ViralGenie: A Comprehensive Pipeline for Viral Metagenomics and Phylogenetic Analysis

Joon Klaps[1,*], Philippe Lemey[1], Liana Kafetzopoulou[1]

[1]Rega Institute for Medical Research, Department of Microbiology, Immunology and Transplantation, KU Leuven, Belgium

[*]Correspondence: joon.klaps@kuleuven.be

June 26, 2025

Abstract

Eukaryotic viruses present significant challenges in genome reconstruction and variant analysis due to their extensive diversity, quasi-species, absence of universal marker genes, and genome segmentation. While de novo assembly followed by reference database matching and consequently, read mapping is a common approach, manual execution of this workflow is extremely time-consuming, particularly due to the extensive reference verification and selection required. There is a critical need for an automated, scalable pipeline that can efficiently handle viral metagenomic analysis without manual intervention.

Here, we present nf-core/viralgenie, a comprehensive viral metagenomic pipeline for untargeted genome reconstruction, and variant analysis of eukaryotic viruses. Viralgenie is implemented as a modular Nextflow workflow that processes metagenomic and hybridization capture enriched samples to automatically detect and assemble viral genomes, while also performing variant analysis. The pipeline features automated reference selection, quality control metrics, comprehensive documentation, and seamless integration with containerization technologies including Docker, Singularity, and Podman. We demonstrate its utility and accuracy through validation on both simulated and real datasets, showing robust performance across diverse viral families and sample types.

nf-core/viralgenie is freely available at https://github.com/nf-core/viralgenie with comprehensive documentation at https://nf-co.re/viralgenie.

Keywords: viralgenie, bioinformatic pipeline, nextflow, viral metagenomics, viral assembly, viral variant analysis

1 Introduction

Reconstructing viral genomes from metagenomic sequencing data presents considerable computational challenges, particularly for viruses that exhibit extensive genetic diversity even within a single

host [1, 2, 3]. This diversity is further compounded by the prevalence of segmented genomes in viral families like influenza, rotavirus, and bunyaviruses, where individual segments can evolve under distinct selective pressures and reassort, contributing to a complex landscape for genome reconstruction. While pipelines are often designed for a specific virus and their subtypes [4], accurate and complete viral genome reconstruction of samples with unknown references typically requires manual curation of contigs and reference matching [5, 6, 7]. This manual curation process is time-consuming, making it impractical for large-scale metagenome studies or rapid response scenarios that involve emerging viral outbreaks of unknown origin.

To address these limitations, we developed nf-core/viralmetagenome, a comprehensive pipeline specifically designed for untargeted viral genome reconstruction. The pipeline is developed using Nextflow [8] within the nf-core framework [9], ensuring reproducibility through containerization with Docker [10] and Singularity [11], and enabling portability across computational platforms such as local desktops, high-performance clusters and cloud environments.

2 Pipeline Description

Nf-core/viralmetagenome implements an automated workflow that performs de novo assembly, reference matching through sequence clustering, and iterative refinement by read mapping and consensus calling to reconstruct viral genomes without prior knowledge of the target sequences. The pipeline consists of five major analytical stages: read preprocessing, read metagenomic diversity assessment, contig assembly and scaffolding, iterative consensus refinement with variant analysis, and consensus quality control (Figure 1). The use of multiple tools at specific steps is described in Supplementary Table 1. Unless otherwise noted, these choices were made to accommodate user preferences. The full source code repository is available at https://github.com/nf-core/viralmetagenome.

3 Implementation

Nf-core/viralmetagenome requires nextflow and a container management system (Docker, Singularity, or Conda). The pipeline can be executed with minimal setup:

```
nextflow run nf-core/viralmetagenome \
  -profile docker \
  --input samplesheet.csv \
  --output results
```

Input data is provided through a sample sheet in CSV, TSV, YAML, or JSON format containing sample names and paths to FASTQ files. The pipeline supports both single-end and paired-end sequencing metagenomic data, and offers optional support for Unique Molecular Identifiers (UMIs) as well as optional merging of sequencing runs.

