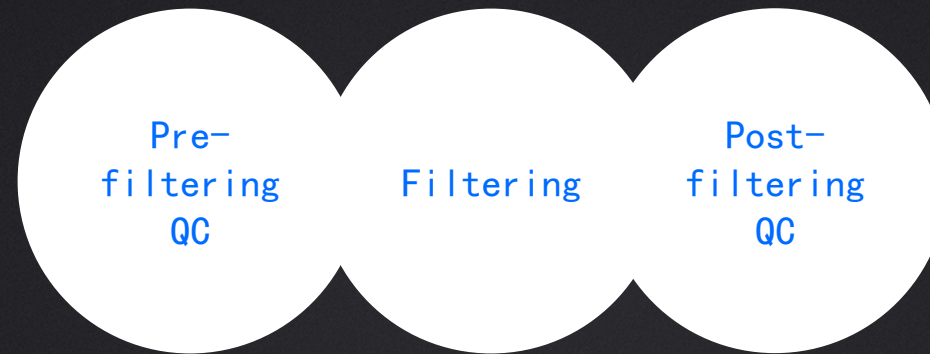


<https://github.com/OpenGene/fastp>

★ 361

| fastp: an ultra-fast all-in-one FASTQ preprocessor

QC + filtering in a single I/O pass



1, QC + filtering can be combined to reduce extra computation and I/O

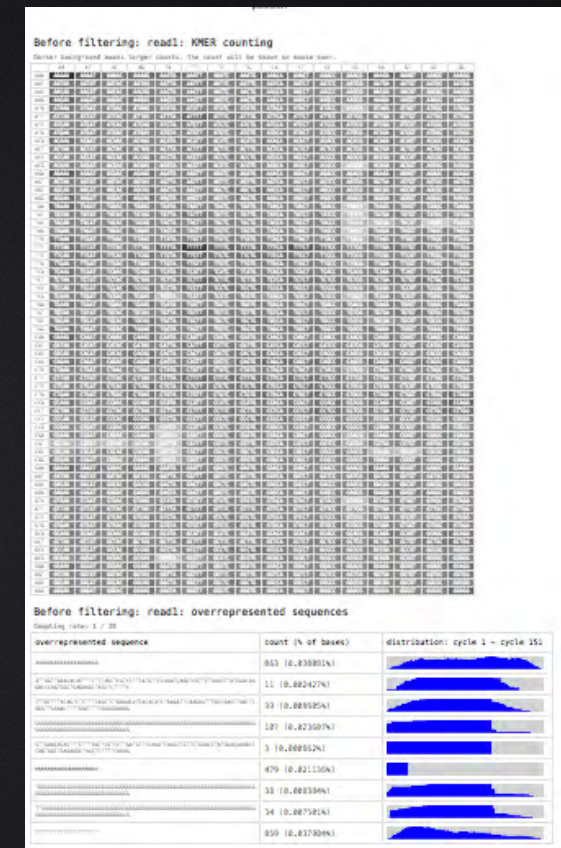
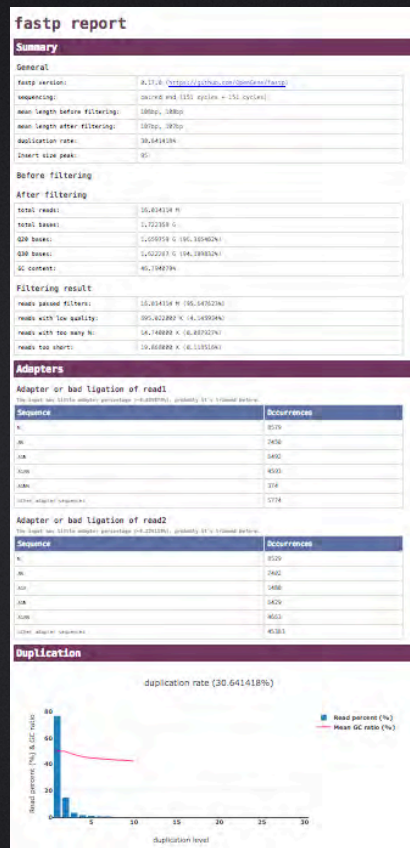
fastp: performs quality control, adapter trimming, quality filtering, per-read quality pruning, and many other operations with a single scan of the FASTQ data

fastp: an ultra-fast all-in-one FASTQ preprocessor

```

fastp.json
{
  "summary": {
    "before_filtering": {
      "total_reads": 5000,
      "total_bases": 735556,
      "q20_bases": 713280,
      "q30_bases": 678519,
      "q20_rate": 0.969715,
      "q30_rate": 0.922457,
      "read1_mean_length": 147,
      "read2_mean_length": 147,
      "gc_content": 0.494407
    },
    "after_filtering": {
      "total_reads": 4954,
      "total_bases": 728474,
      "q20_bases": 706421,
      "q30_bases": 671975,
      "q20_rate": 0.969727,
      "q30_rate": 0.922442,
      "read1_mean_length": 147,
      "read2_mean_length": 147,
      "gc_content": 0.494523
    }
  },
  "filtering_result": {
    "passed_filter_reads": 4954,
    "low_quality_reads": 0,
    "too_many_N_reads": 0,
    "too_short_reads": 2,
    "too_long_reads": 0
  },
  "duplication": {
    "rate": 0.0280876,
    "histogram": [2288, 62, 3, 0, 0, 0],
    "mean_gc": [0.490616, 0.498102,

```



2, the preprocessor should provide QC report readable for both human and computer

fastp: generates human readable JSON report + interactive HTML report

fastp: an ultra-fast all-in-one FASTQ preprocessor

FEATURES:

- 1 Profile the FASTQ data before and after filtering (quality curves, base contents, KMER, Q20/Q30, duplication, read length distribution, adapter contents...)
- 2 Filter out bad reads (with too low quality, low complexity, too short, or too many N...)
- 3 Cut low quality bases for per read in its 5' and 3' by evaluating the mean quality from a sliding window
- 4 Trim all reads in front and tail
- 5 Detect and cut adapters automatically detected, without the need to input the adapter sequences.
- 6 Correct mismatched base pairs in overlapped regions of paired end reads, if one base is with high quality while the other is with ultra low quality
- 7 Trim polyG and polyX artifacts in 3' ends, which are commonly seen in NovaSeq/NextSeq data.
- 8 Preprocess unique molecular identifier (UMI) enabled data by shifting UMI to sequence identifiers.
- 9 Generate JSON report for down-stream interpreting and HTML report with interactive visualization
- 10 Split a FASTQ file to multiple files to support parallel processing.

3, some functions must be integrated into the preprocessing step

fastp: auto filtering, adapter trimming, quality pruning, base correction, polyG/polyX trimming, UMI, insert-size profiling, duplication profiling and many more...

| fastp: an ultra-fast all-in-one FASTQ preprocessor

Splitting:

1. by limiting total split file number
2. by limiting split file size

Streaming:

1. read from STDIN or write to STDOUT
2. automatic interleaved for PE data

4, the preprocessor should support STDIN, STDOUT and splitting the data

fastp: supports splitting when filtering, streaming (STDIN + STDOUT)

fastp: an ultra-fast all-in-one FASTQ preprocessor

| Tool | Input | Time (min) | Throughput (reads/s) |
|-------------|-------|------------|----------------------|
| fastp | SE | 6 | 129397.9 |
| | PE | 13.3 | 116750.0 |
| FASTQC | SE | 13 | 59722.1 |
| | PE | 25.8 | 60185.1 |
| Cutadapt | SE | 18 | 43132.6 |
| | PE | 24.6 | 63120.9 |
| SOAPnuke | SE | 30.7 | 25289.5 |
| | PE | 32.5 | 47777.7 |
| AfterQC | SE | 25.2 | 30809.0 |
| | PE | 57.2 | 27146.4 |
| Trimmomatic | SE | 31.9 | 24338.2 |
| | PE | 60.9 | 25497.1 |

Table 1. Speed comparison of fastp and other software. Results for both paired-end (PE) and single-end (SE) input were compared.

