Enhancing nanopore adaptive sampling for PromethION using readfish at scale.

Rory Munro[*1], Alexander Payne[*1], Nadine Holmes[2], Chris Moore[2], Inswasti Cahyani[1], and Matthew Loose[1]

[1] *School of Life Sciences, University of Nottingham*
[2] *Deepseq, School of Life Sciences, University of Nottingham*
[*] *These authors contributed equally to this work.*

Abstract

A unique feature of Oxford Nanopore Technologies sequencers, adaptive sampling, allows precise DNA molecule selection from sequencing libraries. Here we present enhancements to our tool, readfish, enabling all features for the industrial scale PromethION sequencer, including standard and "barcode-aware" adaptive sampling. We demonstrate effective coverage enrichment and assessment of multiple human genomes for copy number and structural variation on a single PromethION flow cell.

Introduction

Adaptive Sampling (Loose et al. 2016) allows for the optimisation of sequencing efficiency, reducing sequencing capacity wasted on DNA fragments which do not provide utility when answering a given biological question. The application of adaptive sampling to nanopore sequencing can address longstanding challenges inherent to other sequencing methods. It can lower sequencing costs and save time, alongside improving the depth and quality of sequencing data for targeted genomic regions. These issues are particularly present for large genomes, where a large amount of sequencing is required to produce sufficient data (Payne et al. 2021). By enhancing the relevance of the data produced, adaptive sampling can help with personalised medicine and diagnostics (Miyatake et al. 2022; Miller et al. 2021; Chen et al. 2024) and even with genomic environmental surveillance (Urban et al. 2023).

We previously developed readfish (Payne et al. 2021), which uses real-time base-calling to analyse molecules during translocation; determining if they should be sequenced or, instead, ejected from the pore to be replaced with a new molecule. Readfish has been instrumental in the development of adaptive sampling, providing researchers with unparalleled adaptability to address a wide range of biological questions (Patel et al. 2022; Stevanovski et al. 2022; Weilguny et al. 2023). Oxford Nanopore Technologies (ONT) also provide an adaptive sampling implementation built into MinKNOW, their software for controlling sequencing. This implementation is fundamentally the same as that of readfish, however it is deeply embedded in MinKNOW, making it hard to customise and inflexible in certain scenarios. For example, it does not offer the ability to alter genomic targets during sequencing.

With the release of the PromethION, nanopore sequencing has grown in throughput, highlighting the requirement for additional features and improvements in readfish. Combining adaptive sampling with PromethION scale sequencing is extremely beneficial; if the throughput of sequence production is increased, the effect of enrichment is compounded. Increased coverage over target regions allows for the determination of Structural Variants (SVs) and Single Nucleotide Polymorphisms (SNPs) with greater confidence (Beyter et al. 2021). However, the increased data generation rates enabled by PromethION flow cells is too large for the original implementation of readfish. Specifically, the time taken to process all the alignments using `mappy` (Li 2018; Li 2021) (minimap2 Python bindings) led to significant bottlenecks in the analysis pipeline.

To this end, we have addressed issues with previous versions of readfish, refactoring the source code for better maintainability, efficiency, extensibility, and stability. We demonstrate the ability of readfish to keep up with PromethION scale sequencing, using a custom multithreaded implementation of `mappy` in Rust. Readfish has a number of features such as "barcode-awareness" (Munro, Holmes, et al. 2023) and compatibility with the latest dorado versions, all of which are now available for PromethION scale experiments. As a demonstration of these improvements, we multiplexed and sequenced three different target panels across distinct human cell lines, confirming known SVs. We proceed to use the alignment of sequenced and rejected reads for precise copy number variation (CNV) assessment on PromethION.

Results

Barcode demultiplexing

Using base-calling in adaptive sampling decision-making allows existing sequence based tools, such as barcode demultiplexers, to be incorporated into the readfish workflow. We previously adapted readfish to be compatible with built-in Guppy or Dorado demultiplexing (ONT) and incorporated barcode classifications into the data readfish can use to make a decision about sequencing or rejecting a read (Munro, Holmes, et al. 2023). This "barcode-awareness" is now

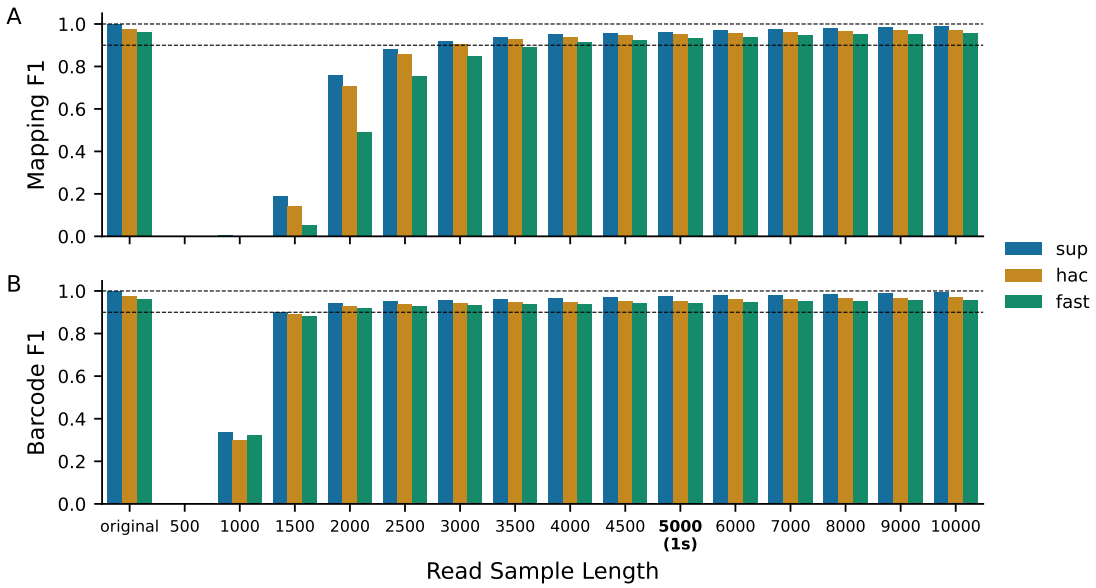

Figure 1: Comparison of Alignment and Barcode classification F1 as a result of signal length. 10,000 reads were truncated into 500 sample increments up to 5000 samples, with a 1000 sample increment after that, to a maximum of 10000. Reads were base-called and demultiplexed using the super accuracy (sup), high accuracy (hac) and fast (fast) dorado models. A) F1 scores for alignment, where "truth" alignments were defined as the start position of the mapping being within 100 base pairs of the full length "sup" model read alignment. B) F1 scores for barcoding classifications, where "truth" barcode classifications were defined as the barcode classification as assigned by the "sup" model for the full length read.

functional on the PromethION, and can be leveraged for greater impact on this platform, due to the larger amount of data generated per barcode. Reads can be rejected based solely on barcode classification (Supplementary Figure 1A), or using independent target sets provided for their corresponding barcodes (Supplementary Figure 1B). This differs from the adaptive sampling built into MinKNOW which can also demultiplex barcoded reads for use in decision-making, as only one panel can be used for all barcodes. Currently, barcode based demultiplexing when used in built-in adaptive sampling does not have the same flexibility as readfish, and can only be used to balance barcodes if they are present at uneven ratios in the sequencing library. All barcode demultiplexing testing was performed on a PromethION

P24, with 2 Nvidia Quadro GV100s, 2 Intel(R) Xeon(R) Platinum 8168 CPU's (96 cores total), and 384GB RAM.

