Bioinformatics, 2019, 1–3
doi: 10.1093/bioinformatics/btz461
Advance Access Publication Date: 13 June 2019
Applications Note

Genome analysis

Nanopype: a modular and scalable nanopore data processing pipeline

Pay Giesselmann[1], Sara Hetzel[1], Franz-Josef Müller[1,2], Alexander Meissner[1,*] and Helene Kretzmer[1]

[1]Department of Genome Regulation, Max Planck Institute for Molecular Genetics, Berlin, Germany and [2]Department of Psychiatry and Psychotherapy, Zentrum für Integrative Psychiatrie gGmbH, Universitätsklinikum Schleswig-Holstein, Campus Kiel, Kiel, Germany

*To whom correspondence should be addressed.

Associate Editor: Alfonso Valencia

Received on December 24, 2018; revised on April 30, 2019; editorial decision on May 27, 2019; accepted on June 7, 2019

Abstract

Summary: Long-read third-generation nanopore sequencing enables researchers to now address a range of questions that are difficult to tackle with short read approaches. The rapidly expanding user base and continuously increasing throughput have sparked the development of a growing number of specialized analysis tools. However, streamlined processing of nanopore datasets using reproducible and transparent workflows is still lacking. Here we present Nanopype, a nanopore data processing pipeline that integrates a diverse set of established bioinformatics software while maintaining consistent and standardized output formats. Seamless integration into compute cluster environments makes the framework suitable for high-throughput applications. As a result, Nanopype facilitates comparability of nanopore data analysis workflows and thereby should enhance the reproducibility of biological insights.

Availability and implementation: https://github.com/giesselmann/nanopype, https://nanopype.readthedocs.io.

Contact: meissner@molgen.mpg.de

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Third-generation sequencing techniques are currently introducing new perspectives to the field of genome analysis by generating previously unattainable read lengths with averages in the tens of thousands of nucleotides. Among other devices distributed by Oxford Nanopore Technologies (ONT), the MinION in particular is gaining prominence. In brief, the nanopore sequencing process is based on guiding a nucleotide polymer through a pore inserted in a membrane while measuring a change in ionic current as a proxy signal over time. This signal is then interpreted to determine the underlying DNA or RNA sequence. The nanopore technology enables direct readout of sequences from individual DNA or RNA molecules including base modifications since no synthesis or amplification is required.

Due to constant development and improvement of applications, frequent reprocessing of the raw signal and downstream data is necessary. Thus, novel archiving and processing strategies are needed for data storage and handling that scale with the large amount of data produced by the MinION sequencer. This will be even more relevant as higher throughput devices such as the PromethION become more widely available. Furthermore, a limiting factor of the applicability of this new technology is the currently available, research-grade software packages for nanopore long-read data analyses. These tend to be difficult to install and require complex software environments. Despite the growing number of recently developed algorithms (Magi *et al.*, 2018), primary data processing remains challenging due to stand alone tools without congruent data formats and requirements.

Current examples of nanopore data processing pipelines are *github.com/nanoporetech/katuali* for basecalling and assembly and *github.com/nanoporetech/pipeline-pinfish-analysis* for RNA isoform detection from cDNA and direct RNA sequencing experiments.

However they have been developed to perform specific analysis workflows without integrated handling of the critical raw data storage or version control of wrapped tools.

To overcome these issues, we have developed Nanopype, a pipeline designed explicitly for streamlined and automated nanopore long read processing. Apart from the integration of essential base calling, quality control and alignment tools, we facilitate a set of published analysis applications for barcode demultiplexing, DNA methylation readout, structural variant calling and RNA isoform detection.

Based on the Snakemake engine (Köster and Rahmann, 2012) our method integrates established error handling and uniform output structures across multiple experiments. Furthermore, Nanopype can be run in a parallel setup on both, single computers and server clusters. Deployed as a python module, Nanopype is mostly build from source with encapsulated routines to simplify the initial setup and integration into existing environments. Additionally, we provide Singularity images for all modules and an automatically built all-in-one Docker container. This enables the usage of the pipeline for both, less bioinformatically experienced experimental scientists and bioinformaticians. Lastly, Nanopype provides a well-defined framework for standardized processing independent of the underlying operating system.

2 Design

Nanopype's core element is a modular setup to easily update existing tools and to seamlessly integrate the latest developments. Nonetheless, each pipeline release is freezing the included tool versions to guarantee reproducible results. In a nutshell, Nanopype has been designed around three key components: raw data storage, tool encapsulation and standardized directory structures that mirror the applied toolchain.