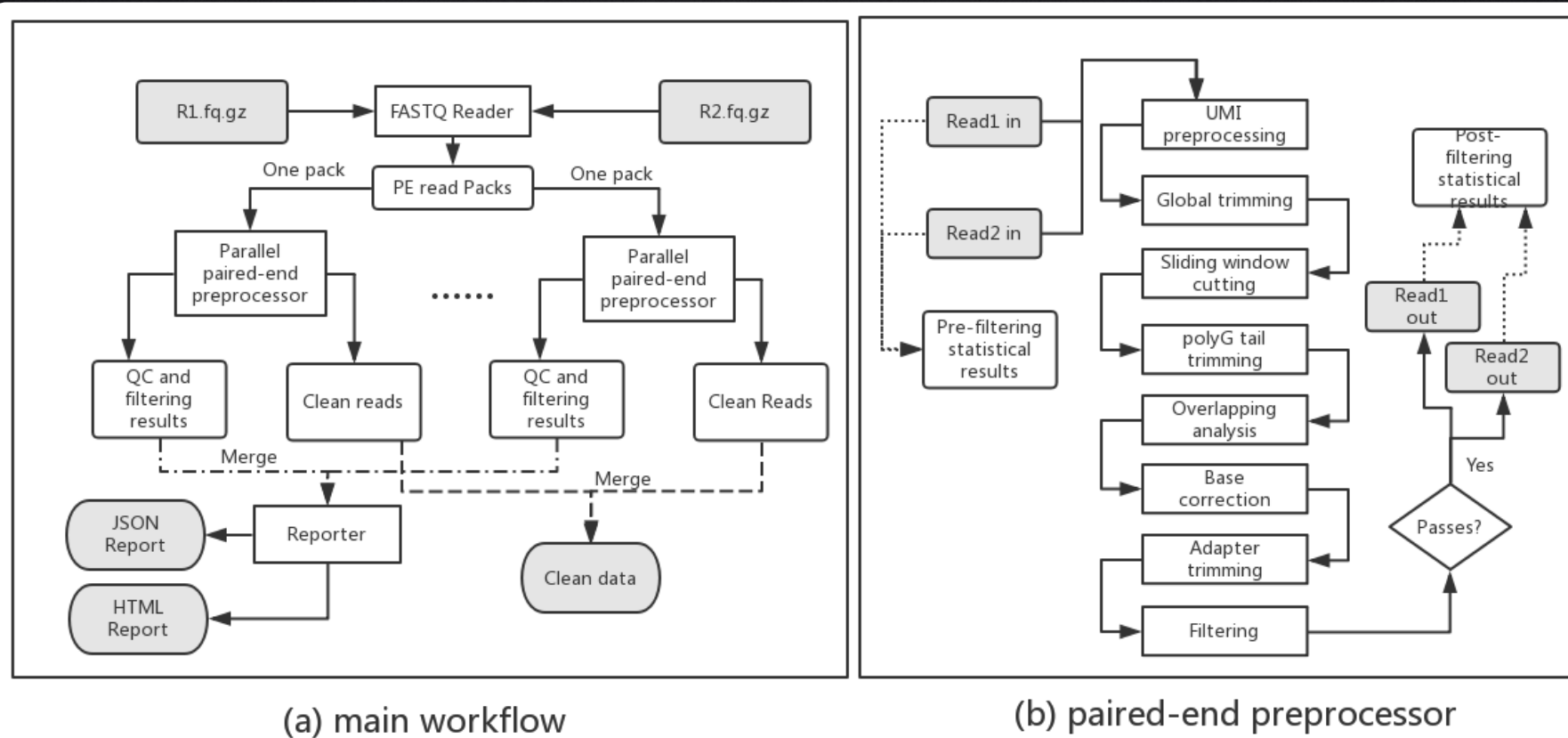
Remember that fastp does much more than other tools in this evaluation

20X WGS
in one hour!

5, the preprocessor should be ultra-fast

fastp: developed in C++ with solid multi-threading support, 2~5X faster than other tools

How we make it?



The workflow of fastp

(a) Main workflow of paired-end data processing, and (b) paired-end preprocessor of one read pair. In the main workflow, a pair of FASTQ files is loaded and packed, after which each read pair is processed individually in the paired-end preprocessor, described in (b).

PE overlap analysis for error correction

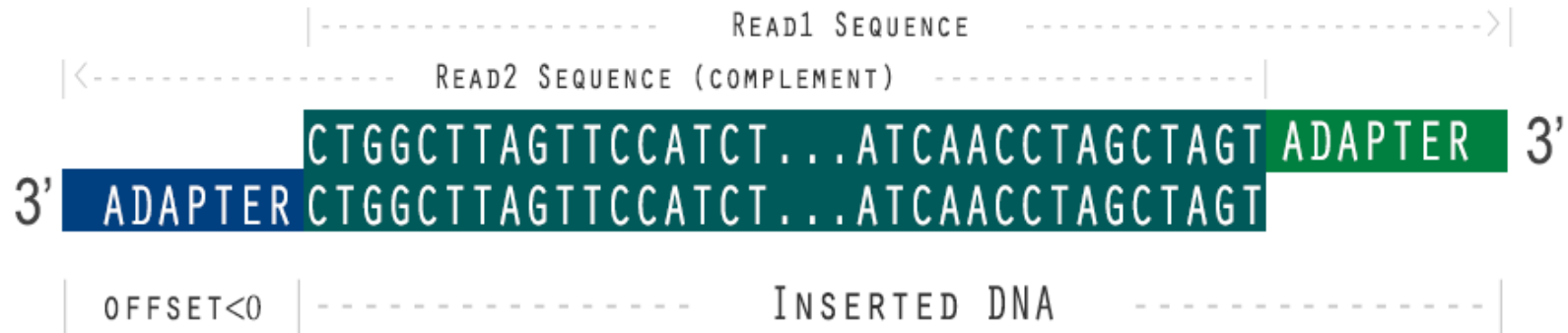
5' TTTAGGCCTGTCACGTGTGAACGCTATCAGCAAGCCTTTGCATGATTTTTC
TCACTGTGAACGCTATCTGCAAGCCTTTGCATGATTTTCTCTTTCCCAC 5'

■ R1 ■ R2 (reverse complement) ■ overlapped ■ mismatch with a low quality base

Error correction for PE data by overlap analysis

For each pair of reads, fastp computes their overlapping region and correct the unbalanced mismatches (1 extreme high quality base + 1 extreme low quality base)

