Subject Section

Panalyze: automated virus pangenome variation graph construction and analysis

Chandana Tennakoon[1,*], Thibaut Freville[1] and Tim Downing[1,*]

[1]Pirbright Institute, Surrey, UK

*To whom correspondence should be addressed

Abstract

Motivation: Constructing and studying reference-free pangenome variation graphs (PVG) can support new insights into viral genomic diversity. Interpreting the information arising from this is challenging, so automating these processes to allow exploratory investigations for optimisation is essential. To address this , we developed an easily deployable pipeline to facilitate the rapid creation of virus pangenome variation graphs that applies a broad range of analyses to these graphs.

Results: We present panalyze, a scalable and unbiased virus pangenome variation graph construction and analysis tool implemented in NextFlow and containerised in Docker. Panalyze avails of NextFlow to efficiently complete tasks across multiple compute nodes and in diverse High-Performance Computing (HPC) environments. We illustrate how Panalyze works and the valuable outputs it can generate using a range of common viral pathogens

Availability: Panalyze is released under a MIT open-source license, available on GitHub with documentation accessible at https://github.com/downingtim/LSDV-PVG/ .

Contact: Chandana.tennakoon@pirbright.ac.uk, tim.downing@pibright.ac.uk

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

1 Introduction

Pangenome variation graphs (PVGs) offer new insights into genome evolution over short and long scales. A PVG is a sequence graph whose nodes represent sequences and whose edges represent connections observed in data (Hein 1989) such that a genome can be decoded by traversing a series of paths (Garrison et al 2018). PVGs combat reference allele bias by allowing reads to map to the most similar combination of paths in a PVG, instead of a single haplotype (Paten et al 2018). Reference bias is not obviated by using a consensus reference sequence or by using a multi-sample reference (Chen et al 2021, Chen et al 2024). This affects the chance of a read mapping to a location, impacting mutation detection, phylogenetic analyses, and associated inferences in pathogens (Valiente-Mulor 2017) particularly at variable regions that are often strongly associated with phenotypic changes (Haga et al 2024). This results in unaligned reads, missed mutations, poorer inferences, and less precise evolutionary models (Baaijens et al 2017, Hickey et al 2020). Novel pathogen isolates genetically distinct from known references will be characterised less accurately (Boehm et al 2021, Garrison 2023). In viruses, although genome assembly can be effective, extreme AT/GC content, host/vector contamination, short output contig lengths, and repetitiveness all contribute to incomplete and biased genomes (Alser et al 2021, Lischer et al 2017). Moreover, contigs often need to be aligned to existing databases either as sequences or k-mers, or an existing unavoidably-biased scaffold may be used as a reference (Bradley et al 2019, Ondov et al 2016).

Little work on viral PVGs has been completed (Downing 2024): only one study has yet been published: in this, PVGs helped recover >8% of sequence that failed to align to known linear reference genomes (Duchen et al 2024). A PVG can efficiently represent the diversity of multiple viral strains and so act as a better reference structure than a single linear reference. This means reads can map to different parts of haplotypes represented in the PVG, which reduces the effect of known mutations and improves mutation discovery (Paten et al 2017, Rakocevic et al 2018). This circumvents the issue of reference genetic distance, improves alignment quality, and may permit more accurate genetic reconstruction of the underlying consensus sequence (Garrison et al 2023, Moshiri et al 2022). Thus, there is a need for PVG creation and analysis to explore parameter spaces and optimise processes for viral PVGs.

There are many tools available for PVG creation and analysis. However, these tools can be hard to install due to complex installation procedures, assumption of installation dependencies, and some steps requiring root access. It would be beneficial to have a curated set of pre-installed tools that can be easily deployed to automate PVG construction and analysis.