Storage: The first core design component is the consistent storage of the raw signal data from any ONT sequencer. Nanopype is backward compatible with datasets of single read FAST5 files, for which we provide an import and packaging into TAR archives module. Indexing the content of both, packaged and the more recently realized bulk-FAST5 output enables the fast retrieval of individual reads. This raw data archive forms the basis for any downstream analyses and enables smooth re-processing of legacy datasets as soon as improved basecalling algorithms become available.

Encapsulation: The installation of experimental software packages still being actively developed with complex library dependencies can be time-consuming but remains essential to make use of the current-generation nanopore analysis workflows. On a base level, Nanopype uses Snakemake rules to wrap the build from source and installation process of its dependencies and therefore does not require root privileges for the setup on common Linux and MacOS systems. The build from source is the most customizable setup option and moreover allows the integration into complex, preexisting environments.

Internal wrappers are used to automatically build and deploy Singularity images for preset modules and pipeline versions. This mechanism enables the complete function set of Snakemake and Nanopype while only requiring a system-wide Python and Singularity installation.

An all-in-one Docker container is provided, wrapping the entire pipeline into a single environment. Primarily aimed for stand-alone usage, Windows systems and for initial testing, this method does not offer support for cluster computation. Consistent versioning ensures reproducibility independent of the installation method.

Transparency: Nanopype follows the Snakemake concept of output file driven computation: a single user command typically provokes transparent processing of any required intermediate result. Preexisting tools are integrated into consistent workflows and provide standard output formats to connect to workflows of established next-generation sequencing data analysis tools. The processing and subsequent output is intuitively organized in modules and underlying application directories.

3 Modules

Nanopype's backbone consists of a set of modules that resolve a specific task, like basecalling, alignment or further downstream analyses. If available, alternative applications are provided for the same task and grouped into a module with a coherent output format. Integrating first and foremost low-level nanopore data processing applications provided by ONT, established community developed software packages have been included in the first Nanopype release, as well.

Basecalling: The basecalling module translates raw nanopore signals into nucleotide sequences and is utilized by most subsequent pipeline layers. With the initial release, we include the established packages Guppy, Albacore and Flappie, all provided by ONT (Wick *et al.*, 2019). The default basecaller package is set to the recently released Guppy. Albacore is supported for backward compatibility but deprecated by ONT. The experimental Flappie is ONT's first DNA methylation-aware basecaller. It extends the usual four-letter nucleotide alphabet by a fifth letter for methylated cytosine in CpG contexts. For all basecallers, the output is either the standardized FASTA or FASTQ and supplemented by us with a basic quality control summary.

Alignment: The core functionality of the pipeline is the alignment of reads against a reference genome or draft assembly. Here, we provide three different aligners with distinct advantages, which make them favorable for different applications downstream. While Minimap2 (Li, 2018) is a fast, low memory footprint solution suitable for both DNA and RNA alignments, GraphMap (Sović *et al.*, 2016) is a sensitive aligner but with comparably high memory requirements. NGMLR (Sedlazeck *et al.*, 2018) is the recommended tool for the structural variation module. Any combination of basecalling, alignment and reference genome is supported and reports BAM format files.

DNA methylation: Sequencing without prior DNA amplification enables the direct readout of DNA base modifications. The current state of the art approach, Nanopolish (Simpson *et al.*, 2017) as well as the more experimental flip-flop basecaller Flappie, are incorporated into Nanopype. Subsequently, Nanopype splits Flappie's atypical sequence output into standard FASTQ and methylation status. DNA methylation at CpG dinucleotides of both tools is reported in a table format for single reads. Furthermore, we provide common bedGraph and bigWig files for genome-wide methylation tracks and thus enable downstream processing using established workflows, e.g. calling of differential methylated regions and comparison to bisulfite sequencing.

Structural variation: Detection and characterization of structural variation play a central role in cancer research and population genetics. Long read sequencing particularly facilitates investigation of variants with unprecedented accuracy and resolution. Therefore, Nanopype encompasses the variant caller Sniffles (Sedlazeck *et al.*, 2018) and provides output in the standard variant calling format (VCF).

Transcriptome: Another application of the long-read nanopore technology is sequencing of cDNA and RNA molecules directly. Recovery of full-length transcripts enables, for instance, the detection of alternatively spliced isoforms and is implemented in Nanopype using the Pinfish package (*github.com/nanoporetech/pinfish*). The output of polished transcripts is provided in the GFF format.