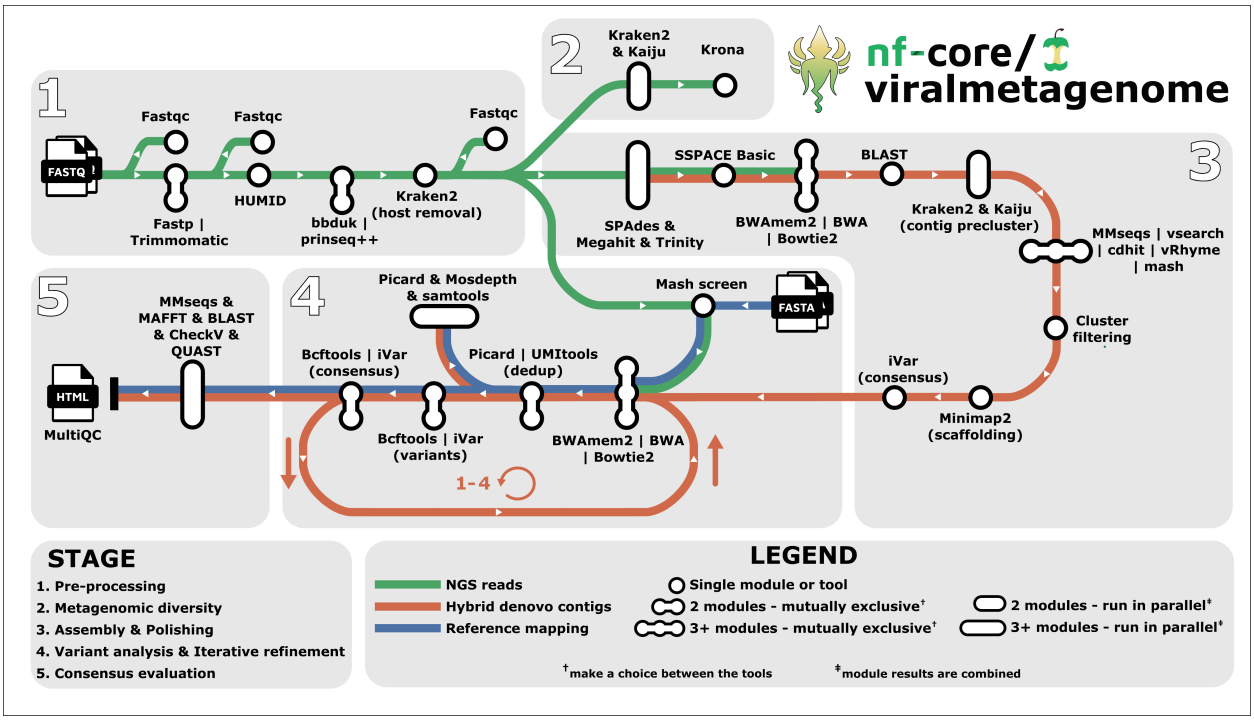

Figure 1: Visual overview of the nf-core/viralmetagenome pipeline for untargeted viral genome reconstruction. nf-core/viralmetagenome processes short FASTQ files through optionally read pre-processing (adapter removal, quality filtering, host removal), metagenomic diversity assessment, de novo assembly with multi-assembler support, scaffolding with automated reference identification and contig taxonomy-guided clustering, and iterative consensus refinement through read mapping and variant calling. Quality control metrics, assembly statistics, and coverage data are integrated into interactive MultiQC reports and standardised overview tables for downstream analysis.