Barcodes can be accurately identified with very little signal data, as shown in Figure 1B. Given that the default number of samples taken per read chunk is 5000 on the PromethION (1 second at 5kHz), barcode classification has an F1 score of ≥ 0.9 at this number of samples, when compared to those the full length read produced. It would be possible to take less signal data and still receive an accurate alignment and barcode classification, which can be seen when examining both Figures 1A and 1B.

	Yield (Gb)		**N50 (bases)**		**Sample**	**Panel**	**Gene number**	**Off Target Med. Cov.**	**Target Med. Cov.**	**Fold Enrichment**
Barcode	**Seq.**	**Unb.**	**Seq.**	**Unb.**						
01	0.334	3.47	8,149	555	GM12878	TruSight 170 Tumour Panel	170	1	12	12
02	1.24	4.84	7,191	552	NB4	TruSight RNA Fusion Panel	508	1	15	15
03	1.25	3.84	6,858	556	22Rv1	COSMIC	717	1	12	12
unclassified	0.170	3.66	923	792	*	*	*	0	1	*
Total	**2.99**	**15.80**	**5,780**	**614**				**1**	**12**	
05	1.28	14.76	7,163	917	GM12878	TruSight 170 Tumour Panel	170	4	35	9
06	4.36	23.39	7,349	919	NB4	TruSight RNA Fusion Panel	508	8	52	7
07	3.01	13.63	6,999	923	22Rv1	COSMIC	717	4	27	7
unclassified	0.703	15.50	543	989	*	*	*	4	5	*
Total	**9.35**	**67.29**	**5,514**	**937**				**4**	**31**	

Table 1: Sample Performance. Run metric performance per barcode and over the entire flow cell. Metrics are derived from real-time monitoring with minoTour (Munro, Santos, et al. 2022), and from analysis of the final dataset. Barcodes 01-03 were run on a single GridION flow cell, barcodes 05-07 were run on a single PromethION flow cell.

PromethION and GridION experimental comparison

To test the performance difference when applying adaptive sampling on a PromethION compared to GridION, we used three previously described cell lines: GM12878, from the Utah/CEPH pedigree; NB4, a cell line carrying a fusion between *PML* and *RARA* representing an acute promyelocytic leukaemia (APL); and 22Rv1, a prostate cancer derived cell line containing significant chromosomal abnormalities (Jain et al. 2018; Liu et al. 2010; Mozziconacci et al. 2002). For each sample, we chose a specific gene panel. GM12878 was targeted using a panel defined by the gene list in the commercially available TruSight 170 Tumor panel (Na et al. 2019). As the NB4 cell line contains an APL fusion, we selected the TruSight RNA Fusion Panel (Siegfried et al. 2019). For the more complex 22Rv1 prostate cancer cell line, we used the previously described COSMIC panel (Payne et al. 2021; Tate et al. 2019). Samples were barcoded and sequenced on a single flow cell, and run for 72 hours (see methods and Table 1), on both PromethION and GridION. Alignment on GridION was performed by `mappy`, and on PromethION alignment was performed by Guppy, and *not* `mappy-rs`. These alignments are the same as those which would be used by MinKNOW's inbuilt adaptive sampling. At the time the experiment was run, adaptive sampling had only just become feasible on the PromethION. Therefore, we had implemented the most straightforward method to keep up with PromethION data production, which was inbuilt base-caller alignments. Whilst using built-in alignments is a feasible alternative to alignment by readfish, using the base-caller for alignment imposes restrictions, as it ties base-calling to dorado or guppy. Neither of these tools allow for alterations to the reference sequence during an experiment, limiting the ability to run true adaptive experiments.

On GridION, in a single experiment using a flow cell with 1,330 pores, 18.79 Gb of data were generated, with a total of 15 Gb successfully demultiplexed into barcoded data (Table 1). Inspection of individual targets *PML* and *RARA* demonstrates the ability to specifically target unique regions on each barcoded sample (Figures 2A and 2B). Current best practice for single nucleotide variant calling requires higher minimal depth than we achieve when looking at three samples on a MinION flow cell. However, long range Structural Variants can be determined, and so we used cuteSV (Jiang et al. 2020) to analyse these three samples. As expected, multiple reads supporting the detection of a fusion between *PML* and *RARA* were detected in the NB4 cell line (barcode 02, 06), as visualised using Genome Ribbon (Nattestad et al. 2021), and shown in Figure 3B. A full comparison of Structural variance across this region for all samples can be seen in Supplementary Figure 3.

For PromethION testing, a single experiment using a flow cell with 6,960 pores generated a total of 78.2 Gb of data, of which 61 Gb could be demultiplexed into barcoded data. See Table 1 for a complete breakdown of experimental statistics. Again, long range SVs can be

