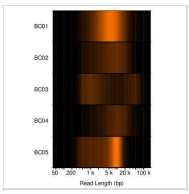


Version 8

Oct 19, 2022

Demultiplexing Nanopore reads with LAST V.8

In 2 collections

David A Eccles[1]

[1]Malaghan Institute of Medical Research (NZ)

1 *Works for me* Share

dx.doi.org/10.17504/protocols.io.14egnxw4zl5d/v8

David A Eccles
Malaghan Institute of Medical Research (NZ)

ABSTRACT

This protocol is for a semi-manual method for read demultiplexing, as used after my presentation *Sequencing DNA with Linux Cores and Nanopores* to work out the number of reads captured by different barcodes.

This approach has now been bundled up into a single Perl script, *fastq-dental.pl* which can be found here.

Input: reads as a FASTQ file, barcode sequences as a FASTA file
Output: reads split into single FASTQ files per target [barcode]

Note: barcode / adapter sequences are not trimmed by this protocol. This allows for subsequent adapter-associated downstream processing (e.g. strand correction for cDNA reads).

DOI

dx.doi.org/10.17504/protocols.io.14egnxw4zl5d/v8

EXTERNAL LINK

https://doi.org/10.5281/zenodo.2535894

PROTOCOL CITATION

David A Eccles 2022. Demultiplexing Nanopore reads with LAST. **protocols.io** https://dx.doi.org/10.17504/protocols.io.14egnxw4zl5d/v8 Version created by David A Eccles

MANUSCRIPT CITATION please remember to cite the following publication along with this protocol

COLLECTIONS

Nanopore Data Analysis

Nanopore Data Analysis

KEYWORDS

demultiplexing, nanopore, high-throughput sequencing

CREATED

Oct 19, 2022

LAST MODIFIED

Oct 19, 2022

PROTOCOL INTEGER ID

71517

PARENT PROTOCOLS

Part of collection

Nanopore Data Analysis

Nanopore Data Analysis

Perl Script

1 *Note: This approach has now been bundled up into a single Perl script, fastq-dental.pl which can be found here. The steps below demonstrate how to carry out similar operations manually using the command line.*

Here is an example for running the fastq-dental script:

```
./fastq-dental.pl -barcode barcode_full_PBK004.fa -mat bc.mat
reads_all.fastq.gz
```

Example output count information, demonstrating that reads are primarily mapped to BC06 or BC07 (i.e. the adapters that were added during sample preparation):

```
      9 BC01
     39 BC02
      1 BC02_BC07
     16 BC03
     38 BC04
      2 BC04_BC06
      1 BC04_BC07
      1 BC04_BC09
    643 BC05
      2 BC05_BC07
 493175 BC06
```


Citation: David A Eccles Demultiplexing Nanopore reads with LAST https://dx.doi.org/10.17504/protocols.io.14egnxw4zl5d/v8

```
      115 BC06_BC06
        2 BC06_BC07
   663161 BC07
        1 BC07_BC05
      117 BC07_BC07
        1 BC07_BC09
      106 BC08
        1 BC08_BC06
       75 BC09
       72 BC10
        1 BC10_BC07
       41 BC11
       44 BC12
        1 BC12_BC06
    13535 BCchim
   237796 BCnoadapt
      152 BCnone
       90 RB12A
```

Combine read sequences

2 Combine all input reads into a single file

```
pv ../called_all/*.fastq | gzip > reads_all.fastq.gz
```

Note: I'm using the pipe viewer command *pv* to produce a progress indicator while the command is running. If this command is not available, it can be replaced with *cat* with no change in function (apart from not showing progess).

Generating Barcode Index

3 Prepare a FASTA file containing barcode sequences, ideally *including* adapters (see attached FASTA file).

Using adapters reduces both the false negative rate, and false positive as it relates to barcode sequences being accidentally mapped to input non-adapter sequences, with some increase in false positive mapping for non-library barcode sequences. I have found that the new LAST model, *RY4*, together with *lastal --split*, substantially reduce these non-library barcode hits to similar or better levels than a mapping of the unique component of the barcode sequences alone.