PE overlap analysis for adapter trimming



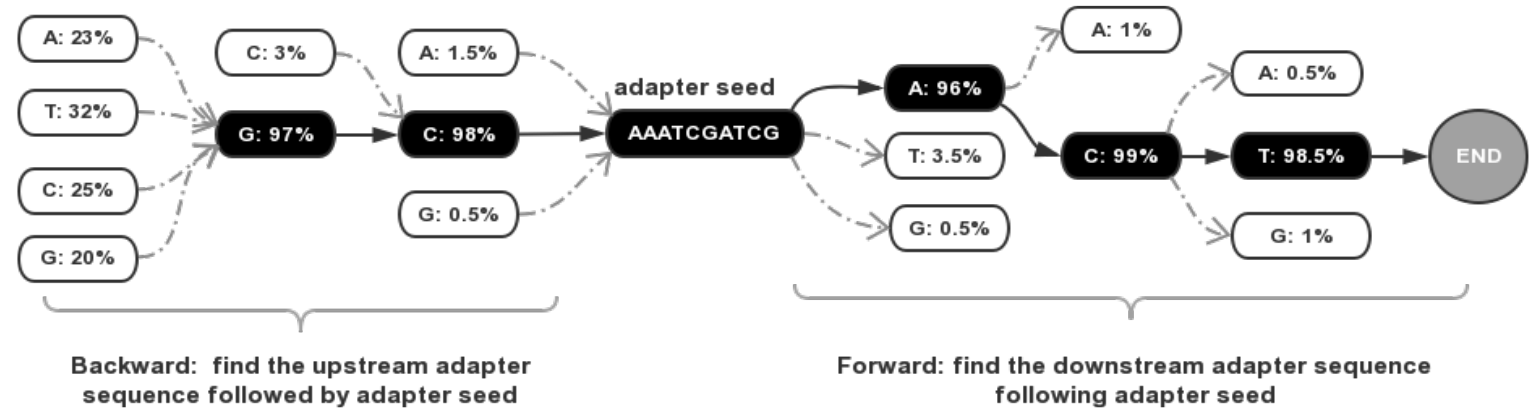
Automatic adapter trimming for PE data by overlap analysis

For each pair of reads, fastp computes their overlapping region and then computes the insert length. If insert length is shorter than the sequencing length, then adapter sequence will be present in the 3 end.

Automatic adapter sequence detection for SE data

Algorithm 1: adapter sequence detection

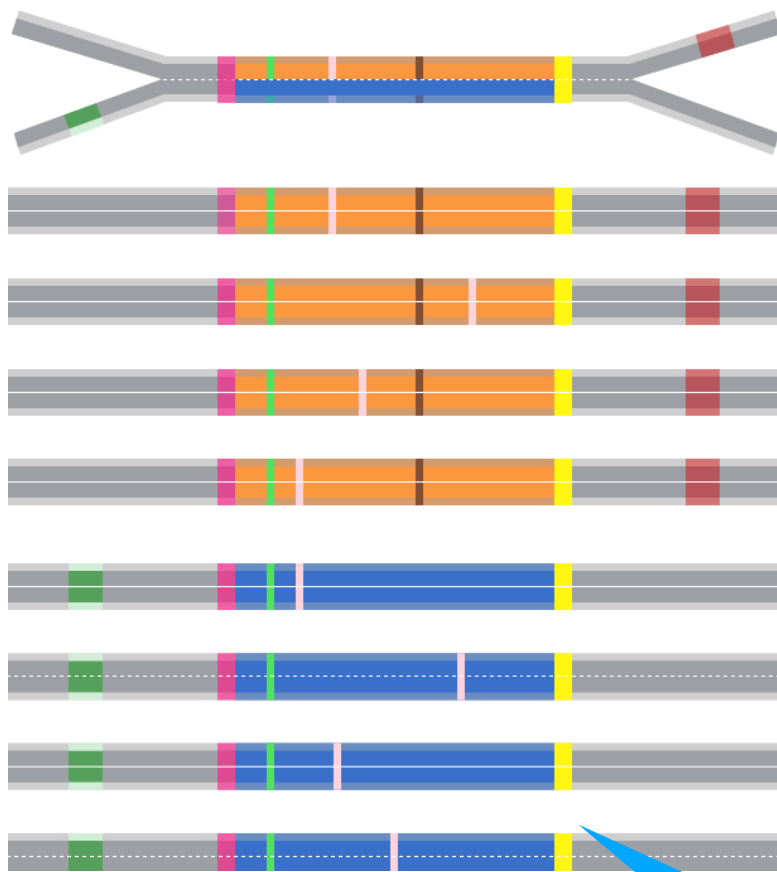
```
for seed in sorted_adapter_seeds:
    seqs_after_seed = get_seqs_after(seed)
    forward_tree = build_nucleotide_tree(seqs_after_seed)
    found = True
    node = forward_tree.root
    after_seed = ""
    while node.is_not_leaf():
        if node.has_dominant_child():
            node = node.dominant_child()
            after_seed = after_seed + node.base
        else:
            found = False
            break
    if found == False:
        continue
    else:
        seqs_before_seed = get_seqs_before(seed)
        backward_tree = build_nucleotide_tree(seqs_before_seed)
        node = backward_tree.root
        before_seed = ""
        while node.is_not_leaf():
            if node.has_dominant_child():
                node = node.dominant_child()
                before_seed = node.base + before_seed
            else:
                break
        adapter = before_seed + seed + after_seed
        break
```



Detect adapter sequence by constructing nucleotide tree

This algorithm tries to extend an adapter seed in the forward direction to check its validity since a valid adapter can always be extended to the read tails. And if this adapter seed is valid, a backward extension is applied to obtain the complete adapter sequence.

UMI preprocessing



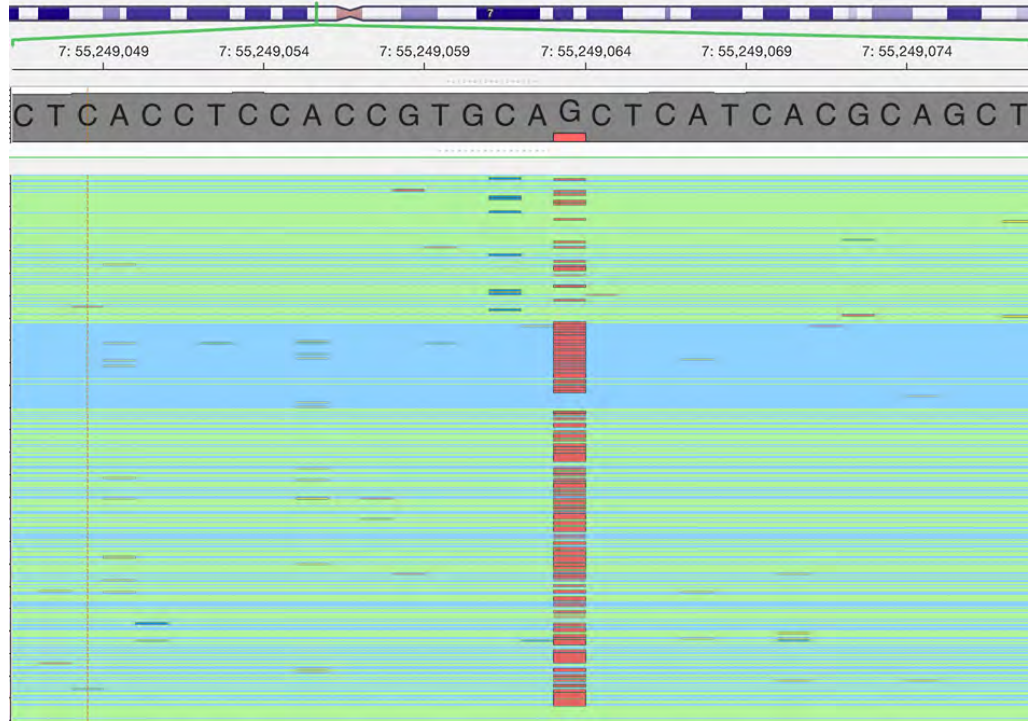
UMI sequencing

| Tools | Time (min) | Throughput (read/s) |
|-----------|------------|---------------------|
| fastp | 4.6 | 104302.9 |
| umis | 12.43 | 38599.6 |
| UMI-tools | 28.38 | 16906.4 |