3.1 Read preprocessing

The read preprocessing module performs quality control and filtering of raw sequencing reads. Initial quality assessment is conducted using FastQC before and after each processing step to monitor data quality throughout the workflow. Adapter trimming and read processing are performed using either Fastp [12] (default) or Trimmomatic [13]. Fastp is overall faster and has automated adapter detection and trimming [12]. For libraries prepared with UMIs, deduplication is implemented using HUMID [14] and with UMI-tools [15] once reads are mapped to a reference. If sequencing run merging is required, this can be done after adapter trimming and read-level deduplication by specifying a group in the input samplesheet. Complexity filtering, implemented through BBduk [16] or prinseq++ [17], removes low-complexity sequences containing repetitive elements that can lead to spurious alignments or misclassifications during downstream analysis. Host and contamination removal is performed using Kraken2 [18] against a user-specified host genome database. The default database contains a subset of the human genome. However, users are encouraged to employ more comprehensive databases, including complete host genome and transcriptome (human and otherwise), common sequencer contaminants, and bacterial genomes, to ensure thorough decontamination [19].

3.2 Metagenomic diversity assessment

Taxonomic classification of preprocessed reads is performed using two complementary approaches - Kaiju [20] and Kraken2 [18] - to maximise detection sensitivity across diverse viral families. Results from both classifiers are visualised using Krona [21].

3.3 *De novo* assembly and clustering

The assembly workflow implements a multi-assembler approach followed by clustering and scaffolding procedures. De novo assembly is performed using one or multiple assemblers: SPAdes [3] (configured for RNAviral mode by default), MEGAHIT [22], and Trinity [23]. This multi-assembler strategy capitalises on the distinct algorithmic strengths of each tool to maximise genome recovery across diverse viral families and variable read depths. Assembled contigs can be subjected to an optional extension step using SSPACE Basic [24].

Reference identification is conducted through BLASTn [25] searches against a comprehensive reference sequence pool, with the default being the latest clustered Reference Viral Database [26]. To facilitate identification of related genomic segments and appropriate reference sequences for contig scaffolding, the top five BLAST hits for each contig are retained and incorporated into the subsequent taxonomy-guided clustering step.

Taxonomy-guided clustering employs a two-stage process to cluster related contigs. Initial preclustering uses taxonomic assignments from both Kraken2 [18] and Kaiju [20]. For more efficient targeted analyses, the user can opt to focus on specific taxonomic clades. Subsequent nucleotide similarity clustering is performed using one of six available algorithms: CD-HIT-EST [27], VSEARCH [28], MMseqs2 [29], vRhyme [30], or Mash [31] with network-based community detection using the Leiden algorithm [32] or through single linkage. All tools are valid options, though performance

may vary depending on the dataset; for comprehensive benchmarking, we refer to Zielezinski et al. (2025) [33] and Steinegger and Söding (2017) [29].

As an optional filtering step of contig clusters, after assembly and extension, reads can be mapped to all contigs using BWAmem2 [34] (default), BWA [35], or Bowtie2 [36]. Clusters are filtered based on the cumulated percentage of reads mapped to the contigs of a cluster. By filtering clusters, low-coverage assemblies can be identified that likely represent assembly artefacts.

The final scaffolding step maps all cluster members to the cluster representative using Minimap2 [37], followed by consensus calling with iVar [38] to generate reference-assisted assemblies. Regions with zero coverage depth can optionally be represented by the reference genome to produce a more complete scaffold genome for consensus calling.

3.4 Iterative consensus refinement and variant calling

The consensus and variant calling module supports two distinct pathways: external reference-based analysis and scaffold refinement. In external reference-based analysis, users can provide reference genomes through the argument `-mapping_constraints`, which allows specifying a separate reference genome or reference set for each sample. When multiple genomes are provided for a single sample, the genome with the highest similarity is used as a reference using Mash [31]

Within the scaffold refinement, the pipeline can perform up to 4 cycles of iterative improvement (default 2) of the scaffolded de novo assembled contigs. Each iteration maps reads back to the current consensus using BWAmem2 [34], BWA [35], or Bowtie2 [36], followed by variant calling and consensus generation with BCFtools [39] or iVar [38]. Benchmarking by Bassano et al. [40] showed that BCFtools outperformed iVar in precision and recall. iVar tends to detect more low-frequency variants, which may increase false positives but also reduce false negatives. Users are recommended to consider prioritising sensitivity or specificity when selecting the variant caller.