[use the adapter-excluded *barcode_base_96.fa* if other sequences are not appropriate]

barcode_full_PBK004.fa **barcode_full_RBK004.fa** **barcode_full_RBK110.96.fa** **barcode_base_96.fa**

Create a shell variable containing this file name:

```
barcodeFile="barcode_full_PBK004.fa"
```


Citation: David A Eccles Demultiplexing Nanopore reads with LAST https://dx.doi.org/10.17504/protocols.io.14egnxw4zl5d/v8

4 Newer versions of LAST (v1409+) include a new seeding scheme, '-uRY4' [and other related RYX schemes], which improves mapping accuracy and reduces polyA matches; low-complexity regions are also converted to lower case. This will generate seven additional files of the form <index name>.XXX:

```
lastdb -uRY4 -R01 ${barcodeFile} ${barcodeFile}
```

5 Prepare a substitution matrix for barcode mapping. The default substitution matrix is swayed too much by INDELs in the barcode sequences, so here's one that I've developed using last-train using super-accuracy Nanopore reads called using Guppy v5.1.15:

```
#last -Q 1
#last -t4.22004
#last -a 16
#last -A 15
#last -b 4
#last -B 5
#last -S 1
# score matrix (query letters = columns, reference letters = rows):
       A      C      G      T
A      6    -33    -28    -34
C    -33      6    -34    -34
G    -14    -33      6    -34
T    -35    -34    -34      5
```

bc.mat

I generated this matrix from the following command:

```
last-train -Q 1 -P 10 ${barcodeFile} reads_all.fastq.gz > bc.mat
```

This quality-adjusted matrix has a small penalty for opening gaps (i.e. insertions and deletions), and a smaller penalty for inserting them. Insertions and deletions are considered to be equally likely in the barcode region. It also has a moderate penalty for A/G transition variants, with other substitution penalties similar.

Mapping Reads to Barcodes

6 Use LAST in *split* mode, using the pre-defined substitution matrix to map the reads. In this example, it is distributed over 10 processing threads (-P 10). Here *maf-convert* is used to convert to a single line per match, *cut* retains only the barcode and read IDs, and *uniq* is used to make sure that multiple same barcodes per read (e.g. for reverse / complement barcodes at each end) will not produce duplicates:

```
lastal --split -p bc.mat -P 10 ${barcodeFile} <(pv reads_all.fastq.gz) | \
  maf-convert -n tab | cut -f 2,7 | sort | uniq | \
```


Citation: David A Eccles Demultiplexing Nanopore reads with LAST https://dx.doi.org/10.17504/protocols.io.14egnxw4zl5d/v8

```
  gzip > barcode_assignments.txt.gz
```

Stringency can be altered by adjusting the query letters per random alignment setting (-D <value>, 1e6 by default). Lowering this number will produce more matches, at the expense of more false positive matches:

```
lastal --split -D 1e5 -p bc.mat -P 10 ${barcodeFile} <(pv
reads_all.fastq.gz) | \
  maf-convert -n tab | cut -f 2,7 | sort | uniq | \
  gzip > barcode_assignments.txt.gz
```

The output of this command will be a gzipped tab-separated 2-column file with barcode names in the first column, and read IDs in the second column.

