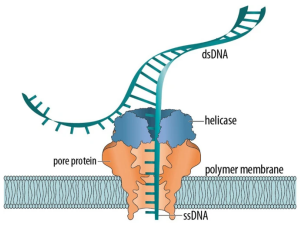


ONT Sequence enrichment using selective sequencing

Caroline Van Damme and Stéphane Plaisance [VIB - Nucleomics Core, nucleomics@vib.be]

July 6th, 2021 - version 1.0

Contents

last edits: Tue Jul 13, 2021

Aim

In this short project, we evaluate the efficiency and throughput of selective sequencing on GridION. The method is very easy to implement as it requires only to ligate genomic DNA and provide a list of loci (BED) or a multifasta target file to be used. For this evaluation we chose the backer yeast for which we had gDNA available at the core (kind gift of Brigida Gallone, VIB Microbial and Plant Genetics).

The yeast reference genome version **R64-1-1** and gene annotations were obtained from ensembl (https://fungi.ensembl.org/Saccharomyces_cerevisiae/Info/Index) and prepared for Read Until enrichment, read mapping, and visualization (IGV).

- fasta file: Saccharomyces_cerevisiae.R64-1-1.dna.toplevel.fa.gz
- GTF file: Saccharomyces_cerevisiae.R64-1-1.51.gtf.gz

Run on GridION

Two read Until enrichment multifasta files were build by splitting the yeast chromosomes in odd and even sets.

- Saccharomyces_cerevisiae.R64-1-1.odd-chr.fa
- Saccharomyces_cerevisiae.R64-1-1.odd-chr.fa

More info about adaptive sequencing can be found at **adaptive-sampling-ADS_S1016_v1_revB_12Nov2020-any.pdf** (Nanopore 2020)

chrom	size
chrI	230218
chrII	813184
chrIII	316620
chrIV	1531933
chrIX	439888
chrV	576874
chrVI	270161
chrVII	1090940
chrVIII	562643
chrX	745751
chrXI	666816
chrXII	1078177
chrXIII	924431
chrXIV	784333
chrXV	1091291
chrXVI	948066
chrM	85779

The first library (**RBK-004** kit) consisting in a pool of two barcoded samples (bc1 and bc2, same gDNA used for both) was loaded on a GridION flow-cell as detailed in the ONT protocol and run for 24h under MinKnow while enriching against **Saccharomyces_cerevisiae.R64-1-1.even-chr.fa**.

The run was terminated and the flow-cell was striped and washed using the ONT **EXP-WSH003** kit. The cleaned flow-cell was then loaded with a pool of next two separate samples (bc3 and bc4, same gDNA used for both) and run for another 24h using the second Read Until target **Saccharomyces_cerevisiae.R64-1-1.odd-chr.fa**.

NOTE: The second run was not as productive as the first one due to flow-cell exhaustion.

Run statistics

Both runs were analyzed using high-quality basecalling and demultiplexed.

Run#1: even chromosome

```
Run Info
Host Name GXB03198 (localhost)
Experiment Name 3831
Sample ID test_SC
Run ID a5423938-5a67-4fdd-a506-921a0284dcb2
Flow Cell Id FAQ01722
Start Time June 15, 11:52
Run Length 1d 0h 4m
Run Summary
Reads Generated 916.01 K
Passed Bases 4.76 Gb
Failed Bases 111.15 Mb
Estimated Bases 4.69 Gb
Read Until
reference_files=
["/data/adaptive_references/Saccharomyces_cerevisiae.R64-1-1.evenchr.
fa"],filter_type=enrich,first_channel=1,last_channel=512
```

Run#2: odd chromosome

```
Run Info
Host Name GXB03198 (localhost)
Experiment Name 3831
Sample ID test_SC
Run ID 77f7c337-0381-4427-8555-50144983800e
Flow Cell Id FAQ01722
Start Time June 17, 09:57
Run Length 2d 0h 3m
Run Summary
Reads Generated 466.88 K
Passed Bases 981.29 Mb
Failed Bases 25.45 Mb
Estimated Bases 1.14 Gb
Read Until
reference_files=
["/data/adaptive_references/Saccharomyces_cerevisiae.R64-1-1.oddchr.
fa"],filter_type=enrich,first_channel=1,last_channel=512
```