Optional deduplication can be performed using Picard or when UMI's are available with UMI-tools [15]. Comprehensive mapping statistics are generated using samtools (flagstat, idxstats, stats) [39], Picard CollectMultipleMetrics [41], and coverage analysis with mosdepth [42].

3.5 Consensus Quality Control

4 Applications

4.1 Efficacy on simulated HIV read dataset

To evaluate the performance of nf-core/viralgenie, we simulated coinfection scenarios by mixing paired-end reads from public HIV-1 genomes with varying diversity (80-99% similarity), resulting in 13 samples (see supplementary table 1). nf-core/viralgenie successfully identified coinfections in all mixed samples when genetic similarity was low to moderate ($\leq$ 96.7% ANI). In contrast, for highly similar mixtures (98.7% ANI), the 2 original genomes were identified and reconstructed once out of 3 times.

We determined the influence of the reference used during scaffolding on the final consensus genome.

Here, we observed that the influence is small to negligible when the reference used closely matches ($\geq 90\%$) the original genome. However, if the reference is more distinct, the number of mismatches between final consensus genomes could increase up to 187 nucleotides (Figure **??**). This highlights the importance of appropriate reference selection for scaffolding or, in general, sequence alignment.

The hybrid consensus strategy implemented in nf-core/viralgenie combines de novo scaffolding with reference-guided consensus calling. Unlike traditional scaffolding that inserts ambiguous bases (Ns) in regions with no contig coverage, our method optionally fills these gaps with the corresponding reference sequence. This reference-filled scaffold serves as an improved template for the iterative refinement process. During subsequent read mapping and consensus calling cycles, any reference-filled regions that lack sufficient read coverage (below the minimum depth threshold) are explicitly replaced with ambiguous bases (Ns), ensuring that only evidence-supported positions are retained in the final consensus. Overall, this approach improved consensus completeness, reducing ambiguous bases by an average of 4 positions and increasing sequence identity by 0.08%. However, this improvement was not universal, with rare cases showing increased mismatches compared to traditional scaffolding approaches (Supplementary Figure X).

4.2 Validation on real-world datasets

To validate nf-core/viralgenie's performance on real-world datasets, we applied the pipeline to 28 publicly available metagenomic samples spanning several viral species. Here, the pipeline successfully generated high-quality or near-complete genomes for all species across different viral families, including both segmented viruses (Lassa virus and Orthonairovirus) and non-segmented viruses (SARS-CoV-2, West Nile virus, and Monkeypox virus).

The computational requirements for analyzing these 28 samples were modest, requiring 412 CPU hours and a maximum of 79GB RAM on an HPC system (excluding taxonomic classification steps). The automated reference-contig clustering strategy represents a considerable advancement over manual curation, significantly reducing processing time while maintaining accuracy. Our analysis demonstrates that pipeline performance is strongly influenced by reference database quality and comprehensiveness, with closer reference similarity yielding more complete and accurate consensus genomes. This underscores the importance of maintaining up-to-date databases such as the Reference Viral Database (Goodacre et al. 2018) for reference selection and Virosaurus (Gleizes et al. 2020) for contig annotation. The pipeline uses the Virosaurus-vertebrate database by default; users analyzing plant pathogens should switch to the Virosaurus plant database for optimal results. While nf-core/viralgenie is specifically designed for eukaryotic viruses, bacteriophage analysis requires different approaches and users should consider specialized pipelines such as VIRify (Rangel-Pineros et al. 2022), VIBRANT (Kieft et al. 2020), or VirSorter2 (Guo et al. 2021).