Splitting Read File Per Barcode

7 For each discovered barcode, using the appropriate read category assignment file, find the corresponding read IDs, then extract those IDs out of the read FASTQ file. This uses one of my scripts, fastx-fetch.pl, to do this directly from a FASTQ file. The '*-lengths*' command-line parameter also outputs sequence lengths for each read (see next step):

fastx-fetch.pl

```
mkdir -p demultiplexed
fastx-fetch.pl -lengths -demultiplex barcode_assignments.txt.gz \
  -prefix 'demultiplexed/reads' <(pv reads_all.fastq.gz) >
barcode_counts.txt
```

Note: this demultiplexing code will only by default put reads into a barcode bin if they have a single unique barcode sequence detected. Otherwise, they will be put into a 'BCchim' bin if multiple adapters are detected (i.e. a chimeric read), or a 'BCmiss' bin if no adapters are detected. If these reads should be duplicated and put in one bin per barcode, then the *-chimeric* option can be added to the command arguments:

```
mkdir -p demultiplexed
fastx-fetch.pl -lengths -demultiplex barcode_assignments.txt.gz -chimeric
\
  -prefix 'demultiplexed/reads' <(pv reads_all.fastq.gz) >
barcode_counts.txt
```

[optional] Displaying Read Length Statistics

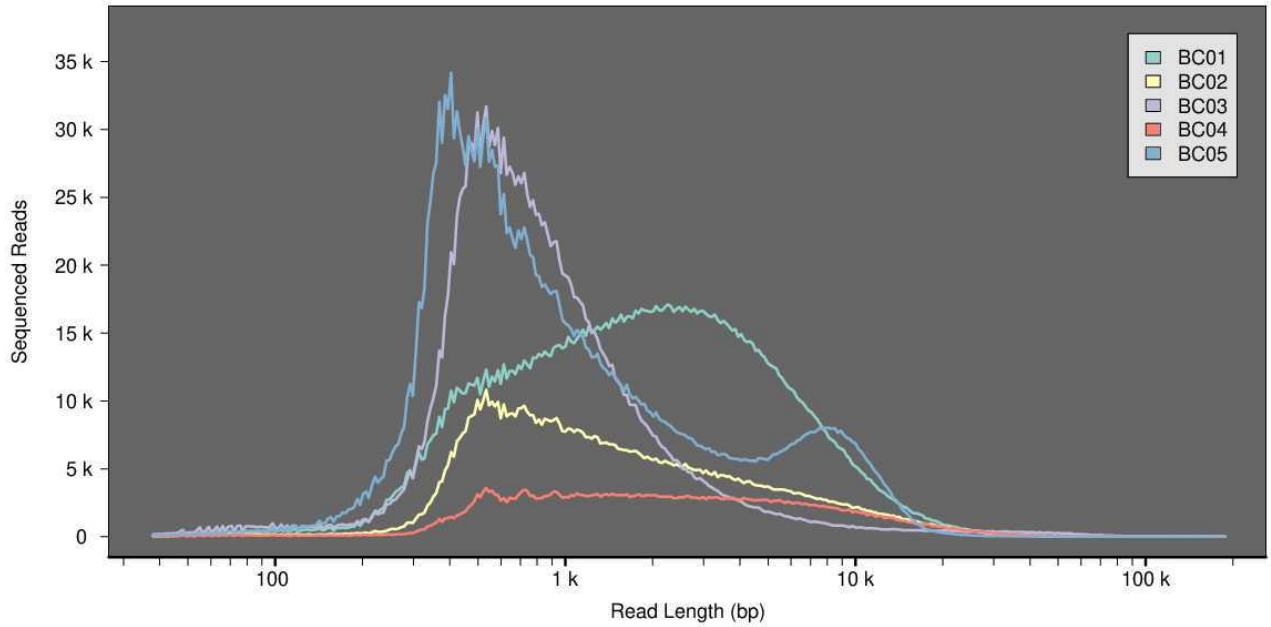
8 The *lengths* output from the demultiplexing step can be fed into another one of my scripts, length-plot.r, in order to display length-based QC plots:

```
length_plot.r demultiplexed/lengths_*.txt.gz
```

As output, this produces a multi-page PDF file, *Sequence_curves.pdf*. Here are some examples of the plots that are produced:

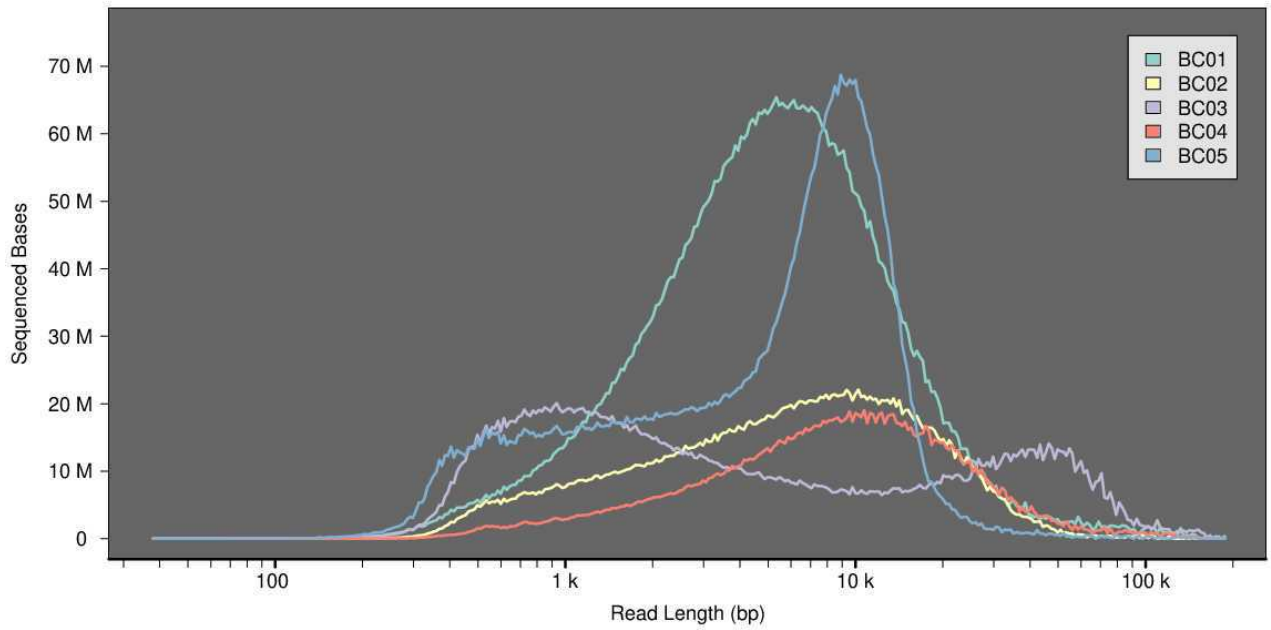
Citation: David A Eccles Demultiplexing Nanopore reads with LAST https://dx.doi.org/10.17504/protocols.io.14egnxw4zl5d/v8

1. Read Count Frequency Curve



Read count frequency curve for five samples, showing a variety of different read length distributions