Adaptive sequencing classification of the reads

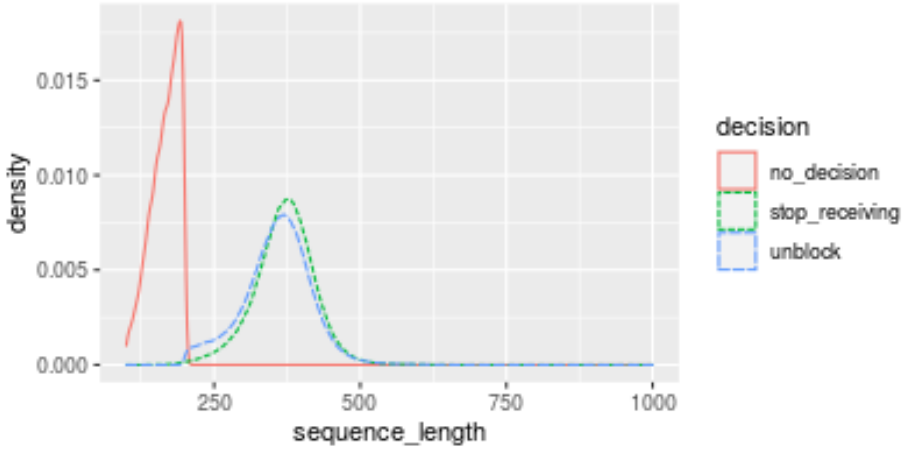
Principle: During the first few 100 base sequencing from each pore, the nascent read is basecalled and aligned to the capture reference and a decision is taken whether to keep sequencing it or to eject it. As soon as this decision is taken, the result and current read length are written to a file (*read_until_<FCID>_<RunID>.csv*) used below to create a density plot for each run.

Note The lengths reported in this file are the lengths at the instant of the decision and not the final read length (except for no_decision reads).

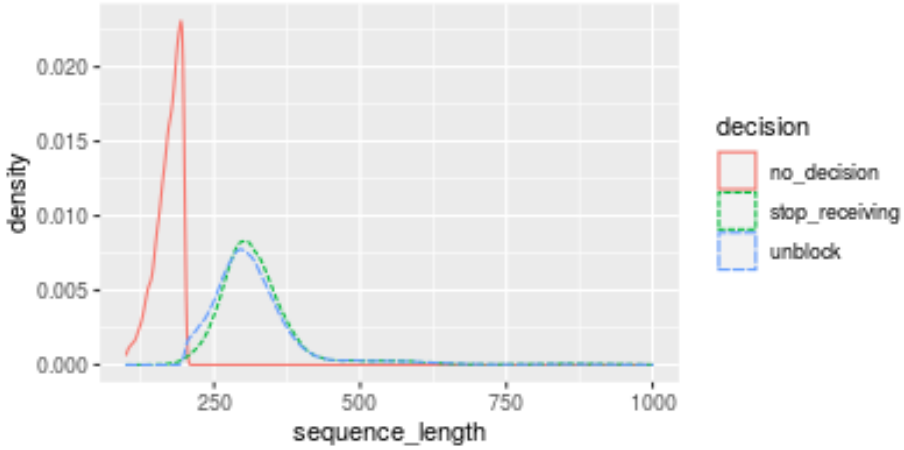
The possible decisions labels are:

- "**unblock**" are the rejected reads
- "**stop_receiving**" are the accepted reads
- "**no_decision**" were fully sequenced before a decision could be made.

Read distribution for Run1



Read distribution for Run2



As expected, the *no-decision* reads are short as they ended before the decision point (decision made after ~200bps) while *unblock* and *stop_receiving* read class largely overlap and show a larger size distribution (that of the library). *unblock* reads are of intermediate length due to the time needed to physically reject the strand from the pore (generally sequenced to less than 500bps).

Final read size distribution

The final basecalled **Fastq_pass** data was parsed together with the *read_until_<FCID>_<RunID>.csv* classification results to get final size information from the *stop_receiving* reads and create a new fastq file with only these reads for remapping.

- "**absent**" are reads absent from the above *read_until_<FCID>_<RunID>.csv* file and for which ONT should give us feedback in the future. They were not used further in this analysis.