5 Conclusion

nf-core/viralgenie addresses a critical need in viral genomics by providing an automated, scalable solution for untargeted viral genome reconstruction. The pipeline successfully automates the traditionally manual and time-consuming process of viral genome assembly from metagenomic data through its integrated workflow of de novo contig assembly, automated reference selection, clustering

algorithms, and iterative refinement strategies.

Our validation demonstrates the pipeline's broad applicability across diverse eukaryotic viral families, achieving high-quality genome reconstruction while ensuring reproducibility and ease of deployment across different computational environments.

As viral surveillance and outbreak response increasingly rely on metagenomic sequencing, automated pipelines like viralgenie will be essential for timely pathogen strain identification. The pipeline represents a significant step forward in making viral genome reconstruction accessible to researchers without requiring extensive bioinformatics expertise, facilitating broader adoption of metagenomic approaches in viral research and public health applications.

Acknowledgments

We thank the nf-core community for their support and feedback during development.

Funding

This work was supported by [funding sources to be added].

Author Contributions

J.K. designed and implemented the pipeline, performed validation analyses, and wrote the manuscript. P.L. and L.K. supervised the project and provided critical feedback. All authors reviewed and approved the final manuscript.

Data Availability

The nf-core/viralgenie pipeline is freely available at https://github.com/nf-core/viralgenie. All test datasets and validation scripts are available in the project repository.

Conflict of Interest

The authors declare no competing interests.

References

[1] Jasmijn A Baaijens, Amal Zine El Aabidine, Eric Rivals, and Alexander Schönhuth. De novo assembly of viral quasispecies using overlap graphs. 27:835–848.

[2] Zhi-Luo Deng, Akshay Dhingra, Adrian Fritz, Jasper Götting, Philipp C Münch, Lars Steinbrück, Thomas F Schulz, Tina Ganzenmüller, and Alice C McHardy. Evaluating assembly and variant calling software for strain-resolved analysis of large DNA viruses. 22.

[3] Dmitry Meleshko, Iman Hajirasouliha, and Anton Korobeynikov. coronaSPAdes: from biosynthetic gene clusters to RNA viral assemblies. # RNAviral SPAdes: a assembler specific for RNA viruses. Solve problems of the bias RNA genomes have in evolution (biases from the reverse transcription and polymerase chain reaction):- Huge diversity - Quasi species- Presence of IsoformsMAIN GOAlL: to remove possible variation due to quasispecies, strain variation and sequencing artifactsThe solution they have for is to modify some steps and include a HMM profile to simplify their de bruijn graph.

[4] Samuel S Shepard, Sarah Meno, Justin Bahl, Malania M Wilson, John Barnes, and Elizabeth Neuhaus. Viral deep sequencing needs an adaptive approach: IRMA, the iterative refinement meta-assembler. 17:708.

[5] Christopher Tomkins-Tinch, Simon Ye, Hayden Metsky, Irwin Jungreis, Rachel Sealfon, Xiao Yang, Kristian Andersen, Michael Lin, and Daniel Park. broadinstitute/viral-ngs: v1.14.0.

[6] Jutte J C de Vries, Julianne R Brown, Nicole Fischer, Igor A Sidorov, Sofia Morfopoulou, Jiabin Huang, Bas B Oude Munnink, Arzu Sayiner, Alihan Bulgurcu, Christophe Rodriguez, Guillaume Gricourt, Els Keyaerts, Leen Beller, Claudia Bachofen, Jakub Kubacki, Cordey Samuel, Laubscher Florian, Schmitz Dennis, Martin Beer, Dirk Hoeper, Michael Huber, Verena Kufner, Maryam Zaheri, Aitana Lebrand, Anna Papa, Sander van Boheemen, Aloys C M Kroes, Judith Breuer, F Xavier Lopez-Labrador, and Eric C J Claas. Benchmark of thirteen bioinformatic pipelines for metagenomic virus diagnostics using datasets from clinical samples. 141:104908.