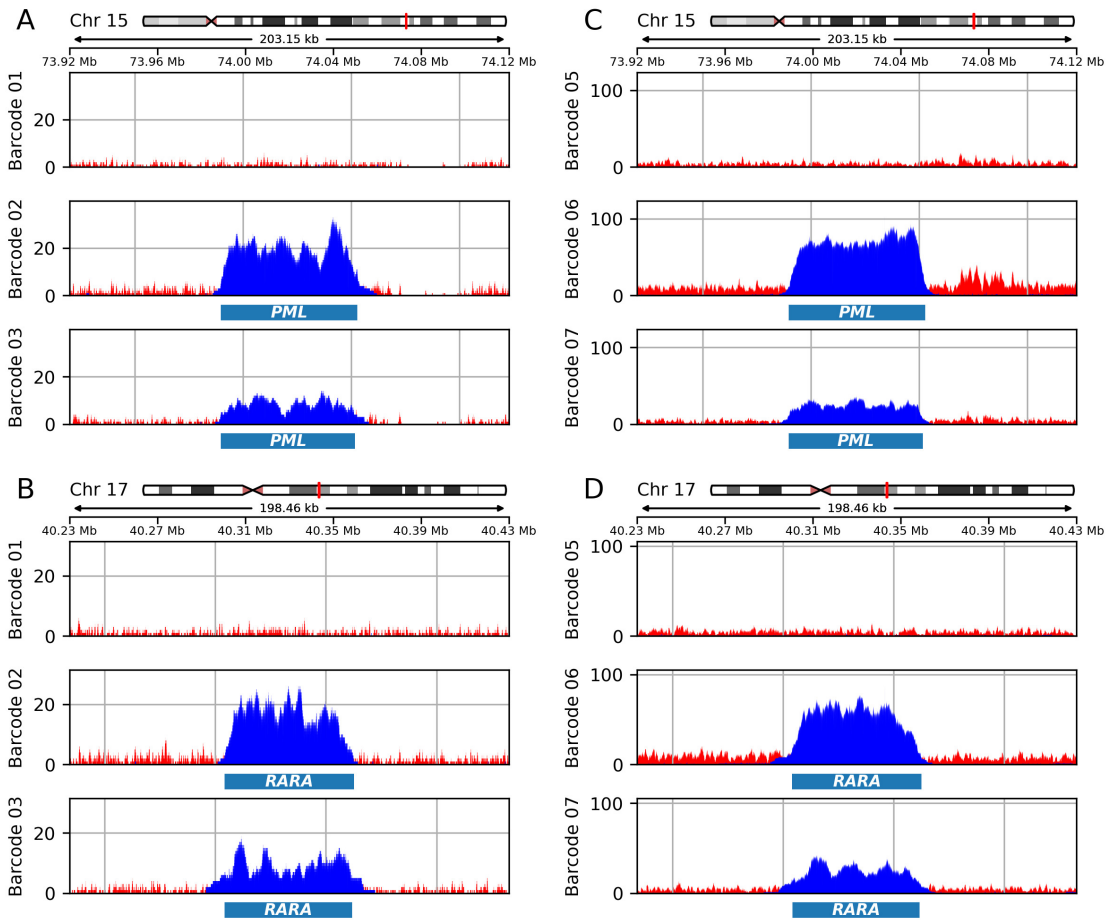

Figure 2: Target and barcode specific gene coverage. Illustration of coverage over each barcoded sample for the target genes *PML* and *RARA*. Blue is coverage from accepted read, red illustrates coverage from rejected reads. Barcodes 01 and 05: samples prepared from NA12878 cells; Barcodes 02 and 06: NB4; and Barcodes 03 and 07: 22Rv1. A-B) Data generated on a single MinION flowcell. The targeted regions are illustrated below the coverage plots. C-D) Data generated on a single PromethION flowcell.

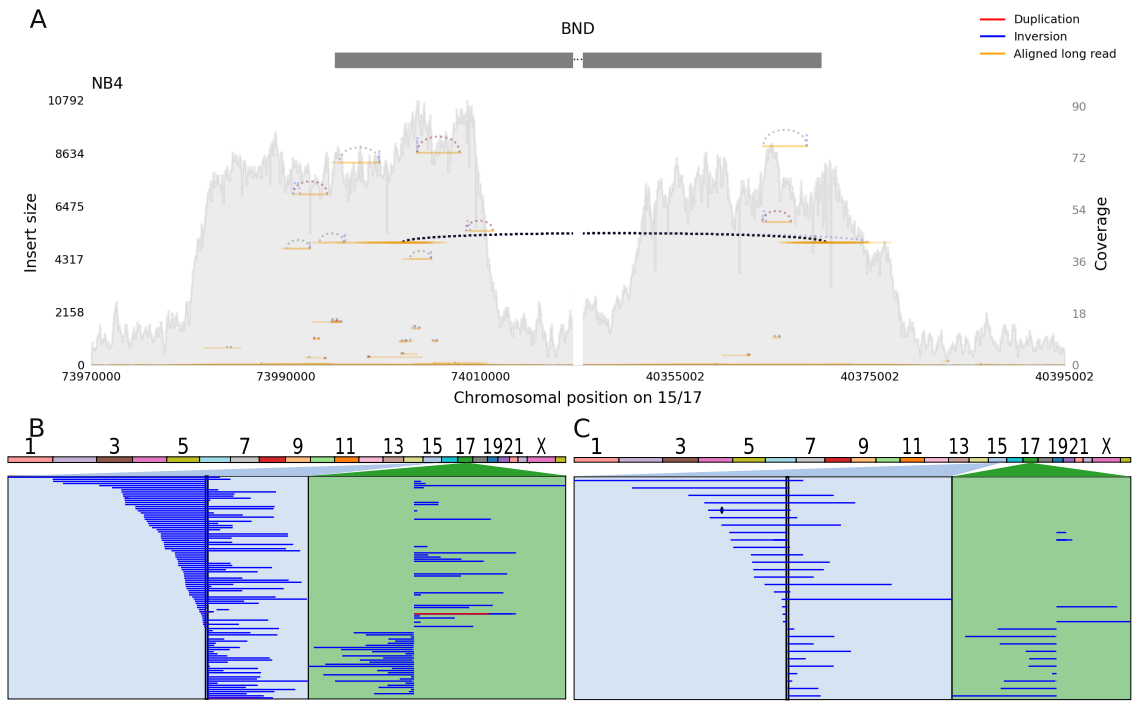

Figure 3: Visualising Structural Variation. A) Using samplot(Belyeu et al. 2021), reads from the PromethION run linking *PML* (Chromosome 15) to *RARA* (Chromosome 17) in a known fusion were visualised. Only the NB4 sample carries this fusion (indicated by the dashed lines). B-C) Using Ribbon (Nattestad et al. 2021) we can visualise individual reads which span the fusion from B) the GridION NB4 sample and C) the PromethION NB4 sample. SVs in this case were identified using CuteSV (Jiang et al. 2020).

determined, and as expected, multiple reads supporting the detection of a fusion between *PML* and *RARA* were detected in the NB4 cell line (barcode 02, 06), visualised using Genome Ribbon (Nattestad et al. 2021) and Samplot (Belyeu et al. 2021) (Figures 3A and 3C). To achieve this coverage across three samples without adaptive sampling would require 300–400 Gb of untargeted read data. Coverage compared with GridION is greatly improved (Figures 2C and 2D).