To address these issues, we present Panalyze, a scalable and rapid virus PVG construction and analysis tool operating in NextFlow v24.10.3 and containerised in Docker for broad computer server application. Panalyze wraps a series of tools to allow sequence retrieval, PVG creation, phylogenetic analysis, PVG visualisation using multiple methods, PVG size calculation, PVG feature summarisation using different tools, PVG openness analysis with three approaches, VCF generation, BUSCO gene detection, and PVG community detection. Advantageously, viral genomes can be interpreted as individual haplotypes in small PVGs that do not need pruning. Constructing PVGs for viruses will reduce reference genome bias and facilitate more accurate characterisation of the genetic variability when mapping reads to a PVG. Panalyze runs in Docker containers deployed via NextFlow. It can use cluster nodes efficiently, and resolves the issue of complex installations and dependencies. We illustrate how Panalyze can work effectively for DNA and RNA viruses, including segmented ones.

2 Methods

2.1 Pipeline overview

Panalyze has two entry points: a pre-existing FASTA file, or a search query from the user (Figure 1A). The latter seeks all complete genomes or genomic sequences matching the query in the Nucleotide database (Sayers et al 2021). The sequences are aligned with Mafft v7.453 (Katoh & Standley 2013) using automatic optimisation and default parameters. Evolutionary relationships are reconstructed using RAxML-NG (Randomised Axelerated Maximum Likelihood) v1.2.0 (Kozlov et al 2019) with a GTR (general time reversible) model and gamma substitution rate heterogeneity. Other models can be selected with modeltest-ng (Darriba et al 2020). These phylogenies are mid-pointed rooted and visualised using R v4.3.2 (R Core Team 2025) packages ape v5.7-1 (Paradis & Schliep 2019), ggtree v3.8.2 (Yu et al 2017), phangorn v2.11.1 (Schliep 2011), Rcpp v1.0.11 (Eddelbuettel et al 2024b), RcppArmadillo v0.12.6.6.0 (Eddelbuettel et al 2024a), phytools v2.0-3 (Revell 2024) and treeio v1.24.3 (Wang et al 2020).

Panalyze creates PVGs using the pangenome graph builder (PGGB) v0.5.4 pipeline (Garrison et al 2024), which initiates all-to-all alignment with WFMASH v0.7.0 (Guarracino et al 2021) subject to a threshold of 90% identity per Kb (Figure 1B). PGGB keeps all paths and was used to reconstruct the input haplotypes directly. PGGB creates the PVG using SEQWISH (Garrison & Guarracino 2023) using a k-mer of 19 bp and a window size of 27 bp, and then sorts and orders it to produce a progressively linearised PVG based on partial order alignment with SMOOTHXG v0.7.0.0. This also removes erroneous redundant nodes (Garrison & Guarracino 2023). Multiqc v1.14 (Ewels et al 2016) collates PGGB metrics. The PVGs were generated in Graphical Fragment Assembly (GFA) format version 1. Panalyze does not use PVG pruning because virus PVGs are comparatively small. In the same way, no PVG chunking or lacing is completed. These PVGs are converted with the convert function in variation graph (VG) v1.43.0 (Garrison et al 2018), indexed with VG's autoindex function. The corresponding FASTA file is indexed with SAMtools (Li 2009).

A range of analyses are applied. GFASTATS v1.3.6 (Formenti 2022), and ODGI (optimized dynamic genome/graph implementation) v0.8.3 (Guarracino et al 2022) produce PVG summary statistics. Visualisations are created by ODGI (Guarracino et al 2022) (Figure 1) and PGGB (Garrison et al 2023). The number of bases in the PVG and the presence-absence variation (PAV) comes from ODGI (Guarracino et al 2022): this is visualised with ggplot2 v2_3.4.4 (Wickham 2016). A VCF is produced by GFAUTIL v0.3.2 (Fischer 2021). The relative differences between the sequences' coordinate systems is computed by GFAUTIL and visualised with ggplot2 v2_3.4.4 (Wickham 2016). PVG openness is computed firstly by Panacus v0.2.3 (Parmigiani et al 2024) using the nodes in the PVG with default coverage thresholds that included all segments, secondly with k-mers using Pangrowth (Parmigiani et al 2024) and thirdly with sequences extracted from PVG nodes using ODGI's heaps function using 1,000 simulations (Guarracino et al 2022). PVG communities can be created by PGGB (Garrison et al 2023) where the number of clusters can be determined by the user and the file names are indexed using fastix. This PVG community data is parsed, visualised and pairwise alignment format (PAF) summary files are generated for input to PAFGNOSTIC to numerically summarise the properties. Optional analyses include determining key gene presence-absence via BUSCO (Manni et al 2021). Additionally, the presence-absence variation (PAV) can be determined by ODGI (Guarracino et al 2022) and visualised, but should only be used for smaller or less diverse datasets.