`Classification results in read numbers for Run1`

classified	count	freq
stop_receiving	500820	61.44
unblock	289913	35.57
absent	24033	2.95
no_decision	332	0.04

`Classification results in Mbases for Run1`

classified	megabases	freq
stop_receiving	4602.76	96.61
unblock	141.80	2.98
absent	19.44	0.41
no_decision	0.08	0.00

`Classification results in read numbers for Run2`

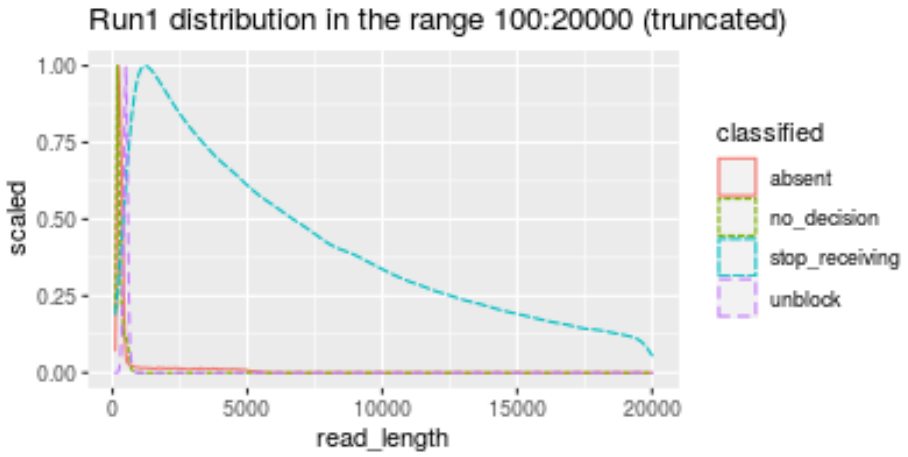
classified	count	freq
unblock	283932	67.47
stop_receiving	97598	23.19
absent	37975	9.02
no_decision	1341	0.32

`Classification results in Mbases for Run2`

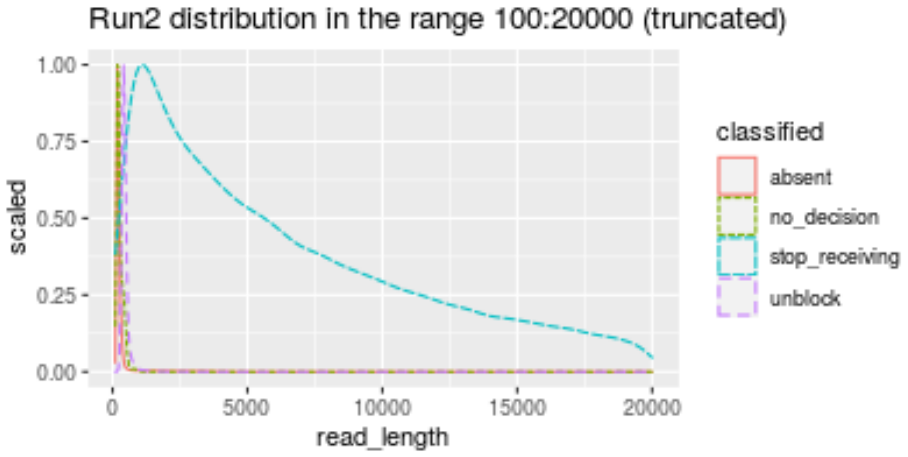
classified	megabases	freq
stop_receiving	839.13	85.51
unblock	129.41	13.19
absent	12.39	1.26
no_decision	0.36	0.04