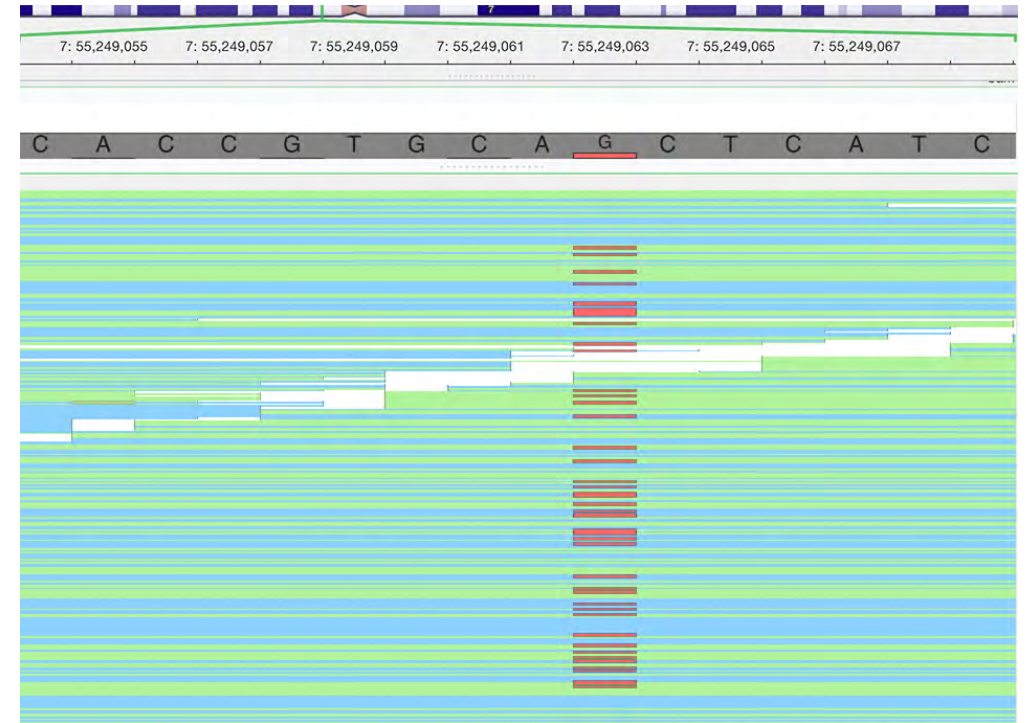
Handle UMI easily by fastp

fastp supports UMI in either a sample index or inserted DNA (or both). Compared to UMI-tools or umis, fastp runs approximately 3 times faster even when performing other tasks simultaneously (i.e., QC and filtering). Performance evaluation results are discussed in the next section.

fastp + gencore for UMI-based deduplication and error elimination



(a) the noisy data before processing

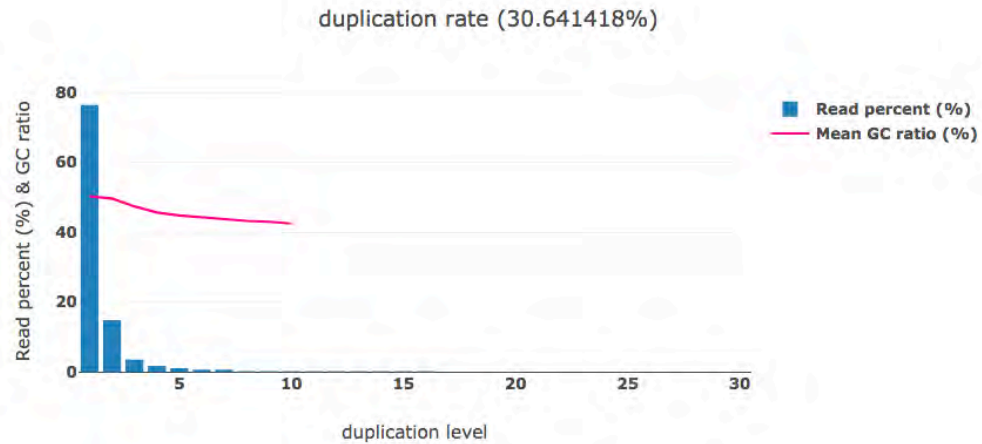


(b) the clean data after processing

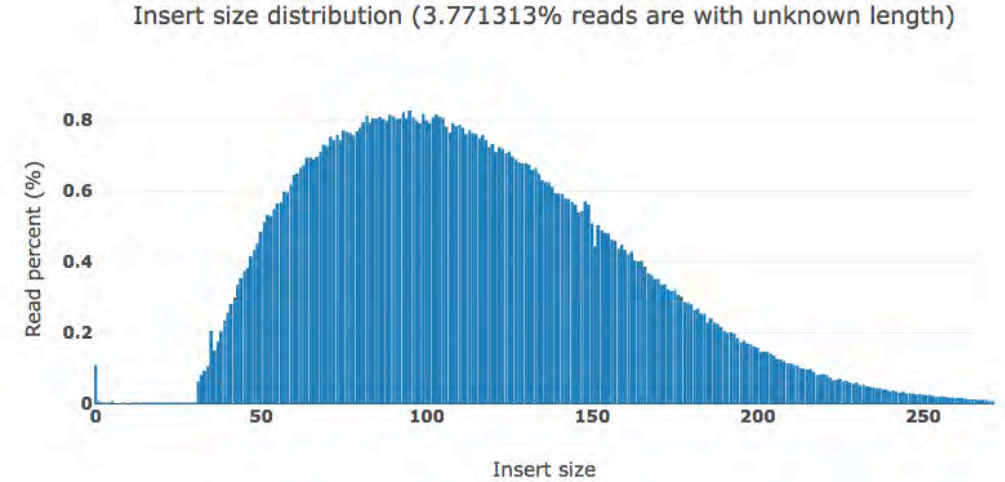
The effect of using fastp and gencore to reduce sequencing errors

gencore is a tool to generate consensus reads, which is also developed by me: <https://github.com/OpenGene/gencore>

Evaluate duplication level and insert-size distribution



(a) the duplication level and distribution



(b) the insert-size distribution

fastp automatically evaluates duplication rate and insert-size with almost zero overhead
fastp supports duplication level evaluation for both single-end and paired-end data. Different from FASTQC that uses a hash table to store the duplication keys, fastp stores them by a duplication array D and a counting array C to provide much faster access.

Detect the overrepresented sequences

[illegible]

fastp detects not only the overrepresented sequences, but also their distributions

Interactive full k-mer table

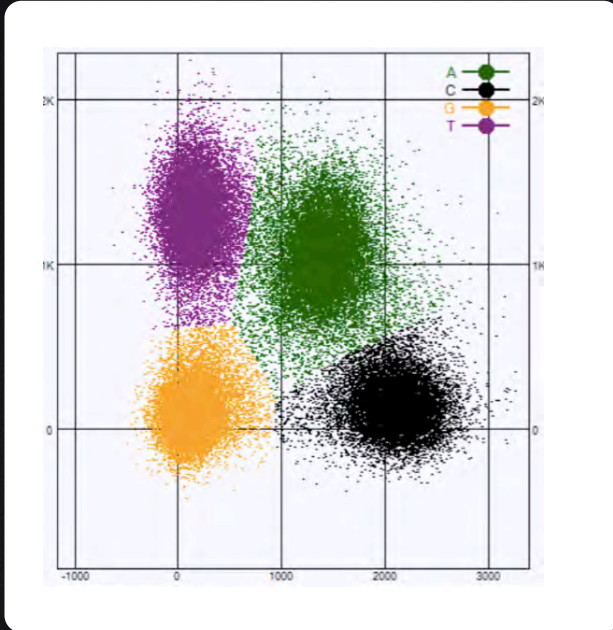
fastp computes the exactly occurrence table of 5-mer, and present it in a interactive HTML table.