2.2 Testing the workflow

To illustrate that Panalyze's PVG construction and analysis approach was broadly applicable to all virus genomes, a range of test datasets were analysed (Table 1). Firstly, we applied it to a set of 121 curated lumpy skin disease virus (LSDV) genomes, representing large DNA viruses. This genetically homogeneous dataset took 33 CPU hours (605 minutes) to complete. Secondly, we evaluated Panalyze's performance on 142 foot-and-mouth disease virus (FMDV) serotype A genomes, representing small ssRNA viruses, which completed in 14 CPU hours (four minutes). We repeated this with 188 FMDV serotype A genomes (23 CPU hours, 25 minutes), and with 441 serotype O genomes (26 CPU hours, 31 minutes). Thirdly, we evaluated its effectiveness in interpreting a segmented genome, Rift Valley fever virus (RVFV), an ssRNA virus whose genome has three segments (S in 15 CPU hours / 7.1 minutes, M in 18 CPU hours / 10.3 minutes, L in 22 CPU hours / 14.8 minutes). Fourthly, we demonstrate the scalability of Panalyze by applying it to 2,358 mpox genomes in 23 CPU hours (249 minutes).

Table 1. Properties of virus PVGs.

Virus	Samples	Average size (bp)	Total length (bp)	Nodes	Edges	Openness
LSDV	121	150,641	156,966	7,203	9,827	1.88
MPOX	2,358	27,730	32,086	7,559	10,233	0.54
PRCV	15	7,347	40,305	1,286	1,722	1.13
FMDV O	441	8,125	16,888	14,062	22,988	1.15
FMDV A	188	8,121	21,433	14,877	24,287	1.12
FMDV C	18	8,142	10,779	5,959	8,359	1.35
RVFV S	414	1,690	2,739	1,738	2,495	0.97
RVFV M	302	3,885	6,182	3,639	5,091	0.93
RVFV L	306	6,404	9,092	5,527	7,676	0.98

Average size refers to the lengths of the input genomes, and the total length refers to the sum of the sequence lengths of the nodes in the PVG. Openness was estimated from alpha values from Pangrowth.