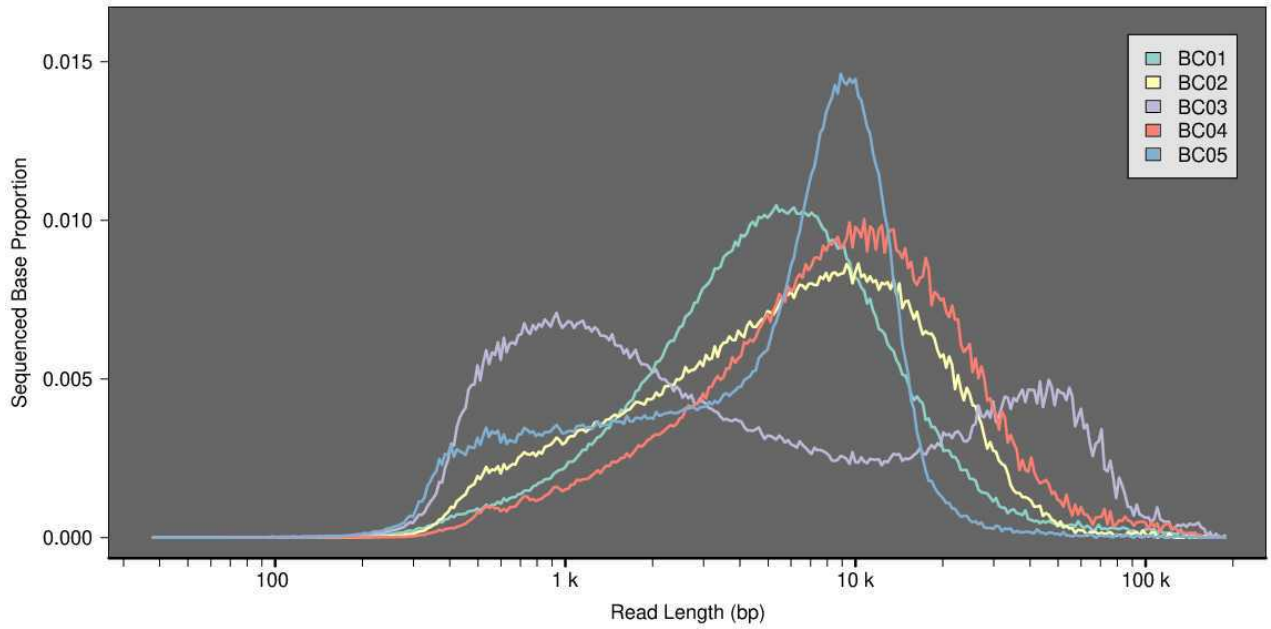
2. Called Bases Frequency Curve



Called bases frequency curve for five samples, showing a variety of different read length distributions

