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 (Baaijens et al. 2017; Deng et al. 2021; Meleshko, Hajirasouliha and Korobeynikov 2021). 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 (Shepard et al. 2016), accurate and complete viral genome reconstruction of samples with unknown reference typically requires manual curation of contigs and reference matching (Tomkins-Tinch et al. 2017; de Vries et al. 2021; Li et al. 2025). 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/viralgenie, a comprehensive pipeline specifically designed for untargeted viral genome reconstruction. The pipeline is developed using Nextflow (Di Tommaso et al. 2017) within the nf-core framework (Ewels et al. 2020), ensuring reproducibility through containerization with Docker (Merkel 2014) and Singularity (Kurtzer, Sochat and Bauer 2017), and enabling portability across computational platforms such as local desktops, high-performance clusters and cloud environments.

2 Pipeline Description

Nf-core/viralgenie 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. 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/viralgenie.

3 Implementation

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

```
nextflow run nf-core/viralgenie \
   -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.

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 (Chen et al. 2018) (default) or Trimmomatic (Bolger, Lohse and Usadel 2014). For libraries prepared with UMIs, deduplication is implemented using HUMID (Laros) and with UMI-tools (Smith, Heger and Sudbery 2017) 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 (Bushnell) or prinseq++ (Cantu), 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 (Wood, Lu and Langmead 2019) 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 (Forbes et al. 2025).

3.2 Metagenomic diversity assessment

Taxonomic classification of preprocessed reads is performed using two complementary approaches - Kaiju (Menzel, Ng and Krogh 2016) and Kraken2 (Wood, Lu and Langmead 2019) - to maximise detection sensitivity across diverse viral families. Results from both classifiers are visualised using Krona (Ondov, Bergman and Phillippy 2011).

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 (Meleshko, Hajirasouliha and Korobeynikov 2021) (configured for RNAviral mode by default), MEGAHIT (Li et al. 2016), and Trinity (Grabherr et al. 2011). 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 (Boetzer et al. 2011).

Reference identification is conducted through BLASTn (Altschul et al. 1990) searches against a comprehensive reference sequence pool, with the default being the latest clustered Reference Viral Database (RVDB) (Goodacre et al. 2018). 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 pre-clustering uses taxonomic assignments from both Kraken2 (Wood, Lu and Langmead 2019) and Kaiju (Menzel, Ng and Krogh 2016). To accelerate the analysis, specified taxonomic clades and their descendant clades can be excluded. Subsequent nucleotide similarity clustering is performed using one of six available algorithms: CD-HIT-EST (Li and Godzik 2006), VSEARCH (Rognes et al.

2016), MMseqs-linclust (Steinegger and Söding 2017), MMseqs-cluster (Steinegger and Söding 2017), vRhyme (Kieft et al. 2022), or Mash (Ondov et al. 2019) with network-based community detection using the Leiden algorithm (Traag, Waltman and van Eck 2019) or through connected components. As such, the choice of clustering method can be tailored to specific dataset characteristics.

As an optional filtering step of contig clusters, after assembly and extension, reads are mapped towards all contigs using BWAmem2 (Vasimuddin et al. 2019) (default), BWA (Li 2013), or Bowtie2 (Langmead et al. 2019). Clusters are filtered based on the cumulated percentage of reads mapping towards the contigs of a cluster. By filtering clusters, we identify low-coverage assemblies that likely represent assembly artefacts.

The final scaffolding step maps all cluster members to the cluster representative using Minimap2 (Li 2018), followed by consensus calling with iVar (Grubaugh et al. 2019) to generate reference-assisted assemblies. Regions with zero coverage depth can optionally be called using the reference genome to produce an optimal 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`, where, for each sample individually, a reference genome or reference set can be specified. When multiple genomes are given for a single sample, the genome with the highest similarity is used as a reference using Mash (Ondov et al. 2019).

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 (Vasimuddin et al. 2019), BWA (Li 2013), or Bowtie2 (Langmead et al. 2019), followed by variant calling and consensus generation with BCFtools (Danecek et al. 2021) or iVar (Grubaugh et al. 2019). Optional deduplication can be performed using Picard or when UMI's are available with UMI-tools (Smith et al. 2017). Comprehensive mapping statistics are generated using samtools (flagstat, idxstats, stats) (Danecek et al. 2021), Picard CollectMultipleMetrics (Broad Institute), and coverage analysis with mosdepth (Pedersen and Quinlan 2018).

3.5 Consensus Quality Control

Comprehensive quality assessment of reconstructed viral genomes is performed through multiple complementary analyses. CheckV (Nayfach et al. 2021) estimates genome completeness and contamination. Consensus genomes undergo similarity analysis through BLASTn (Altschul et al. 1990) searches against the reference pool, and MMseqs (Steinegger and Söding 2017) searches against comprehensive annotation databases such as Virosaurus (Gleizes et al. 2020), enabling species identification, segment designation, host associations, and any other additional customised metadata embedded in the database.

Multiple sequence alignment using MAFFT (Katoh et al. 2002) aligns the final consensus genomes with the de novo contigs, the consensus genomes during iterations and the reference used for scaffolding, enabling assessment of assembly accuracy. Quality control metrics are integrated into interactive MultiQC reports (Ewels et al. 2016), providing a comprehensive visualisation of the pipeline results. Standalone overview tables are generated by extracting key metrics from MultiQC into dataframes, facilitating downstream analysis of all samples.

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 x). 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