Demultiplexing: Barcoded sequencing allows pooling of multiple samples on a single flow-cell. The demultiplexing module uses Deepbinner (Wick *et al.*, 2018) to assign a barcode label to the individual reads. Thus, sequencing of comparable small bacterial genomes can be efficiently parallelized to use the available sequencing depth optimally.

Miscellaneous: Complemented by samtools (Li *et al.*, 2009), bedtools (Quinlan and Hall, 2010) and UCSCtools (Kent *et al.*, 2010) our pipeline establishes a comprehensive framework for ONT sequencing data processing.

4 Usage

Due to the functional range of Nanopype, dependent on the operating system and selected installation method the setup can require advanced information technology knowledge. However, after deployment, the subsequent usage is straightforward given basic command line understanding. Complete Nanopype workflows can be executed with a single concise command line call. For instance, local processing of multiple flow cells into a collective genome-wide methylation track of at least 5× coverage on reference hg38 requires only the following line:

```
> snakemake –snakefile ~/nanopype/Snakefile
methylation/nanopolish/ngmlr/guppy/sample.5x.hg38.bw
```

This command invokes basecalling, alignment and methylation detection using declared tools without further user interaction. The basecalling and alignment outputs are kept and can be reused to avoid redundant processing.

The automatic distribution of workflows into independent batches enables efficient handling of high-throughput experiments. This feature becomes particularly relevant for scaling in cluster environments and most importantly in case of terminated or failed batches. As a result, only failed batches require reprocessing by resuming the workflow from where it left off, using the same command which enhances the overall error robustness.

The source and Singularity versions of Nanopype do not interfere with the extensive cluster support of Snakemake.

5 Summary

Here we present Nanopype, a modular and easy-to-use data processing pipeline with a detailed online documentation, specifically designed to handle nanopore sequenced long-read data.

Nanopype provides end-to-end processing of the raw sequencer signal into standard data formats and consequently closes the gap to downstream next-generation sequencing algorithms. Single command invocations of entire workflows reduce the hands-on-time for users to receive the desired output. Implicitly, this also lowers the potential of user mistakes and deviations in processing of multiple datasets. As a consequence, workflows are easier to reproduce with fixed versions among datasets or repeated with improved tool releases on existing ones.

Nanopype is implemented as a python package and additionally provides pre-built and versioned Singularity and Docker images, making it favorable for effective usage in cluster and single computer environments. The pipeline design ensures portability and version controlled usage of the implemented tools, to enable consistent results across platforms.

Acknowledgments

We thank Jocelyn Charlton and Charles Haggerty for helpful discussion and proofreading.

Funding

F.J.M. was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy EXC 22167-390884018. P.G., S.H., A.M. and H.K. are supported by the Max Planck Society.

Conflict of Interest: none declared.

References

Kent,W.J. *et al.* (2010) BigWig and BigBed: enabling browsing of large distributed datasets. *Bioinformatics*, **26**, 2204–2207.

Köster,J. and Rahmann,S. (2012) Snakemake—a scalable bioinformatics workflow engine. *Bioinformatics*, **28**, 2520–2522.

Li,H. (2018) Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, **1**, 7.

Li,H. *et al.* (2009) The sequence alignment/map format and SAMtools. *Bioinformatics (Oxford, England)*, **25**, 2078–2079.

Magi,A. *et al.* (2018) Nanopore sequencing data analysis: state of the art, applications and challenges. *Brief. Bioinform.*, **19**, 1256–1272.

Quinlan,A.R. and Hall,I.M. (2010) BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics*, **26**, 841–842.

Sedlazeck,F.J. *et al.* (2018) Accurate detection of complex structural variations using single-molecule sequencing. *Nat. Methods*, **15**, 461.

Simpson,J.T. *et al.* (2017) Detecting DNA cytosine methylation using nanopore sequencing. *Nat. Methods*, **14**, 407.

SovíC,I. *et al.* (2016) Fast and sensitive mapping of nanopore sequencing reads with graphmap. *Nat. Commun.*, **7**, 11307.

Wick,R.R. *et al.* (2018) Deepbinner: demultiplexing barcoded oxford nanopore reads with deep convolutional neural networks. *PLoS Comput. Biol.*, **14**, e1006583.

Wick,R.R. *et al.* (2019) Performance of neural network basecalling tools for Oxford Nanopore sequencing. doi: 10.1101/543439.