Darker background means larger counts. The count will be shown on mouse over. By moving mouse over, it will show how frequent the 5-len sequence appears comparing to the mean value. From this table, you can find the abnormal sequencing artifacts, such like polyG problem.

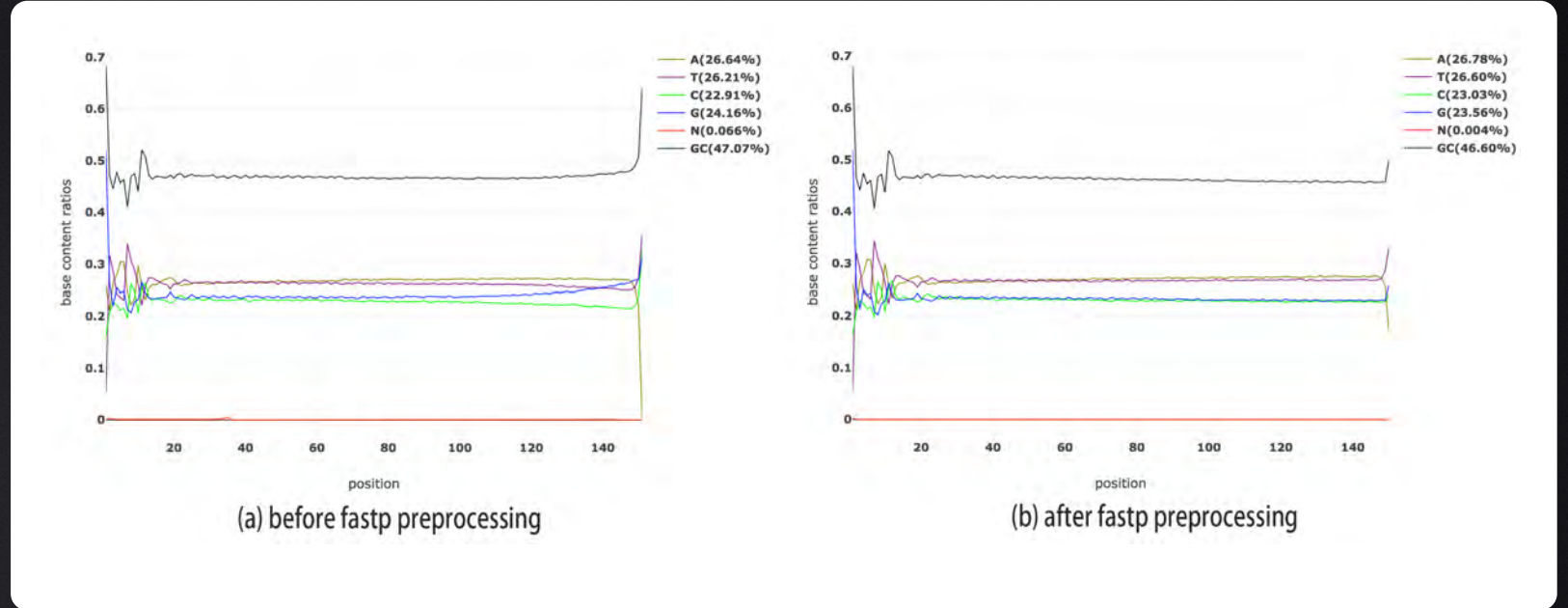
| | AA | AT | AC | AG | TA | TT | TC | TG | CA | CT | CC | CG | GA | GT | GC | GG |
|------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|---------|---------|---------|
| AAA | AAAA | AAAT | AAAAC | AAAAG | AAATA | AAATT | AAATC | AAATG | AAACA | AAACT | AAACC | AAACG | AAAGA | AAAGT | AAAGC | AAAGG |
| AAT | AATAA | AATAT | AATAC | AATAG | AATTA | AATTT | AATTC | AATTG | AATCA | AATCT | AATCC | AATCG | AATGA | AATGT | AATGC | AATGG |
| AAC | AACAA | AACAT | AACAC | AACAG | AACTA | AACCT | AACTC | AAC TG | AACCA | AACCT | AACCC | AACCG | AACGA | AACGT | AACGC | AACGG |
| AAG | AGAA | AGAT | AGAC | AGAG | AGTA | AGTT | AGTC | AGTG | AGCA | AGCT | AGCC | AGCG | AGGA | AGGT | AGGC | AGGG |
| ATA | ATAAA | ATAAT | ATAAC | ATAAG | ATATA | ATATT | ATATC | ATATG | ATACA | ATACT | ATACC | ATACG | ATAGA | ATAGT | ATAGC | ATAGG |
| ATT | ATTAA | ATTAT | ATTAC | ATTAG | ATTTA | ATTTT | ATTTT | ATTTG | ATTCA | ATTCT | ATTCC | ATT CG | ATTGA | ATTGT | ATTGC | ATTGG |
| ATC | ATCAA | ATCAT | ATCAC | ATCAG | ATCTA | ATCTT | ATCTC | ATCTG | ATCCA | ATCCT | ATCCC | ATCCG | ATCGA | ATCGT | ATCGC | ATCGG |
| ATG | ATGAA | ATGAT | ATGAC | ATGAG | ATGTA | ATGTT | ATGTC | ATGTG | ATGCA | ATGCT | ATGCC | ATGCG | ATGGA | ATGGT | ATGGC | ATGGG |
| ACA | ACAAA | ACAAT | ACAAC | ACAAG | ACATA | ACATT | ACATC | ACATG | ACACA | ACACT | ACACC | ACACG | ACAGA | ACAGT | ACAGC | ACAGG |
| ACT | ACTAA | ACTAT | ACTAC | ACTAG | ACTTA | ACTTT | ACTTC | ACTTG | ACTCA | ACTCT | ACTCC | ACTCG | ACTGA | ACTGT | ACTGC | ACTGG |
| ACC | ACCAA | ACCAT | ACCAC | ACCAG | ACCTA | ACCTT | ACCTC | ACCTG | ACCCA | ACCTT | ACCCC | ACCCG | ACCGA | ACCGT | ACCGC | ACCGG |
| ACG | ACGAA | ACGAT | ACGAC | ACGAG | ACGTA | ACGTT | ACGTC | ACGTG | ACGCA | ACGCT | ACGCC | ACGCG | ACGGA | ACGGT | ACGGC | ACGGG |
| AGA | AGAAA | AGAAT | AGAAC | AGAAG | AGATA | AGATT | AGATC | AGATG | AGACA | AGACT | AGACC | AGACG | AGAGA | AGAGT | AGAGC | AGAGG |
| AGT | AGTAA | AGTAT | AGTAC | AGTAG | AGTTA | AGTTT | AGTTC | AGTTG | AGTCA | AGTCT | AGTCC | AGTCG | AGTGA | AGTGT | AGTGC | AGTGG |
| AGC | AGCAA | AGCAT | AGCAC | AGCAG | AGCTA | AGCTT | AGCTC | AGCTG | AGCCA | AGCCT | AGCCC | AGCCG | AGCGA | AGCGT | AGCGC | AGCGG |
| AGG | AGGAA | AGGAT | AGGAC | AGGAG | AGGTA | AGGTT | AGGTC | AGGTG | AGGCA | AGGCT | AGGCC | AGGCG | AGGGA | AGGGT | AGGGC | AGGGG |
| TAA | TAAAA | TAAAT | TAAAC | TAAAG | TAATA | TAATT | TAATC | TAATG | TAACA | TAACT | TAACT | TAA CG | TAA GA | TAA GT | TAA GC | TAA GG |
| TAT | TATAA | TATAT | TATAC | TATAG | TATTA | TATTT | TATTC | TAT TG | TATCA | TATCT | TATCC | TATCG | TATGA | TATGT | TATGC | TATGG |
| TAC | TACAA | TACAT | TACAC | TACAG | TACTA | TACTT | TACTC | TACTG | TACCA | TACCT | TACCC | TACCG | TACGA | TACGT | TACGC | TACGG |
| TAG | TAGAA | TAGAT | TAGAC | TAGAG | TAGTA | TAGTT | TAGTC | TAGTG | TAGCA | TAGCT | TAGCC | TAGCG | TAGGA | TAGGT | TAGGC | TAGGG |
| TTA | TTAAA | TTAAT | TTAAC | TTAAG | TTATA | TTATT | TTATC | TTATG | TTACA | TTACT | TTACC | TTACG | TTAGA | TTAGT | TTAGC | TTAGG |
| TTT | TTTAA | TTTAT | TTTAC | TTTAG | TTTTA | TTTTT | TTTTT | TTTTG | TTTCA | TTTCT | TTTCC | TTTCG | TTTGA | TTTGT | TTTGC | TTTGG |
| TTC | TTCAA | TTCAT | TTCAC | TTCAG | TTCTA | TTCTT | TTCTC | TTCTG | TTCCA | TTCTT | TTCCC | TTCCG | TTCSA | TTCSGT | TTCCGC | TTCCGG |
| TTG | TTGAA | TTGAT | TTGAC | TTGAG | TTGTA | TTGTT | TTGTC | TTGTG | TTGCA | TTGCT | TTGCC | TTGCG | TTGGA | TTGGT | TTGGC | TTGGG |
| TCA | TCAAA | TCAAT | TCAAC | TCAAG | TCATA | TCATT | TCATC | TCATG | TCACA | TCACT | TCACC | TCACG | TCAGA | TCAGT | TCAGC | TCAGG |
| TCT | TCTAA | TCTAT | TCTAC | TCTAG | TCTTA | TCTTT | TCTTC | TCTTG | TCTCA | TCTCT | TCTCC | TCTCG | TCTGA | TCTGT | TCTGC | TCTGG |
| TCC | TCCAA | TCCAT | TCCAC | TCCAG | TCCTA | TCCTT | TCCTC | TCCTG | TCCCA | TCCCT | TCCCC | TCCCG | TCCGA | TCCGT | TCCGC | TCCGG |
| TCG | TCGAA | TCGAT | TCGAC | TCGAG | TCGTA | TCGTT | TCGTC | TCGTG | TCGCA | TCGCT | TCGCC | TCGCG | TCGGA | TCGGT | TCGGC | TCGGG |
| TGA | TGAAA | TGAAT | TGAAC | TGAAG | TGATA | TGATT | TGATC | TGATG | TGACA | TGACT | TGACC | TGACG | TGAGA | TGAGT | TGAGC | TGAGG |
| TGT | TGTAA | TGTAT | TGTAC | TGTAG | TGTTA | TGT TT | TGTTC | TGT TG | TGTCA | TGTCT | TGTCC | TGT CG | TGTGA | TGTGT | TGTGC | TGTGG |
| TGC | TGCAA | TGCAAT | TGCAAC | TGCAAG | TGCTA | TGCTT | TGCTC | TGCTG | TGCCA | TGCTT | TGCCC | TGCCG | TGCSA | TGCSGT | TGCCGC | TGCCGG |