Finally, we turned to a natural application for adaptive-sampling, which considers the mappings of rejected reads. Various approaches have been developed using binning of short reads to detect CNV by applying a variety of statistical approaches (Zhang et al. 2019). These methods also work with nanopore sequencing (Magi et al. 2019), but the resolution of detection will be dependent on the total number of reads generated during a sequencing run. Adaptive sampling increases read count as a consequence of rejecting molecules once they are confidently mapped to an off-target region. As expected, CNV plots generated in this manner for NB4 (barcode 06) and 22Rv1 (barcode 07) (Figures 4A and 4C) both closely recapitulate CNV plots generated by Bionano optical mapping (Figures 4B and 4D). A full comparison across all samples can be seen in Supplementary Figure 4.

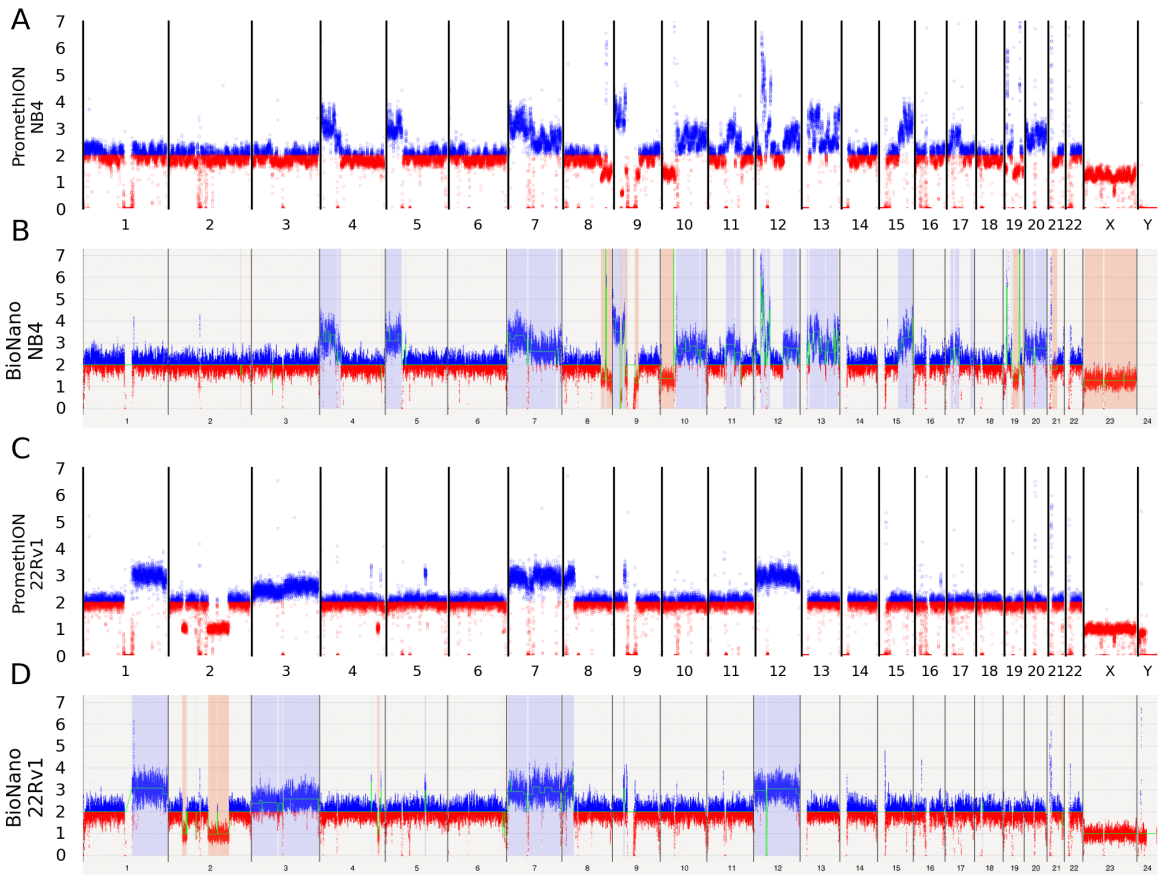

Figure 4: Matched Nanopore and Bionano CNV visualisation. Nanopore sequence data from PromethION compared with Bionano optically mapped reads, all mapped against hg38. Blue points show where binned data indicates greater than expected copy number, red points where binned data indicates lower than expected copy number. A) NB4 PromethION sample compared with B) showing Bionano optical mapping data for this cell line. C) 22rv1 PromethION sample compared with D) showing Bionano optical mapping data for this cell line.

Alignment throughput increase

Single threaded mappy is unable to keep up with the data generation rate of the PromethION. This can be clearly seen in Figure 5D with alignment times lagging upwards of 14 seconds within 2 minutes of sequencing. Simply implementing a multithreaded version of mappy through Python bindings resulted in unnecessary memory utilisation, as the reference library cannot easily be shared across threads. To address this, we wrote and integrated `mappy-rs` (https://github.com/Adoni5/mappy-rs) into readfish. `mappy-rs` exploits the ability of rust to better integrate with the underlying minimap2 code facilitating multithreaded alignment against a single instance of the reference in memory. By implementing a separate aligner, we can also update the reference during a run for increased customisability.

Signal data chunks are collected from the PromethION once every second as default. Therefore, if the combined time of basecalling, analysis and decision-making exceeds one second, the analysis will rapidly fall behind and negatively impact the ability to enrich target molecules. Using `mappy-rs`, alignment can be dramatically sped up (Supplementary Figure 2), preventing alignment from being a bottleneck which causes an increasing build up of signal and consequential time lag.

Figure 5 displays the time taken for the base-calling and alignment steps within the readfish decision-making loop when using `mappy`, `mappy-rs` and `dorado` for alignment at the GridION and PromethION scale. Figures 5A, 5C and 5E show that the choice of aligner makes little difference when sequencing with a MinION flow cell on a GridION MK1, with all three aligners performing similarly. Figure 5F confirms that the performance of `mappy-rs` on PromethION is sufficient to keep ahead of signal collection batch times on the PromethION. Performance is also largely comparable with using the built-in alignments which can be returned by dorado (Figure 5B), the same alignments that would be used by ONT's implementation of adaptive sampling. Although the times of several batches exceed the 1-second threshold for dorado and mappy-rs alignments in Figures 5B and 5F, the distribution plots in the margins show that this is actually quite a rare occurrence, with the distribution centred comfortably under 1 second. When comparing the peak of the alignment time distributions between Dorado and `mappy-rs` alignments in Figures 5B and 5F, we can see that `mappy-rs` is slightly slower. It is, however, more consistent, never exceeding 1.3 seconds, a threshold which is occasionally passed when using dorado alignments, for reasons we cannot currently explain. The colouring of the batches indicates the mean base-called read lengths contained in a batch. The cache for signal chunks from each channel is cumulative, if a read does not have a decision made on it by readfish in a batch, the next chunk of signal is appended to the already held signal. As we can see, for all but mappy on promethION Figure 5D, the mean read length of reads in a batch is almost always below 1000 bases (2.5 seconds of sequencing