[7] Yiqiao Li, Mariana Polychronopoulou, Ine Boonen, Antonios Fikatas, Sophie Gryseels, Anne Laudisoit, Joelle Gouy de Bellocq, Bram Vrancken, Gkikas Magiorkinis, Philippe Lemey, and Magda Bletsa. Evaluation of metatranscriptomic sequencing protocols to obtain full-length RNA virus genomes from mammalian tissues. 20:e0324537.

[8] Paolo Di Tommaso, Maria Chatzou, Evan W Floden, Pablo Prieto Barja, Emilio Palumbo, and Cedric Notredame. Nextflow enables reproducible computational workflows. 35:316–319.

[9] Philip A Ewels, Alexander Peltzer, Sven Fillinger, Harshil Patel, Johannes Alneberg, Andreas Wilm, Maxime Ulysse Garcia, Paolo Di Tommaso, and Sven Nahnsen. The nf-core framework for community-curated bioinformatics pipelines. 38:276–278.

[10] D Merkel. Docker: lightweight linux containers for consistent development and deployment. 2014:2.

[11] Gregory M Kurtzer, Vanessa Sochat, and Michael W Bauer. Singularity: Scientific containers for mobility of compute. 12:e0177459.

[12] Shifu Chen, Yanqing Zhou, Yaru Chen, and Jia Gu. fastp: an ultra-fast all-in-one FASTQ preprocessor. 34:i884–i890.

[13] Anthony M Bolger, Marc Lohse, and Bjoern Usadel. Trimmomatic: a flexible trimmer for illumina sequence data. 30:2114–2120.

[14] Jeroen F J Laros. HUMID: HUMID: reference free FastQ deduplication.

[15] Tom Smith, Andreas Heger, and Ian Sudbery. UMI-tools: modeling sequencing errors in unique molecular identifiers to improve quantification accuracy. 27:491–499.

[16] Brian Bushnell. BBMap.

[17] Vito Adrian Cantu, Jeffrey Sadural, and Robert Edwards. PRINSEQ++, a multi-threaded tool for fast and efficient quality control and preprocessing of sequencing datasets.

[18] Derrick E Wood, Jennifer Lu, and Ben Langmead. Improved metagenomic analysis with kraken 2. 20:257.

[19] Matthew Forbes, Duncan Y K Ng, Roisin M Boggan, Andrea Frick-Kretschmer, Jillian Durham, Oliver Lorenz, Bruhad Dave, Florent Lassalle, Carol Scott, Josef Wagner, Adrianne Lignes, Fernanda Noaves, David K Jackson, Kevin Howe, and Ewan Harrison. Benchmarking of human read removal strategies for viral and microbial metagenomics. page 2025.03.21.644587.

[20] Peter Menzel, Kim Lee Ng, and Anders Krogh. Fast and sensitive taxonomic classification for metagenomics with kaiju. 7:11257.

[21] Brian D Ondov, Nicholas H Bergman, and Adam M Phillippy. Interactive metagenomic visualization in a web browser. 12:385.

[22] Dinghua Li, Ruibang Luo, Chi-Man Liu, Chi-Ming Leung, Hing-Fung Ting, Kunihiko Sadakane, Hiroshi Yamashita, and Tak-Wah Lam. MEGAHIT v1.0: A fast and scalable metagenome assembler driven by advanced methodologies and community practices. 102:3–11.

[23] Manfred G Grabherr, Brian J Haas, Moran Yassour, Joshua Z Levin, Dawn A Thompson, Ido Amit, Xian Adiconis, Lin Fan, Raktima Raychowdhury, Qiandong Zeng, Zehua Chen, Evan Mauceli, Nir Hacohen, Andreas Gnirke, Nicholas Rhind, Federica di Palma, Bruce W Birren, Chad Nusbaum, Kerstin Lindblad-Toh, Nir Friedman, and Aviv Regev. Full-length transcriptome assembly from RNA-seq data without a reference genome. 29:644–652.

[24] Marten Boetzer, Christiaan V Henkel, Hans J Jansen, Derek Butler, and Walter Pirovano. Scaffolding pre-assembled contigs using SSPACE. 27:578–579.