| TGG | TGGAA | TGGAT | TGGAC | TGGAG | TGGTA | TGGTT | TGGTC | TGGTG | TGGCA | TGGCT | TGGCC | TGGCG | TGGGA | TGGGT | TGGGC | TGGGG |
| CAA | CAAAA | CAAAAT | CAAAAC | CAAAAG | CAATA | CAATT | CAATC | CAATG | CAACA | CAACT | CAACC | CAACG | CAAGA | CAAGT | CAAGC | CAAGG |
| CAT | CATAA | CATAT | CATAC | CATAG | CATTA | CATTT | CATTC | CAT TG | CATCA | CATCT | CATCC | CATCG | CATGA | CATGT | CATGC | CATGG |
| CAC | CACAA | CACAT | CACAC | CACAG | CAC TA | CAC TT | CAC TC | CAC TG | CACCA | CACCT | CACCC | CACCG | CACGA | CACGT | CACGC | CACGG |
| CAG | CAGAA | CAGAT | CAGAC | CAGAG | CAGTA | CAGTT | CAGTC | CAGTG | CAGCA | CAGCT | CAGCC | CAGCG | CAGGA | CAGGT | CAGGC | CAGGG |
| CTA | CTAAA | CTAAT | CTAAC | CTAAG | CTATA | CTATT | CTATC | CTATG | CTACA | CTACT | CTACC | CTACG | CTAGA | CTAGT | CTAGC | CTAGG |
| CTT | CTTAA | CTTAT | CTTAC | CTTAG | CTTTA | CTTTT | CTTTC | CTT TG | CTTCA | CTTCT | CTTCC | CTTCG | CTTGA | CTTGT | CTTGC | CTTGG |
| CTC | CTCAA | CTCAT | CTCAC | CTCAG | CTCTA | CTCTT | CTCTC | CTCTG | CTCCA | CTCCT | CTCCC | CTCCG | CTCSA | CTCSGT | CTCCGC | CTCCGG |
| CTG | CTGAA | CTGAT | CTGAC | CTGAG | CTGTA | CTGTT | CTGTC | CTGTG | CTGCA | CTGCT | CTGCC | CTGCG | CTGGA | CTGGT | CTGGC | CTGGG |
| CCA | CCAAA | CCAAAT | CCAAAC | CCAAAG | CCATA | CCATT | CCATC | CCATG | CCACA | CCACT | CCACC | CCACG | CCAGA | CCAGT | CCAGC | CCAGG |
| CCY | CCYAA | CCYAT | CCYAC | CCYAG | CCYTA | CCYTT | CCYTC | CCY TG | CCYCA | CCYCT | CCYCC | CCYCG | CCYGA | CCYGT | CCYGC | CCYGG |
| CCC | CCCAA | CCCAT | CCCAC | CCCAG | CCCTA | CCCTT | CCCTC | CCCTG | CCCCA | CCCCT | CCCCC | CCCCG | CCCSA | CCCSGT | CCCCGC | CCCCGG |
| CCG | CCGAA | CCGAT | CCGAC | CCGAG | CCGTA | CCGTT | CCGTC | CCGTG | CCGCA | CCGCT | CCGCC | CCGCG | CCGGA | CCGGT | CCGGC | CCGGG |
| CGA | CGAAA | CGAAT | CGAAC | CGAAG | CGATA | CGATT | CGATC | CGATG | CGACA | CGACT | CGACC | CGACG | CGAGA | CGAGT | CGAGC | CGAGG |
| CGT | CGTAA | CGTAT | CGTAC | CGTAG | CGTTA | CGTTT | CGTTC | CGT TG | CGTCA | CGTCT | CGTCC | CGTCG | CGTGA | CGTGT | CGTGC | CGTGG |
| CGC | CGCAA | CGCAT | CGCAC | CGCAG | CGCTA | CGCTT | CGCTC | CGCTG | CGCCA | CGCCT | CGCCC | CGCCG | CGCSA | CGCSGT | CGCCGC | CGCCGG |
| CGG | CGGAA | CGGAT | CGGAC | CGGAG | CGGTA | CGGTT | CGGTC | CGGTG | CGGCA | CGGCT | CGGCC | CGGCG | CGGGA | CGGGT | CGGGC | CGGGG |
| GAA | GA AAA | GA AAT | GA AAC | GA AAG | GA ATA | GA ATT | GA ATC | GA ATG | GA ACA | GA ACT | GA ACC | GA ACG | GA AGA | GA AGT | GA AGC | GA AGG |
| GAT | GATAA | GATAT | GATAC | GATAG | GATTA | GATTT | GATTC | GAT TG | GATCA | GATCT | GATCC | GATCG | GATGA | GATGT | GATGC | GATGG |
| GAC | GACAA | GACAT | GACAC | GACAG | GACTA | GACTT | GACTC | GACTG | GACCA | GACCT | GACCC | GACCG | GACGA | GACGT | GACGC | GACGG |
| GAG | GAGAA | GAGAT | GAGAC | GAGAG | GAGTA | GAGTT | GAGTC | GAGTG | GAGCA | GAGCT | GAGCC | GAGCG | GAGGA | GAGGT | GAGGC | GAGGG |
| GTA | GTA AA | GTA AT | GTA AC | GTA AG | GTA TA | GTA TT | GTA TC | GTA TG | GTA CA | GTA CT | GTA CC | GTA CG | GTA GA | GTA GT | GTA GC | GTA GG |
| GT T | GT TAA | GT TAT | GT TAC | GT TAG | GT TTA | GT TTT | GT TTC | GT TTG | GT TCA | GT TCT | GT TCC | GT TCG | GT TGA | GT TGT | GT TGC | GT TGG |
| GT C | GT CAA | GT CAT | GT CAC | GT CAG | GT CTA | GT CTT | GT CTC | GT CTG | GT CCA | GT CCT | GT CCC | GT CCG | GT CSA | GT CSGT | GT CCGC | GT CCGG |
| GT G | GT GAA | GT GAT | GT GAC | GT GAG | GT GTA | GT GTT | GT GTC | GT GTG | GT GCA | GT GCT | GT GCC | GT GCG | GT GGA | GT GGT | GT GGC | GT GGG |
| GCA | GCA AA | GCA AT | GCA AC | GCA AG | GCA TA | GCA TT | GCA TC | GCA TG | GCA CA | GCA CT | GCA CC | GCA CG | GCA GA | GCA GT | GCA GC | GCA GG |
| GCT | GCT AA | GCT AT | GCT AC | GCT AG | GCT TA | GCT TT | GCT TC | GCT TG | GCT CA | GCT CT | GCT CC | GCT CG | GCT GA | GCT GT | GCT GC | GCT GG |
| GCC | GCC AA | GCC AT | GCC AC | GCC AG | GCC TA | GCC TT | GCC TC | GCC TG | GCC CA | GCC CT | GCC CC | GCC CG | GCC GA | GCC GT | GCC GC | GCC GG |
| CGG | CGG AA | CGG AT | CGG AC | CGG AG | CGG TA | CGG TT | CGG TC | CGG TG | CGG CA | CGG CT | CGG CC | CGG CG | CGG GA | CGG GT | CGG GC | CGG GG |
| GGA | GG AAA | GG AAT | GG AAC | GG AAG | GG ATA | GG ATT | GG ATC | GG ATG | GG ACA | GG ACT | GG ACC | GG ACG | GG AGA | GG AGT | GG AGC | GG AGG |
| GGT | GGT AA | GGT AT | GGT AC | GGT AG | GGT TA | GGT TT | GGT TC | GGT TG | GGT CA | GGT CT | GGT CC | GGT CG | GGT GA | GGT GT | GGT GC | GGT GG |
| GGC | GGC AA | GGC AT | GGC AC | GGC AG | GGC TA | GGC TT | GGC TC | GGC TG | GGC CA | GGC CT | GGC CC | GGC CG | GGC GA | GGC GT | GGC GC | GGC GG |
| GGG | GGG AA | GGG AT | GGG AC | GGG AG | GGG TA | GGG TT | GGG TC | GGG TG | GGG CA | GGG CT | GGG CC | GGG CG | GGG GA | GGG GT | GGG GC | GGG GG |