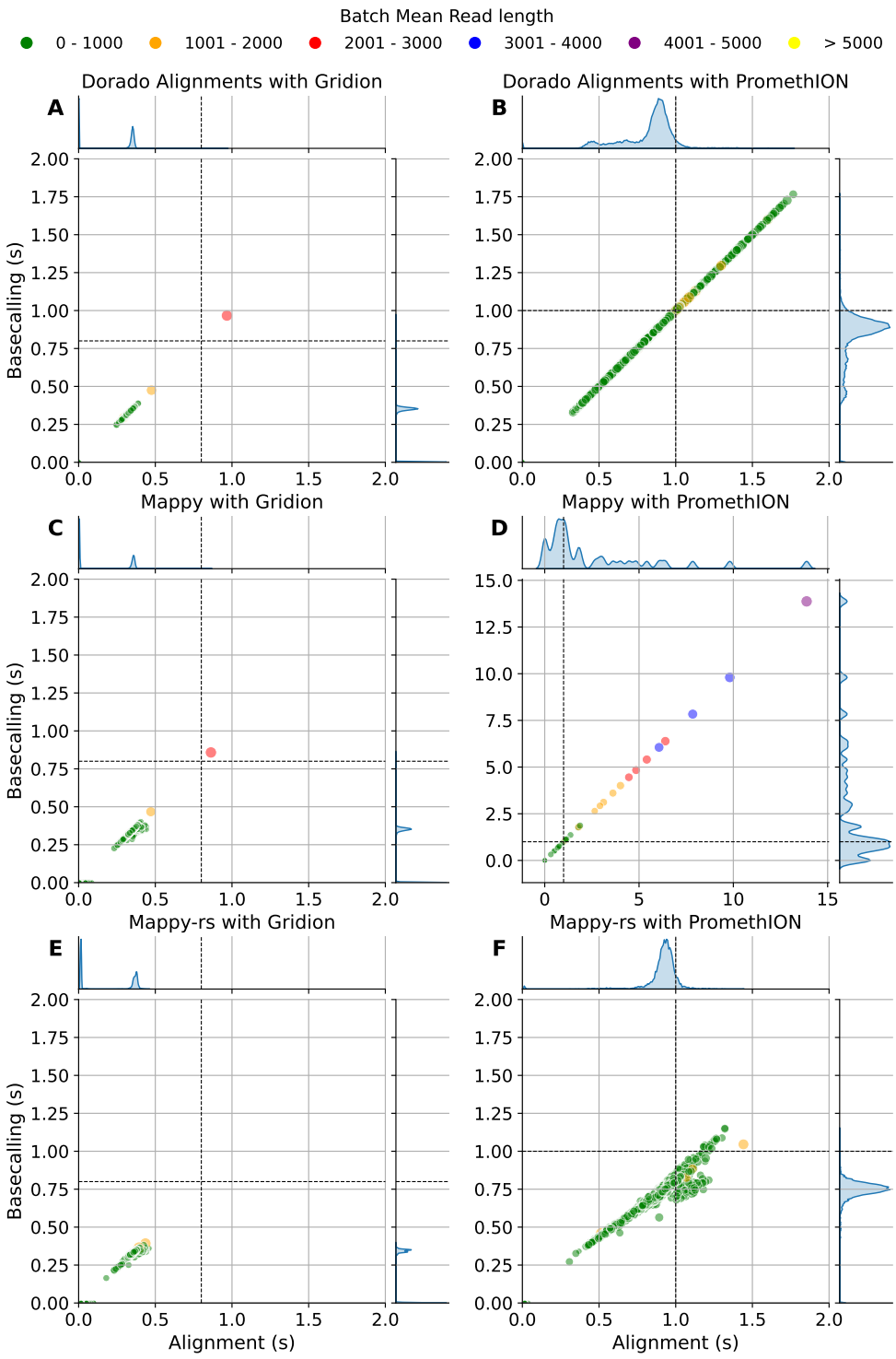

Figure 5: Total absolute alignment and base-calling times in readfish for different aligners on PromethION and GridION. Note that the base-called reads are streamed into the aligner for `mappy` and `mappy-rs`, so the two processes are occurring in parallel. Each scatter point represents a batch of accumulated signal chunks being analysed by readfish, and the colour represents the mean base-called length of reads in the batch. The 'break_read_chunk_ms' for each device is indicated on the x and y-axis of each plot as a dashed line, representing the amount of time each chunk of signal data recorded in a batch represents. Ideally, the total batch time should fall below this line. The marginal axes for each facet display Kernel Density estimation plots of the distribution of times for their axis. The PromethION/mappy combination has a different axes scale than the other facets, as batch processing times quickly lagged. When using the Dorado alignments, it was not possible to deconvolute the Alignment time from the base-calling time, as alignments are returned to readfish alongside the base-called signal, therefore points will fall on $x = y$.

at 400 bases per second), which indicates that there is no build-up of sequence.

Discussion

Extending readfish to process the volume of data a PromethION generates allows more sophisticated selection experiments, that are better able to exploit adaptive sampling's potential. By coupling high throughput PromethION sequencing with readfish's fully customisable "barcode-aware" adaptive sampling, targetted data generation can be fine-tuned based on individual samples, maximising the effect of enrichment per flow cell used. Here we demonstrate that individual samples can be targeted with unique panels of genes, selected based on knowledge of the sample, enabling a user to ask and answer specific questions. On a single MinION flow cell, 3 human genomes can be analysed in real-time, with coverage sufficient to detect SV and CNV. Furthermore, it is possible to target 3 human genomes on a single flow cell on PromethION devices to sufficient depth for further SNP analysis. We anticipate that further optimisation of the underlying software, both within proprietary MinKNOW control software and firmware as well as within tools such as readfish, will further enhance yield and throughput enabling effective targeted sequencing on 6–12 samples on a single PromethION flow cell. We demonstrate that by using the novel `mappy-rs`, it is possible for readfish to perform alignments at sufficient speed to keep up with PromethION, with the latest chemistry. The release of the P2i and the P2 solo will increase the amount of PromethION scale sequencing that is occurring outside larger sequencing centres. Whilst the adaptive sampling runs for this publication were performed on the provided P24 or P48 towers, for running adaptive sampling on both positions on a P2, we would recommend using at the minimum an Nvidia 3080, preferably a 4090, a 24 Core CPU and 64GB of RAM. Ignoring the compute requirements for base-calling, MinKNOW and controlling sequencing, we can see in Supplementary Figure 5 that the memory requirements for running readfish are roughly the size of the reference after transformation into a minimap2 index, and up to 1-2GB extra for storing signal and base-called sequence. In terms of CPU threads, a standard CPU would suffice for GridION, however for PromethION we would recommend at least an 8 core CPU with at least 4 threads dedicated to mappy-rs alignment.