Read size distribution truncated to the range 100-20000bp

Run1

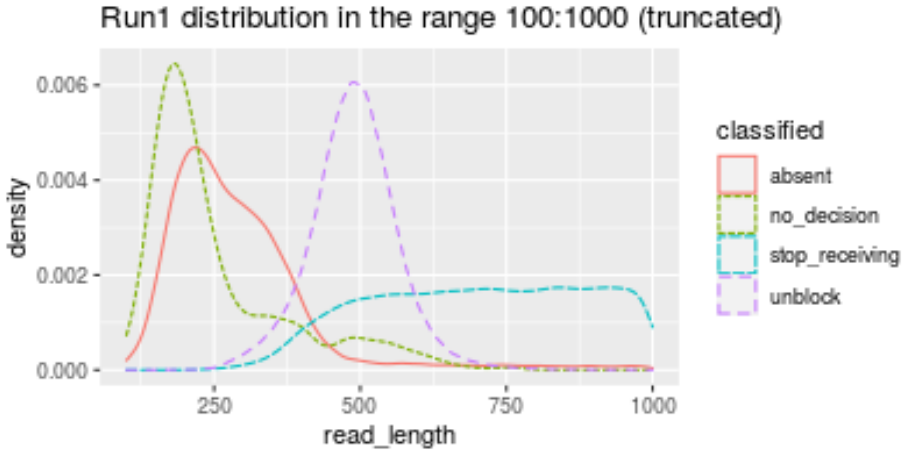


Run2

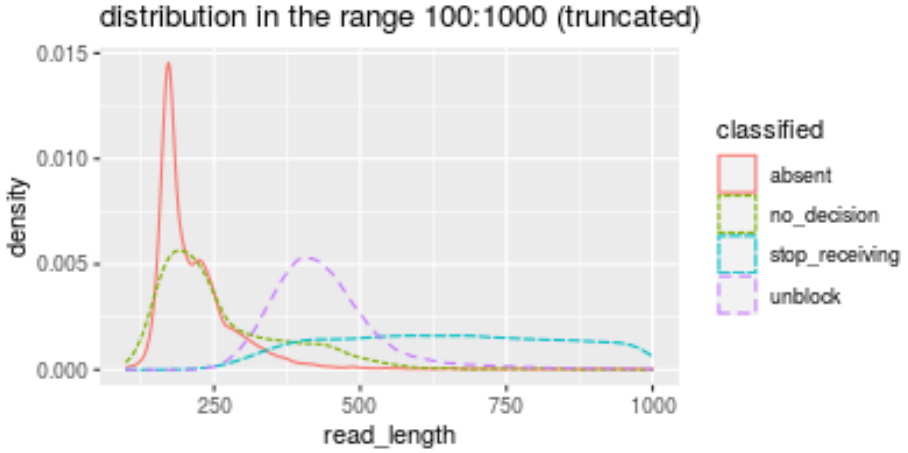


Read size distribution truncated to the range 100-1000bps

Run1



Run2



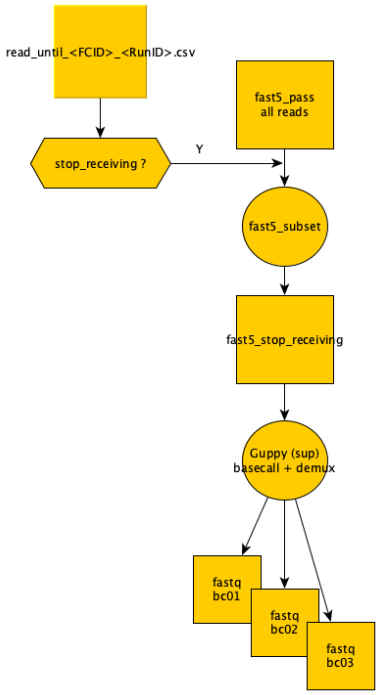
The data shows that a large majority of the reads were rejected (unblock: 35% and 67%) as not aligning to the target references. The selectively enriched class *stop_receiving* (61% and 23% of all reads), shows reads longer than 20kb (truncated to 20kb in the plots to exclude outliers) and represents a large fraction of the total nucleotide data (96% and 85%) while other classes contain reads shorter than ~750bps (see zoom in 1kb range) and represent together a very small portion of the full data (expressed in Mb rrather than read counts, see tables above).

Fast5 data re-basecalling and barcode demultiplexing

The two runs were originally done using high accuracy basecalling (*hac*) while the new version of MinKnow is capable of better basecalling performance thanks to the new **super-accurate** algorithm (*sup*).

We decided to repeat the basecalling and demultiplexing of the data using the latest Guppy version. In order to do so we took the subset of the Fast5 raw data corresponding to the **stop_receiving** reads above and fed it to Guppy for de-novo basecalling and demultiplexing (using MinKnow on the GridION).

The subsetting of the Fast5 data was done using the ONT *Fast5_subset* utility (part of **ont_Fast5_api**) with the list of read IDs classified as **stop_receiving** in the two original *read_until_<FCID>_<RunID>.csv* files and the full Fast5 datasets as shown in the graph below.



The obtained Fast5 subsets were then fed to Guppy v5.0.11 using MinKnow to obtain barcode-split read folders for each run (run done with *sup* basecalling and barcoding without clipping with a barcode cutoff of 40, all other parameters set to default).

Theoretically, Run1 should return only *barcode01* and *barcode02* reads while Run2 should report *barcode03* and *barcode04* reads. The actual distribution is shown next and only the four expected barcodes are further used down this report.