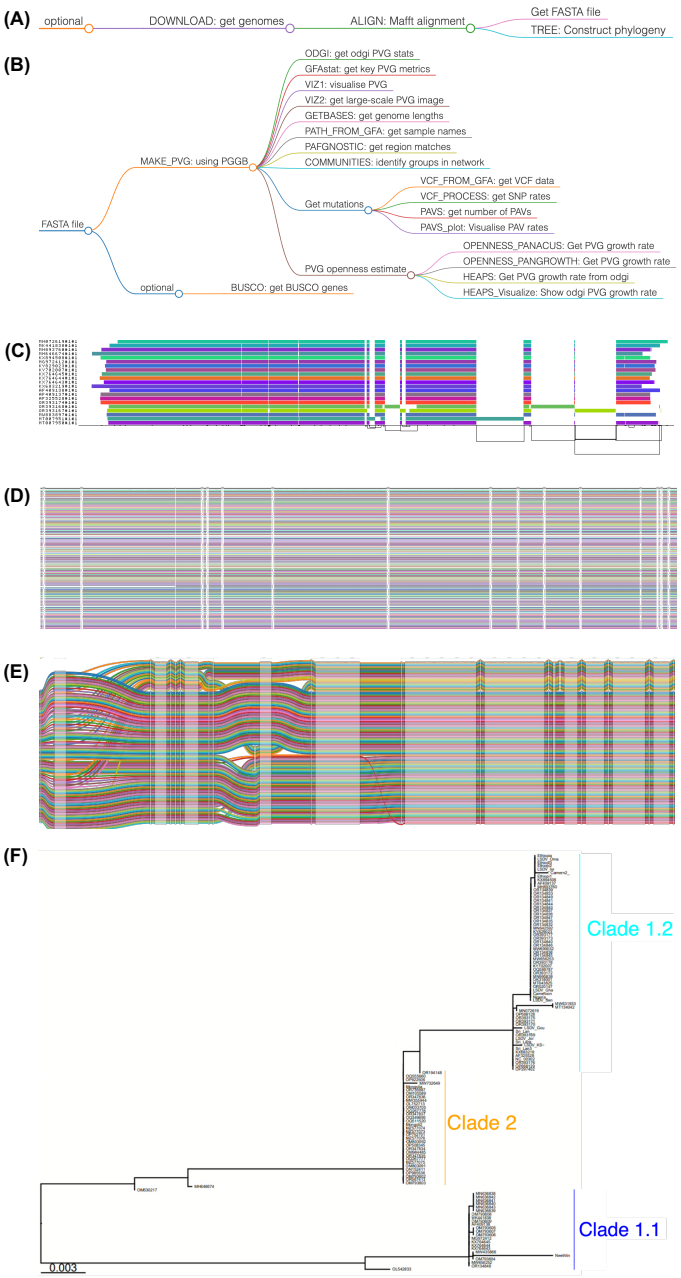


Fig. 1. A representation of Panalyze's workflow processes (right)**.** (A) Optionally, users can retrieve sequences of interest via the DOWNLOAD process to create FASTA input (B) for MAKE_PVG to make the PVG, before running a range of analyses, including: PVG metrics with the ODGI and GFAstat processes, simple and complex visualisations with VIZ1 and VIZ2, genome lengths with GETBASES, creating a pairwise alignment file (PAF) with PAFGNOSTIC, and community detection with COMMUNITIES. Other key processes centre around mutation detection: determining VCFs with VCF_FROM_GFA, VCF_PROCESS to infer mutation rates, getting PAVS with PAVS, and visualising the PAVs with PAVS_plot. In addition, users can implement openness analyses using Panacus, Pangrowth and ODGI using OPENNESS_PANGROWTH, OPENNESS_PANACUS, HEAPS and HEAPS_Vizualize. Users can (optionally) identify Busco genes with BUSCO. (C) 21 LSDV genome graphs from a PVG at 135-140 Kb visualised using viz in ODGI (Optimized Dynamic Genome Graph Implementation visualization) where each is a coloured bar and the PVG topology is shown along the genomic coordinates the x-axis (black lines). (D) 121 LSDV genome graphs at 136-138 Kb visualised with viz in VG (Variation Graph). (E) The same genomes at 5,553-5,600 bp visualised using SequenceTubeMap, showing one minor (22 samples from Clade 1) and one major (99 samples from Clades 1.2 and 2) extended haplotype at LD144 (encoding a kelch-like protein at 135,541-137,208 bp) and LD145 (encoding an ankyrin repeat protein at 137,232-139,136). (F) A phylogeny based on this region showed two main divergent haplotypes.

3 Results

3.1 Building virus pangenome variation graphs

We created PVGs for large DNA (LSDV, mpox), small ssRNA (FMDV) and segmented RNA (RVFV) viruses using Panalyze to illustrate its broad applicability to different virus types, as well as levels of diversity, sequence sizes and sample sizes (Table 1). The varied relative rate of nodes and edges compared to the sample sizes and average genome lengths illustrate how different the levels of sequence variability were. To illustrate the unusual patterns of variation in certain viral genomes, we extracted a portion of the LSDV PVG at which there are 15 mutations spanning two genes (Figure 1E). Phylogenetic reconstruction shows divergent haplotypes: selecting a single representative reference sequence for this region could bias downstream analyses.