Readfish at industrial scale, alongside sample multiplexing, could have numerous potential applications. In healthcare, it can be used for rapid profiling of central nervous system tumours (Vermeulen et al. 2023), and with the addition of multiplexed samples to the high throughput PromethION, multiple samples can now be concurrently analysed. We note that this method is dependent on mean read lengths of sufficient length to enable enrichment and so are dependent on sample extraction methods. The flexibility offered by readfish allows

for the updating of targets during an ongoing sequencing run, meaning that in conjunction with real-time analysis, potential genetic mutations and variations linked to diseases can be added to target regions in real-time if relevant. In the field of genomics, adaptive sampling can be applied to aid de novo assemblies, targeting reads too difficult to assemble regions, increasing the likelihood of longer nanopore reads spanning these regions and resolving them.

Methods

Mappy-rs development

minimap2 (Li 2018; Li 2021) is written in C, and provides a structured application programming interface (API) which allows for other programming languages to create bindings to the compiled C code, and execute it via a Foreign Function interface. This is how `mappy` is designed, using Cython in order to provide access to the C code alignment functions within the Python runtime.

However, in order to perform alignment at sufficient throughput to keep up with the output of a PromethION sequencer in real-time, we decided to design a multithreaded aligner with multiple copies of the Aligner sharing references to a single minimap2 index. Rust was chosen over C++ or C due to familiarity with the language, support for creating Python bindings and excellent support for parallelism and concurrency that is built Rust. Bindings to the minimap2 C library were generated using `bindgen` allowing for Rust code to call the underlying minimap2 C code. Custom Rust structs to represent the Aligner and Alignments were then created, and custom functions to perform the alignment were written, relying on the underlying C code to perform alignment calculations. Python bindings to the Rust code were generated using PyO3, creating `mappy-rs` https://github.com/Adoni5/mappy-rs. `mappy-rs` receives batches of sequence, and places them into a queue. A thread pool, where each thread has access to its own Aligner instance, pulls a sequence out from the queue, aligns it, and creates an Alignment instance to store the results, and then pushes that to a results queue. The results are yielded from the results queue back to the Python runtime in the order they were sent to the aligner.

Readfish and alignment timings

Six simulated sequencing runs were simulated on the PromethION P48 beta tower, with 4 Tesla V100-PCIE-16GB GPUs, 12 32GiB DIMM DDR4 Synchronous 2666 MHz sticks of RAM (384GB total) and 2 Intel(R) Xeon(R) Platinum 8168 CPU @ 2.70GHz (96 core total). Runs were simulated using Icarust 0.0.7, commit cf27f12071f7c9b515883d5e3bf645aad4609831 using the config_dnar10_5Khz_human_barcoded.toml, a copy of which has been included in the accompanying notebook repository. Simulated runs were run for two hours a piece using the command: cargo run -r – -s Profile_tomls/config_dnar10_5khz_human_barcoded.toml -c config_grid.ini -v -p Where the number of channels and the break_read_chunks was changed to 512 and 0.8 for GridION simulation and 3000 and 1.0 for PromethION.

Dorado base-call server v7.3.9 was used for base-calling for readfish and alignment for

using readfish with dorado alignments. Alignment was performed, using the Hg38.p14 reference, with only complete, primary chromosomes present in the reference. Multithreaded alignments for both Dorado and mappy-rs used 16 threads.

Profiling timings were generating using a custom fork of readfish v2024.2.0, commit cd20ff16c5f3a5f54124515fe58aabde3dc8df3a, https://github.com/LooseLab/readfish/tree/cd20ff16c5f3a5f54124515fe58aabde3dc8df3a. Custom analysis was performed in jupyter-notebooks, code available in attached repository.

Truncated read generation and analysis

All analysis code notebooks can be found at https://github.com/LooseLab/barcode_paper_nb. Analysis and base-calling were performed on the same PromethION P48 tower as above. Reads were truncated using https://pypi.org/project/pod5/ from ONT and a custom Python script, and truncated reads were written into valid POD5 files for each truncation length. Truncation was done in 0.1 second increments (500 samples) up until one second's worth of data, after which a 0.2 second increment (1000 sample) was used. 10000 Reads were taken from a human clinical run, the library was prepared using the SQK-NBD114-24 sequencing kit, and was sequenced at 5kHz on a R10.4.1 flow cell.

Each set of truncated reads and the original full length reads were then base-called with Dorado base-call server 7.3.9, using the Fast, High Accuracy and Super base-calling models at v4.3. For barcode demultiplexing of data, we used Dorado v7.3.9 demultiplexing and tested no other approach.

Alignment was performed by Dorado base-call server 7.3.9, which internally uses minimap2 v2.24. Reads were aligned against the Hg38.p14 reference with all but the primary contigs removed.

Running readfish barcoding

Running adaptive sampling requires the ONT Read Until API (version 3.0.0, https://github.com/nanoporetech/read_until_api/tree/release-3.0 and the ONT PyGuppy Client library (version 5.0.13, https://pypi.org/project/ont-pyguppy-client-lib/5.0.13/). Readfish (https://github.com/LooseLab/readfish; commit 9e8794a) was run using a GridION MK1 (MinKNOW v4.3.2; Guppy v5.0.13; minimap2 v2.22), the MinKNOW configuration scripts were configured to serve data in 0.8 second chunks. For PromethION, we ran using a modified version of readfish (https://github.com/LooseLab/readfish; commit c9f5169) on an early release of MinKNOW core 5.1 using a PromethION 24 device (MinKNOW v5.1; Guppy v6.0.6) with the ONT PyGuppy Client library (version 6.0.6, https://pypi.org/project/ont

-pyguppy-client-lib/5.0.13/). MinKNOW configuration scripts were left serving data in 1 second chunks for PromethION.

The readfish script carrying out the selective sequencing was `readfish barcode-targets`. This script runs the core Read Until process as specified in the experiment's TOML file. With a single reference genome, the script can select specific target regions on each barcode by using Guppy to base call and demultiplex the raw signal in real-time. The resultant read is then aligned to the reference using minimap2 and is determined to be on or off target depending on its barcode assignment and mapping start. For PromethION, we used the mapping returned by minimap2 from within Guppy to make decisions.