Run1

barcode	reads	read%
barcode01	232671	49%
barcode02	228399	48%
barcode03	83	0%
barcode04	96	0%
barcode05	79	0%
barcode06	93	0%
barcode07	76	0%
barcode08	21	0%
barcode09	74	0%
barcode10	46	0%
barcode11	15	0%
barcode12	13	0%
unclassified	12877	3%

Run2

barcode	reads	read%
barcode01	46	0%
barcode02	70	0%
barcode03	30866	34%
barcode04	55043	61%
barcode05	21	0%
barcode06	10	0%
barcode07	14	0%
barcode08	10	0%
barcode09	12	0%
barcode10	9	0%
barcode11	11	0%
barcode12	6	0%
unclassified	3770	4%

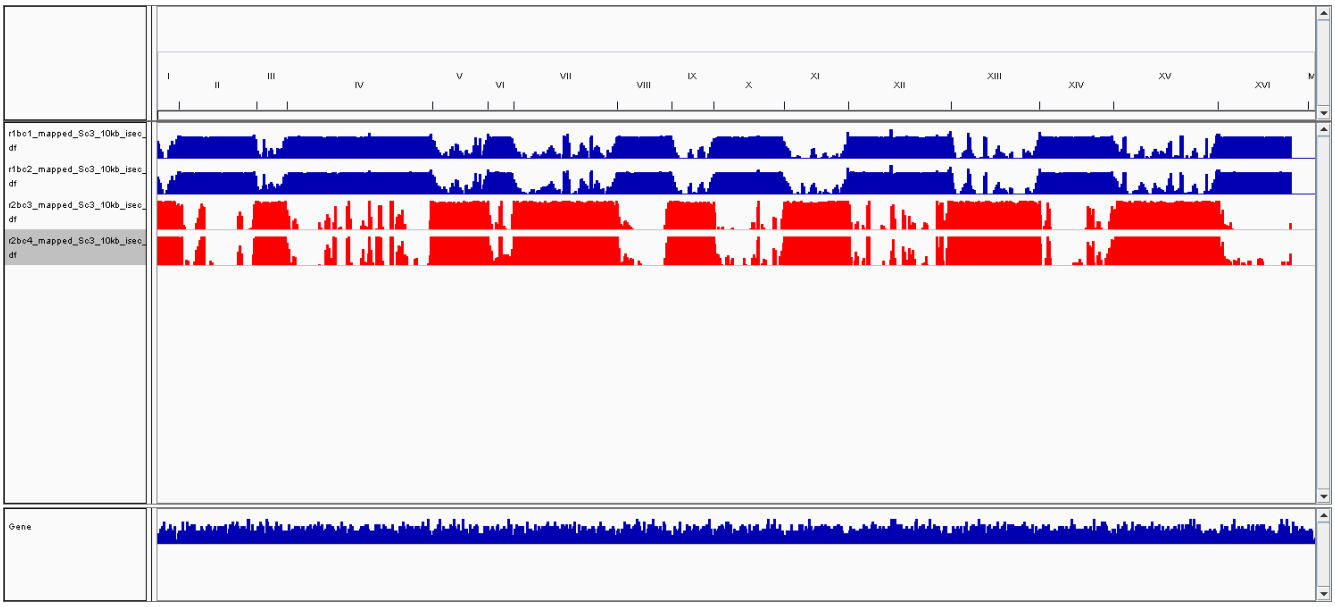
Less than 5% of the reads were returned in unexpected barcode bins, leaving a large majority of the input data for further analysis.

Mapping of barcode read samples to the yeast reference genome

The barcode read sets 1..to..4 were mapped to R64-1-1 using minimap2 and standard settings.

The read mappings were converted to genome coverage (TDF) tracks using Bedtools and igvtools.

The coverage tracks were loaded for display in IGV and used to produce the next picture.



As seen, the coverage of the first two barcode samples (bc1 and bc2) is clearly higher in *even* chromosomes (blue histograms) with a few exceptions in probable homologous regions present on *odd* chromosomes. Similarly, the coverage of the other two barcodes (bc3 and bc4) coincides with *odd* chromosomes (red histograms). The log scales were adapted to compensate for the large difference in coverage between the two runs and y-axis are not identical.

Conclusion

This validation experiment using a yeast gDNA sample and barcoding confirms the efficiency of the Read_Until method and makes it amendable to sequence enrichment and multiplexing for genomes of the size of the backer yeast.

The only limitation of this pipeline is the live-span of the Flow-cell which is seriously affected by the frequent read rejection and leading to approximately 25% throughput as compared to a conventional run.

The obtained read pileup reached several 100's in depth which is perfectly compatible with variant analysis even using a more error prone platforms like ONT.

References

Nanopore, Oxford. 2020. "Designing and running an experiment with adaptive sampling." https://community.nanoporetech.com/info_sheets/adaptive-sampling/v/ads_s1016_v1_revb_12nov2020/designing-and-running-an-experiment-with-adaptive-sampling.