[25] S F Altschul, W Gish, W Miller, E W Myers, and D J Lipman. Basic local alignment search tool. 215:403–410.

[26] Norman Goodacre, Aisha Aljanahi, Subhiksha Nandakumar, Mike Mikailov, and Arifa S Khan. A reference viral database (RVDB) to enhance bioinformatics analysis of high-throughput sequencing for novel virus detection. 3.

[27] Weizhong Li and Adam Godzik. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. 22:1658–1659.

[28] Torbjørn Rognes, Tomáš Flouri, Ben Nichols, Christopher Quince, and Frédéric Mahé. VSEARCH: a versatile open source tool for metagenomics. 4:e2584.

[29] Martin Steinegger and Johannes Söding. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. 35:1026–1028.

[30] Kristopher Kieft, Alyssa Adams, Rauf Salamzade, Lindsay Kalan, and Karthik Anantharaman. vRhyme enables binning of viral genomes from metagenomes. 50:e83.

[31] Brian D Ondov, Gabriel J Starrett, Anna Sappington, Aleksandra Kostic, Sergey Koren, Christopher B Buck, and Adam M Phillippy. Mash screen: high-throughput sequence containment estimation for genome discovery. 20:232.

[32] V A Traag, L Waltman, and N J van Eck. From louvain to leiden: guaranteeing well-connected communities. 9:5233.

[33] Andrzej Zielezinski, Adam Gudyś, Jakub Barylski, Krzysztof Siminski, Piotr Rozwalak, Bas E Dutilh, and Sebastian Deorowicz. Ultrafast and accurate sequence alignment and clustering of viral genomes. 22:1191–1194.

[34] Md Vasimuddin, Sanchit Misra, Heng Li, and Srinivas Aluru. Efficient architecture-aware acceleration of BWA-MEM for multicore systems. In *2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 314–324. IEEE.

[35] Heng Li. Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM.

[36] Ben Langmead, Christopher Wilks, Valentin Antonescu, and Rone Charles. Scaling read aligners to hundreds of threads on general-purpose processors. 35:421–432.

[37] Heng Li. Minimap2: pairwise alignment for nucleotide sequences. 34:3094–3100.

[38] Nathan D Grubaugh, Karthik Gangavarapu, Joshua Quick, Nathaniel L Matteson, Jaqueline Goes De Jesus, Bradley J Main, Amanda L Tan, Lauren M Paul, Doug E Brackney, Saran Grewal, Nikos Gurfield, Koen K A Van Rompay, Sharon Isern, Scott F Michael, Lark L Coffey, Nicholas J Loman, and Kristian G Andersen. An amplicon-based sequencing framework for accurately measuring intrahost virus diversity using PrimalSeq and iVar. 20:8.

[39] Petr Danecek, James K Bonfield, Jennifer Liddle, John Marshall, Valeriu Ohan, Martin O Pollard, Andrew Whitwham, Thomas Keane, Shane A McCarthy, Robert M Davies, and Heng Li. Twelve years of SAMtools and BCFtools. 10.

[40] Irene Bassano, Vinoy K Ramachandran, Mohammad S Khalifa, Chris J Lilley, Mathew R Brown, Ronny van Aerle, Hubert Denise, William Rowe, Airey George, Edward Cairns, Claudia Wierzbicki, Natalie D Pickwell, Myles Wilson, Matthew Carlile, Nadine Holmes, Alexander Payne, Matthew Loose, Terry A Burke, Steve Paterson, Matthew J Wade, and Jasmine M S Grimsley. Evaluation of variant calling algorithms for wastewater-based epidemiology using mixed populations of SARS-CoV-2 variants in synthetic and wastewater samples.

[41] Broad Institute. Picard toolkit.

[42] Brent S Pedersen and Aaron R Quinlan. Mosdepth: quick coverage calculation for genomes and exomes. 34:867–868.