The dark GGGGG indicates polyG

How polyG happens, and how fastp addresses this issue



2-color system causes polyG



fastp addresses the polyG issue

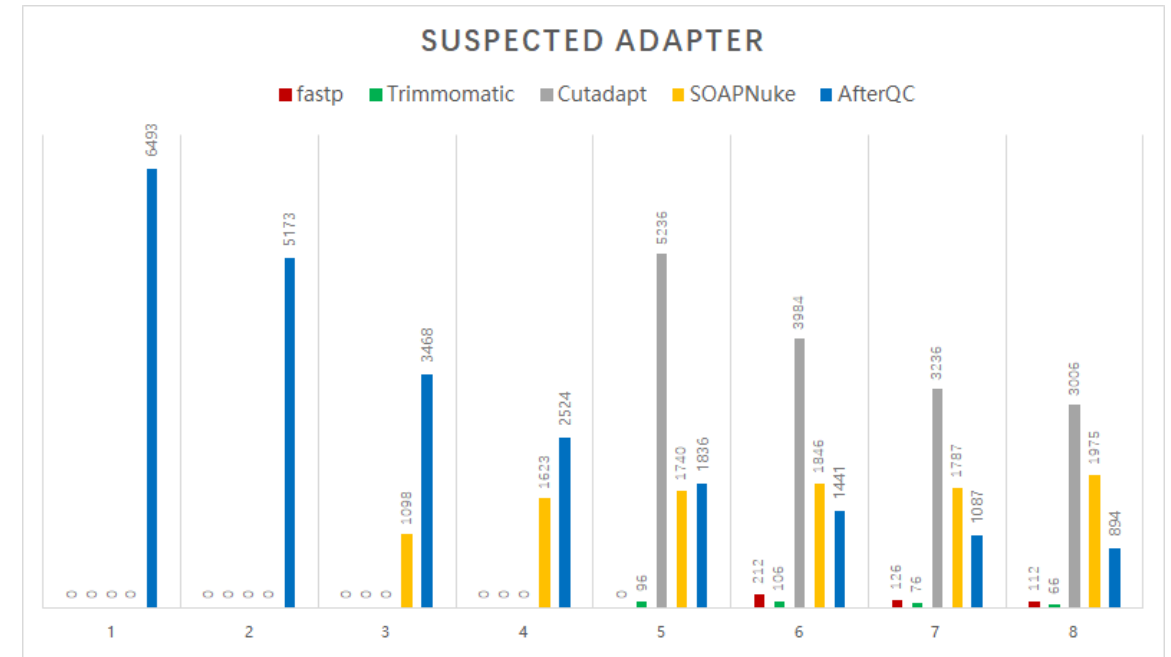
fastp addresses the polyG issue by trimming polyG tails

fastp also implements polyX tail trimming, where X means any base of A/T/C/G. This function can be used to trim the low-complexity consecutive bases in 3' end. PolyX tail trimming and polyG tail trimming can be enabled together.