We modelled the rate of discovery of new mutations as a function of the number of samples added to the PVGs, estimated with Pangrowth's calculation of alpha. We observed that RVFV's segments had a slightly open PVG, whereas PRCV and the FMDV serotypes had slightly closed ones, which contrasted with the closed LSDV PVG and the open mpox PVG (Table 1). Similar values were observed with Panacus. Our estimate of gamma for LSDV's PVG (Panacus 0.04, Pangrowth 0.02) was comparable to a previous estimate of 0.05 derived from a gene-based pangenome (Xie et al 2024).

4 Discussion

We present Panalyze, a scalable, portable, adaptable pipeline for virus PVG construction and analysis. It runs in NextFlow inside Docker containers and is designed in a modular fashion. This allows Nextflow's process management to improve thread allocation and accelerate task completion. Panalyze is written in domain-specific language (DSL) 2 and wraps a range of existing PVG tools to interpret PVG rapidly to support exploration and optimisation. The selection of tools to run can be specified using a configuration file. Panalyze generated PVGs using PGGB and then runs a series of tasks to interpret this PVG using numerical, visual, graphical and other summaries for users to examine. This automation facilitates quicker optimisation and deeper engagement with complex datasets. We demonstrate here how it can be applied to diverse specimens, including large DNA viruses, small ssRNA viruses, and segmented ones too. This means it generates less biased representations of sequence diversity in a reliable, rapid and efficient manner.

Although there are emerging tools for PVG construction and analysis, no tool has yet bridged these nor automated them in Nextflow. Panalyze addresses this in a user-oriented reproducible manner. We hope that Panalyze can help the virus research community progress from using linear reference genomes to more accurate PVGs generated from more representative datasets. Moreover, we expect that superior PVG methods will underpin better novel or recombinant sample characterisation as well as routine PVG analysis and mutation detection. Consequently, Panalyze supports virus evolutionary and epidemiological genomics.

Acknowledgements

We would like to acknowledge comments and ideas from Dr. Caroline Wright and Dr. Lidia Dykes (both Pirbright Institute). We would like to acknowledge support from SIBLI, University of the Philippines Baguio, Philippines.

Funding

We would like to acknowledge the Pirbright Institute's Bioinformatics STP and funding from UK Research and Innovation (UKRI) Biotechnology and Biological Sciences Research Council (BBSRC) grants BBS/E/PI/230002A, BBS/E/PI/230002B, BBS/E/PI/230002C and BBS/E/PI/23NB0003.

Conflicts of Interest: none declared.

References

Alser M, et al. Technology dictates algorithms: recent developments in read alignment. Genome Biol. 2021 22(1):249. doi: 10.1186/s13059-021-02443-7

Baaijens JA, Aabidine AZE, Rivals E, Schönhuth A. De novo assembly of viral quasispecies using overlap graphs. Genome Res. 2017 27(5):835-848. doi: 10.1101/gr.215038.116

Boehm E, Kronig I, Neher RA, Eckerle I, Vetter P, Kaiser L; Geneva Centre for Emerging Viral Diseases. Novel SARS-CoV-2 variants: the pandemics within the pandemic. Clin Microbiol Infect. 2021 27(8):1109-1117. doi: 10.1016/j.cmi.2021.05.022

Bradley P, den Bakker HC, Rocha EPC, McVean G, Iqbal Z. Ultrafast search of all deposited bacterial and viral genomic data. Nat Biotechnol. 2019 37(2):152-159. doi: 10.1038/s41587-018-0010-1

Chen NC, Paulin LF, Sedlazeck FJ, Koren S, Phillippy AM, Langmead B. Improved sequence mapping using a complete reference genome and lift-over. Nat Methods. 2024 21(1):41-49. doi: 10.1038/s41592-023-02069-6.

Chen NC, Solomon B, Mun T, Iyer S, Langmead B. Reference flow: reducing reference bias using multiple population genomes. Genome Biol. 2021 22(1):8 doi: 10.1186/s13059-020-02229-3.