Library preparation, sequencing, and analysis

Barcoded LSK-110 (ONT) sequencing libraries were prepared from either GM12878 cells (Coriell), NB4 cells (gift from M. Hubank) or 22Rv1 cells (ATCC) as described in Jain et al. (Jain et al. 2018). For test experiments, bacterial DNA was extracted using genomic tip (QIAGEN). Extracted DNA was sheared to approximately 12 kb using g-Tube (Covaris). Sequencing used either FLO-MIN106 R9.4.1 flow cells for GridION or FLO-PRO002/FLO-PRO114 R9.4.1/R10.4.1 flow cells for PromethION as approriate. Flow cells were run with flushing and reloading as previously described in (Payne et al. 2021).

For Figure generation, see the following GitHub repository for data and notebooks https://github.com/LooseLab/barcode_paper_nb. To investigate SVs across the dataset, we ran CuteSV https://github.com/tjiangHIT/cuteSV on each barcoded sample using standard options but varying the -s MIN SUPPORT values, altering the minimum number of reads required to support a SV. No SVs in known fusion genes were reported in NA12878 or 22Rv1 (-s 2), known fusions including *PML* and *RARA* were readily detected in NB4 (-s 5)(Jiang et al. 2020). SVs were visualised using Ribbon (Nattestad et al. 2021).

To visualise changes in copy number, reads were mapped to hg38, filtered to mapping scores $>$ 20 and uniquely mapping. Then the first primary mapping for any read was determined and mappings binned into windows along the genome such that on average each bin contains 100 reads. Runs were monitored in real-time using minoTour (https://github.com/LooseLab/minotourapp; commit: 1f9c678), providing coverage statistics, mappings and estimates of copy number variation in real-time (Munro, Santos, et al. 2022). During real-time analysis, reads were mapped to Chm13 telomere-to-telomere assembly (Nurk et al. 2022). Post-run copy number plots were generated using Matplotlib with data mapped to hg38 to compare with the output of the Bionano copy number pipeline (see notebooks https://github.com/LooseLab/barcode_paper_nb).

To visualise coverage over specific targets, reads were divided into those actively sequenced and those unblocked using the unblocked read IDs file generated by readfish. Reads were mapped to hg38, coverage depth calculated using mosdepth v0.3.1 (Pedersen and Quinlan 2018) and visualised using Matplotlib (v3.4.3).

Bionano methods

DNA extraction and labelling for Bionano

DNA was prepared from frozen cell pellets of 1.5 million cells using the Bionano Prep SP Blood and Cell Culture DNA Isolation Kit (Bionano Genomics; 80042) according to the manufacturer's instructions. DNA was homogenised and quantified using Qubit dsDNA BR Kit (Thermo Fisher; Q32853) on a Qubit 4 Fluorometer (Thermo Fisher; Q33238). 750 ng of gDNA was then labelled with Direct Label Enzyme 1 (DLE-1) and DNA backbone stain using the Bionano Prep Direct Label and Stain (DLS) kit (Bionano Genomics; 80005) according to the manufacturer's instructions. Labelled DNA was quantified using the Qubit dsDNA HS Kit (Thermo Fisher; Q32851) on a Qubit 4 Fluorometer. Labelled DNA was loaded onto a Bionano Saphyr G2.3 chip (Bionano Genomics; 20366) and run on a Gen 2 Bionano Saphyr System (Bionano Genomics; 60325) until 1.320 Tbp of data had been collected for each of NB4 and 22Rv1. This data had respective mapping rates to hg38 reference sequence of 89% and 79%, equating to 382x and 337x coverage respectively.

Data analysis

Post run data filtering and analysis was carried out using Bionano Access 1.5.2. For each sample the data set was filtered and sub-sampled to produce 320 Gbp of data with 150 kb minimum length and at least 9 labels per molecule. Filtered data was processed to produce annotated *de novo* assemblies using the default parameters, but with masking using the hg38 DLE-1 SV Mask BED file. SVs and CNVs coordinates were then visualised using Bionano Access. All described analysis was performed on dedicated Bionano compute with the following versions installed: Bionano Access1.5.2, Bionano Tools 1.5.3, Bionano Solve Solve3.5.1_01142020, RefAligner 10330.10436rel, HybridScaffold 12162019, SVMerge 12162019 , VariantAnnotation 12162019, Compute on Demand 1.5.1.

Data access

Data and scripts used for this manuscript are available at GitHub (https://github.com/LooseLab/barcode_paper_nb) and as Supplemental Material. Sequence data and bionano maps

generated for the GridION and PromethION experimental comparison are available from ENA under project accession PRJEB82322.

Acknowledgements

The authors thank Mike Hubank and Nigel Mongan for gifts of cells and useful discussions. We also thank Stu Reid, Graham Hall, Chris Wright and teams at ONT for useful conversations. This work was supported by BBSRC iCASE studentship awards to RM and AP. In addition we acknowledge funding from the BBSRC (BB/N017099/1) and Wellcome Trust (grant number 204843/Z/16/Z). We would like to acknowledge Deepseq Nottingham for the creation and sequencing of all DNA libraries. Author contributions: R.M, A.P and M.L conceptualised the study, R.M and A.P performed data analysis, R.M and M.L wrote the manuscript. N.H, C.M, I.C performed sequencing and DNA extraction for experiments.

Conflict of Interest statements

ML was a member of the MinION access program and has received free flow cells and sequencing reagents in the past. ML has received reimbursement for travel, accommodation and conference fees to speak at events organised by Oxford Nanopore Technologies. Early access to MinKNOW software updates from Oxford Nanopore Technologies enabled this work. RM is currently on a BBSRC iCASE PhD programme, which is in part funded by Nanopore.

References

Belyeu, Jonathan R. et al. "Samplot: a platform for structural variant visual validation and automated filtering". In: *Genome Biology* 22 (1 Dec. 2021), p. 161. ISSN: 1474-760X. DOI: 10.1186/s13059-021-02380-5.

Beyter, Doruk et al. "Long-read sequencing of 3,622 Icelanders provides insight into the role of structural variants in human diseases and other traits". In: *Nature Genetics* 53 (6 June 2021), pp. 779–786. ISSN: 15461718. DOI: 10.1038/s41588-021-00865-4.

Chen, Zhongbo et al. "Adaptive Long-Read Sequencing Reveals GGC Repeat Expansion in ZFHX3 Associated with Spinocerebellar Ataxia Type 4". In: *Movement Disorders* (2024). ISSN: 15318257. DOI: 10.1002/mds.29704.

Jain, Miten et al. “Nanopore sequencing and assembly of a human genome with ultra-long reads”. In: *Nature Biotechnology* 36 (4 Apr. 2018), pp. 338–345. ISSN: 1087-0156. DOI: 10.1038/nbt.4060.