Evaluation of filtering performance for downstream analysis

| NS_PE150 | Total map base (M) | Mis-matched base (M) | Total map read (M) | Clip read (M) | Single-read map |
|-------------|--------------------|----------------------|--------------------|---------------|-----------------|
| Raw data | 3390 | 19.8 | 22.4 | 6.16 | 46025 |
| fastp | 3102 | 10.6 | 21.2 | 0.30 | 271 |
| AfterQC | 3053 | 12.3 | 21.2 | 0.64 | 34099 |
| SOAPnuke | 2981 | 18.3 | 19.7 | 3.46 | 36888 |
| Trimmomatic | 3111 | 12.8 | 22.1 | 0.75 | 229111 |
| Cutadapt | 3287 | 19.8 | 22.4 | 1.05 | 46195 |

(a) Mismatches, clips, and single-read maps of the data filtered with different tools.



(b) Result of adapter trimming performance evaluation.

fastp produces cleaner data for downstream analysis

The standard dataset NA12878 (SRR952827) was selected for benchmarking fastp, Trimmomatic, Cutadapt and SOAPNuke, for their performance on reducing false positive variants.

10T bases generated per day, how to



repaq

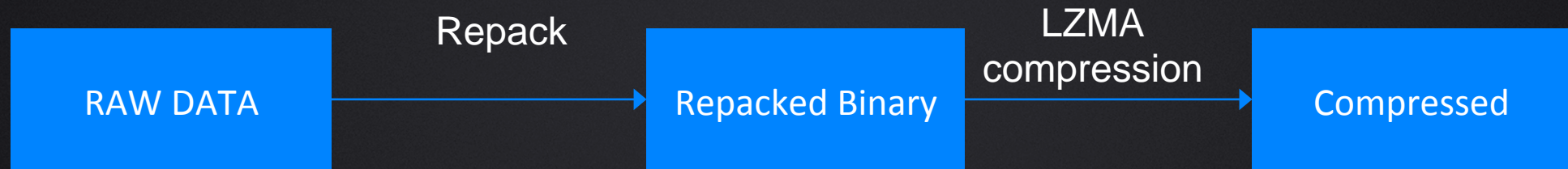
A pilot project for lossless compression of FASTQ data



<https://github.com/OpenGene/repaq>

★ 40

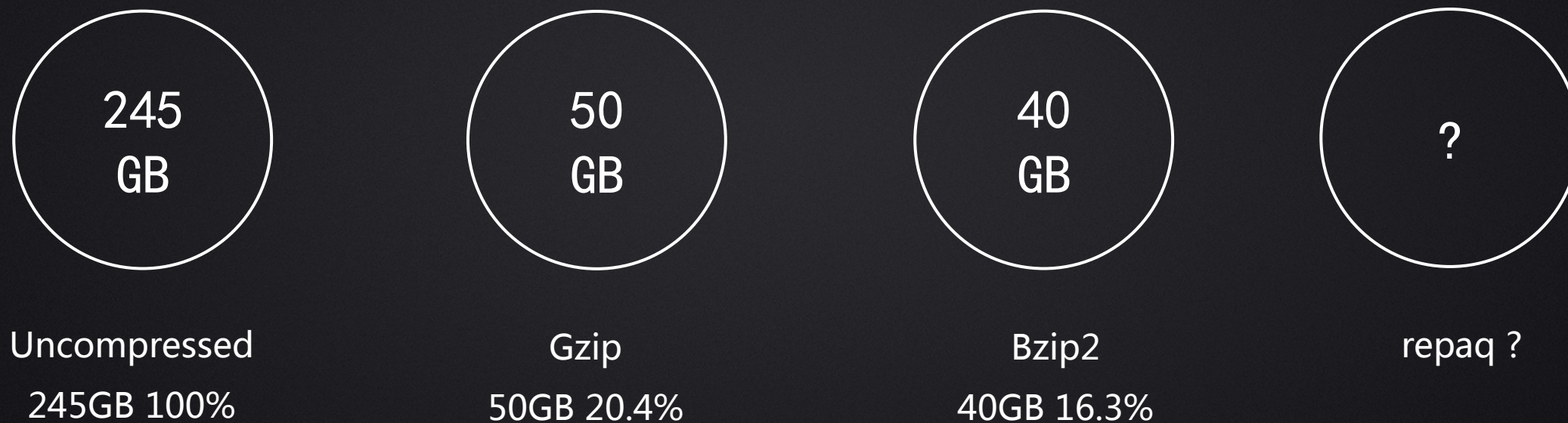
2-stage compression



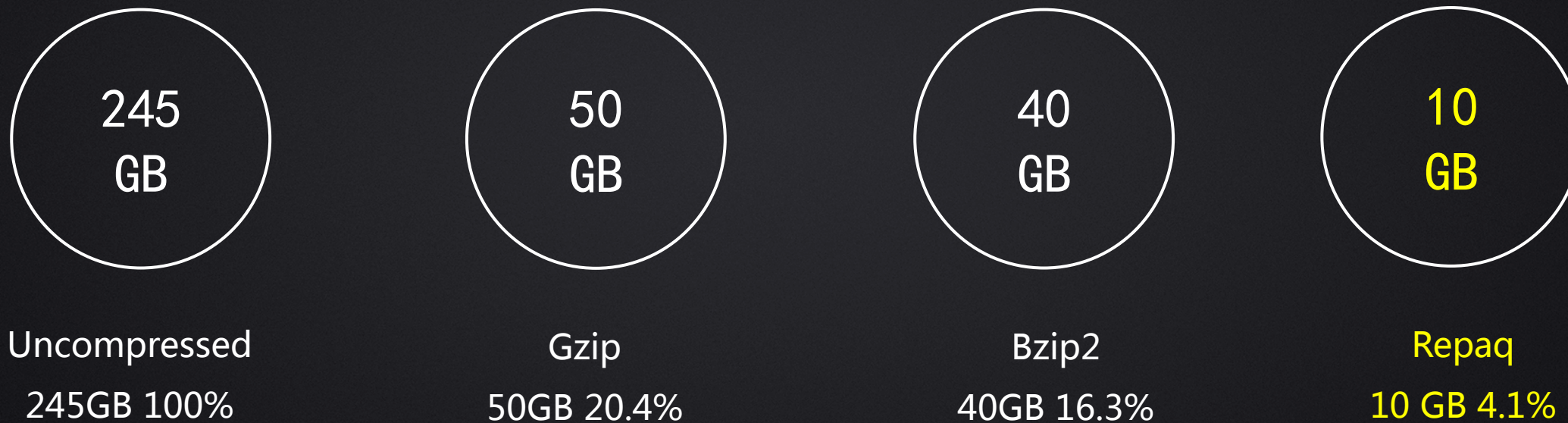
Strategies to repack the original FASTQ

1. Encode and decode by chunks
2. Process the metainfo, sequence and quality separately
3. Extract the common info and ordered info from metainfo
4. Find the overlap of a pair of reads and encode it only once
5. Find the connection of N bases with corresponding quality bin
6. Find the major quality bin, and only encode the others
7. Don't use any entropy encoding to preserve the compressibility for LZMA

FASTQ file of 100G bases from NovaSeq



FASTQ file of 100G bases from NovaSeq





Welcome to Shenzhen
Thank you for your attention!