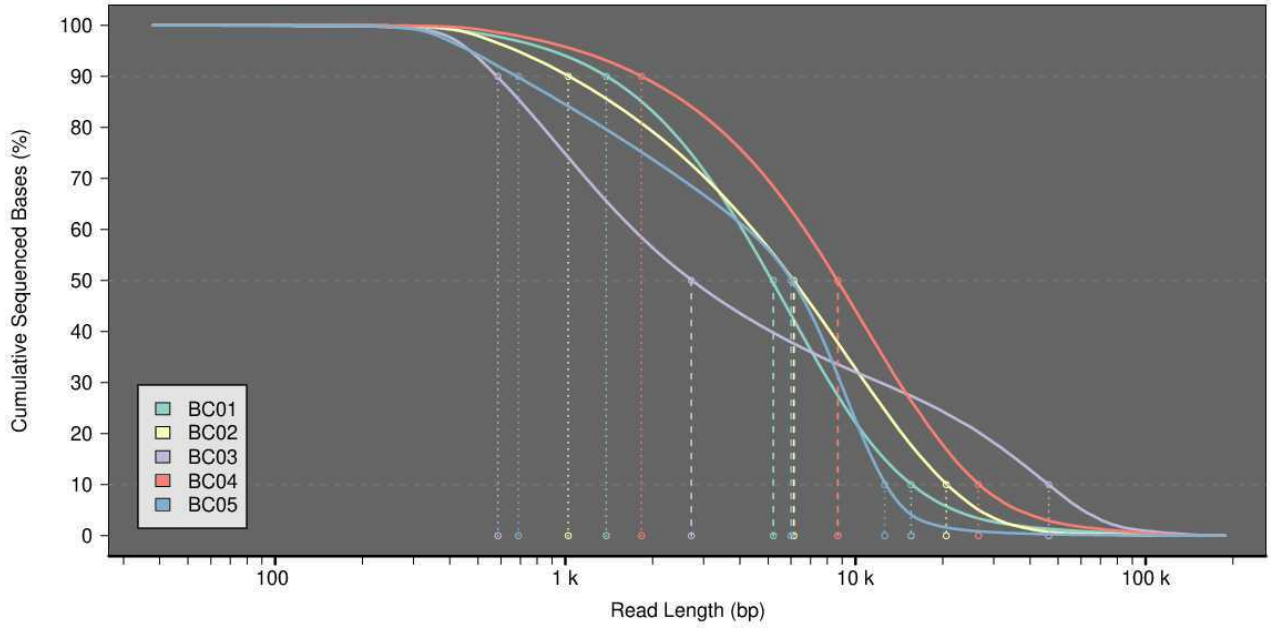
3. Called Bases Density Curve

Citation: David A Eccles Demultiplexing Nanopore reads with LAST https://dx.doi.org/10.17504/protocols.io.14egnxw4zl5d/v8



Sample-normalised called bases density curve for five samples, showing a variety of different read length distributions

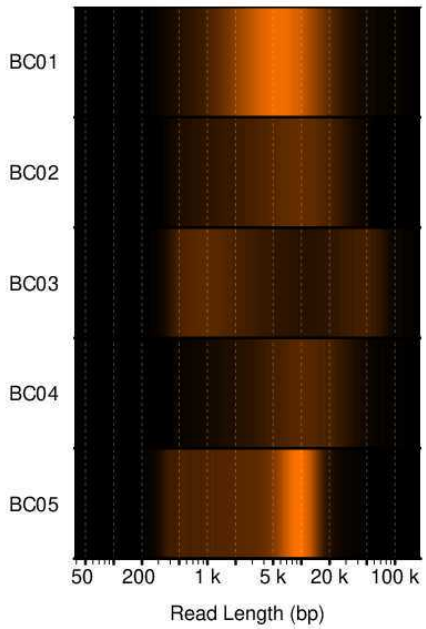
4. Cumulative Sequenced Bases Curve



Sample-normalised cumulative sequenced base curve for five samples, showing a variety of different read length distributions

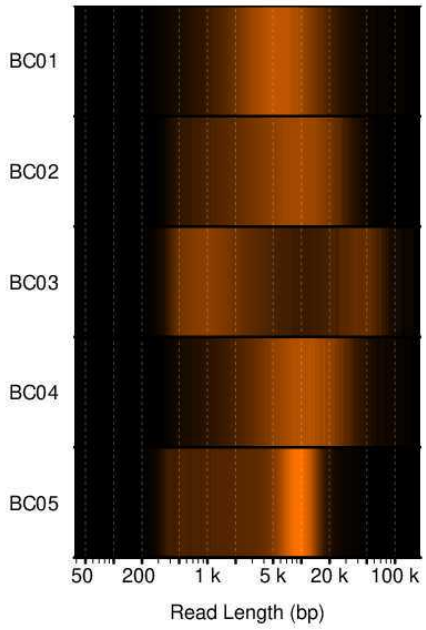
5. Digital Electrophoresis Plot (relative frequency)

Citation: David A Eccles Demultiplexing Nanopore reads with LAST https://dx.doi.org/10.17504/protocols.io.14egnxw4zl5d/v8



Relative digital electrophoresis plot for five samples, showing a variety of different read length distributions

5. Digital Electrophoresis Plot (sample-normalised)



Sample-normalised digital electrophoresis plot for five samples, showing a variety of different read length distributions

Downstream Workflows

9 Following on from here, cDNA reads can be oriented in preparation for stranded mapping:

- Preparing Reads for Stranded Mapping

Plasmid DNA sequences can be mapped or assembled:

Citation: David A Eccles Demultiplexing Nanopore reads with LAST https://dx.doi.org/10.17504/protocols.io.14egnxw4zl5d/v8

- Plasmid Sequence Analysis from Long Reads

Citation: David A Eccles Demultiplexing Nanopore reads with LAST https://dx.doi.org/10.17504/protocols.io.14egnxw4zl5d/v8