Jiang, Tao et al. “Long-read-based human genomic structural variation detection with cuteSV”. In: *Genome Biology* 21 (1 Dec. 2020), p. 189. ISSN: 1474-760X. DOI: 10.1186/s13059-020-02107-y.

Li, Heng. “Minimap2: pairwise alignment for nucleotide sequences”. In: *Bioinformatics* 34 (18 Sept. 2018), pp. 3094–3100. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bty191. URL: https://doi.org/10.1093/bioinformatics/bty191.

Li, Heng. “New strategies to improve minimap2 alignment accuracy”. In: *Bioinformatics* 37 (23 Dec. 2021), pp. 4572–4574. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btab705.

Liu, Te et al. “Establishment and characterization of multi-drug resistant, prostate carcinoma-initiating stem-like cells from human prostate cancer cell lines 22RV1”. In: *Molecular and Cellular Biochemistry* 340 (1-2 July 2010), pp. 265–273. ISSN: 0300-8177. DOI: 10.1007/s11010-010-0426-5.

Loose, Matthew et al. “Real-time selective sequencing using nanopore technology”. en. In: *Nat. Methods* 13.9 (Sept. 2016), pp. 751–754.

Magi, Alberto et al. “Nano-GLADIATOR: Real-time detection of copy number alterations from nanopore sequencing data”. In: *Bioinformatics* 35 (21 Nov. 2019), pp. 4213–4221. ISSN: 14602059. DOI: 10.1093/bioinformatics/btz241.

Miller, Danny E et al. “Targeted long-read sequencing identifies missing disease-causing variation”. In: *The American Journal of Human Genetics* 108 (8 2021), pp. 1436–1449. ISSN: 0002-9297. DOI: https://doi.org/10.1016/j.ajhg.2021.06.006. URL: https://www.sciencedirect.com/science/article/pii/S0002929721002305.

Miyatake, Satoko et al. “Rapid and comprehensive diagnostic method for repeat expansion diseases using nanopore sequencing”. In: *npj Genomic Medicine* 7 (1 Dec. 2022). ISSN: 20567944. DOI: 10.1038/s41525-022-00331-y.

Mozziconacci, Marie-Joelle et al. “Molecular cytogenetics of the acute promyelocytic leukemia-derived cell line NB4 and of four all-trans retinoic acid-resistant subclones”. In: *Genes, Chromosomes and Cancer* 35 (3 Nov. 2002), pp. 261–270. ISSN: 1045-2257. DOI: 10.1002/gcc.10117.

Munro, Rory, Nadine Holmes, et al. “A framework for real-time monitoring, analysis and adaptive sampling of viral amplicon nanopore sequencing”. In: *Frontiers in Genetics* 14 (Mar. 2023). ISSN: 1664-8021. DOI: 10.3389/fgene.2023.1138582. URL: http://dx.doi.org/10.3389/fgene.2023.1138582.

Munro, Rory, Roberto Santos, et al. "minoTour, real-time monitoring and analysis for nanopore sequencers". In: *Bioinformatics* 38 (4 Feb. 2022), pp. 1133–1135. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btab780. URL: https://doi.org/10.1093/bioinformatics/btab780.

Na, Kiyong et al. "Targeted next-generation sequencing panel (TruSight Tumor 170) in diffuse glioma: a single institutional experience of 135 cases". In: *Journal of Neuro-Oncology* 142 (3 May 2019), pp. 445–454. ISSN: 0167-594X. DOI: 10.1007/s11060-019-03114-1.

Nattestad, Maria et al. "Ribbon: intuitive visualization for complex genomic variation". In: *Bioinformatics* 37 (3 Apr. 2021), pp. 413–415. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btaa680.

Nurk, Sergey et al. "The complete sequence of a human genome". In: *Science* 376 (6588 2022), pp. 44–53. DOI: 10.1126/science.abj6987. URL: https://www.science.org/doi/abs/10.1126/science.abj6987.

Patel, Areeba et al. "Rapid-CNS2: rapid comprehensive adaptive nanopore-sequencing of CNS tumors, a proof-of-concept study". In: *Acta Neuropathologica* 143 (5 May 2022), pp. 609–612. ISSN: 14320533. DOI: 10.1007/s00401-022-02415-6.

Payne, Alexander et al. "Readfish enables targeted nanopore sequencing of gigabase-sized genomes". In: *Nature Biotechnology* 39 (4 2021), pp. 442–450. ISSN: 1546-1696. DOI: 10.1038/s41587-020-00746-x. URL: https://doi.org/10.1038/s41587-020-00746-x.

Pedersen, Brent S and Aaron R Quinlan. "Mosdepth: quick coverage calculation for genomes and exomes". In: *Bioinformatics* 34 (5 Mar. 2018), pp. 867–868. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btx699.

Siegfried, Aurore et al. "EWSR1-PATZ1 gene fusion may define a new glioneuronal tumor entity". In: *Brain Pathology* 29 (1 Jan. 2019), pp. 53–62. ISSN: 1015-6305. DOI: 10.1111/bpa.12619.

Stevanovski, Igor et al. "Comprehensive genetic diagnosis of tandem repeat expansion disorders with programmable targeted nanopore sequencing". In: *Marina Kennerson* 8 (2022), p. 17. URL: https://www.science.org.

Tate, John G et al. "COSMIC: the Catalogue Of Somatic Mutations In Cancer". In: *Nucleic Acids Research* 47 (D1 Jan. 2019), pp. D941–D947. ISSN: 0305-1048. DOI: 10.1093/nar/gky1015.

Urban, Lara et al. "Non-invasive real-time genomic monitoring of the critically endangered kākāpō". In: *eLife* 12 (Dec. 2023). ISSN: 2050084X. DOI: 10.7554/eLife.84553.

Vermeulen, C. et al. "Ultra-fast deep-learned CNS tumour classification during surgery". In: *Nature* 622.7984 (Oct. 2023), pp. 842–849. ISSN: 1476-4687. DOI: 10.1038/s41586-023-06615-2. URL: http://dx.doi.org/10.1038/s41586-023-06615-2.

Weilguny, Lukas et al. “Dynamic, adaptive sampling during nanopore sequencing using Bayesian experimental design”. In: *Nature Biotechnology* 41 (7 July 2023), pp. 1018–1025. ISSN: 15461696. DOI: 10.1038/s41587-022-01580-z.

Zhang, Le et al. “Comprehensively benchmarking applications for detecting copy number variation”. In: *PLOS Computational Biology* 15 (5 May 2019), e1007069. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1007069.