Darriba D, Posada D, Kozlov AM, Stamatakis A, Morel B, Flouri T. ModelTest-NG: A New and Scalable Tool for the Selection of DNA and Protein Evolutionary Models. Mol Biol Evol. 2020 37(1):291-294. doi: 10.1093/molbev/msz189.

Downing T. Approaches to studying virus pangenome variation graphs. ArXiv 2024 https://doi.org/10.48550/arXiv.2412.05096

Duchen D, Clipman S, Vergara C, Thio CL, Thomas DL, Duggal P, Wojcik GL. A hepatitis B virus (HBV) sequence variation graph improves sequence alignment and sample-specific consensus sequence construction for genetic analysis of HBV. PLoS One. 2024 19(4):e0301069. doi: 10.1371/journal.pone.0301069

Eddelbuettel D, Balamuta JB. Extending R with C++: A Brief Introduction to Rcpp. 2018 https://doi.org/10.1080/00031305.2017.1375990

Eddelbuettel D, Sanderson C. RcppArmadillo: Accelerating R with high-performance C++ linear algebra. 2014. Computational Statistics and Data Analysis 10.1016/j.csda.2013.02.005

Ewels P, Magnusson M, Lundin S, Käller M. MultiQC: summarize analysis results for multiple tools and samples in a single report. Bioinformatics. 2016 32(19):3047-8. doi: 10.1093/bioinformatics/btw354.

Formenti G, Abueg L, Brajuka A, Brajuka N, Gallardo-Alba C, Giani A, Fedrigo O, Jarvis ED. Gfastats: conversion, evaluation and manipulation of genome sequences using assembly graphs. Bioinformatics. 2022 38(17):4214-4216. doi: 10.1093/bioinformatics/btac460

Garrison E, et al Variation graph toolkit improves read mapping by representing genetic variation in the reference. Nat Biotechnol. 2018 36(9):875-879. doi: 10.1038/nbt.4227

Garrison E, et al. Building pangenome graphs. Nat Methods. 2024 21(11):2008-2012. doi: 10.1038/s41592-024-02430-3

Garrison E, Guarracino A. Unbiased pangenome graphs. Bioinformatics. 2023 39(1):btac743. doi: 10.1093/bioinformatics/btac743

Guarracino A, Heumos S, Nahnsen S, Prins P, Garrison E. ODGI: understanding pangenome graphs. Bioinformatics. 2022 38(13):3319–26. doi: 10.1093/bioinformatics/btac308.

Guarracino A, Mwaniki N, Marco-Sola S, Garrison E. wfmash: whole-chromosome pairwise alignment using the hierarchical wavefront algorithm. 2021 https://github.com/ekg/wfmash.

Haga IR, et al. Sequencing and Analysis of Lumpy Skin Disease Virus Whole Genomes Reveals a New Viral Subgroup in West and Central Africa. Viruses. 2024 16(4):557. doi: 10.3390/v16040557

Hein J. A new method that simultaneously aligns and reconstructs ancestral sequences for any number of homologous sequences, when the phylogeny is given. Mol Biol Evol. 1989 6(6):649-68. doi: 10.1093/oxfordjournals.molbev.a040577.

Hickey G, Heller D, Monlong J, Sibbesen JA, Sirén J, Eizenga J, Dawson ET, Garrison E, Novak AM, Paten B. Genotyping structural variants in pangenome graphs using the vg toolkit. Genome Biol. 2020 21(1):35. doi: 10.1186/s13059-020-1941-7

Katoh K, Standley DM. MAFFT: iterative refinement and additional methods. Methods Mol Biol. 2014 1079:131-46. doi: 10.1007/978-1-62703-646-7_8.

Kozlov AM, Darriba D, Flouri T, Morel B, Stamatakis A. RAxML-NG: a fast, scalable and user-friendly tool for maximum likelihood phylogenetic inference. Bioinformatics. 2019 35(21):4453-4455. doi: 10.1093/bioinformatics/btz305.

Li H, Handsaker B, Wysoker A, Fennell T, Ruan J, Homer N, Marth G, Abecasis G, Durbin R; 1000 Genome Project Data Processing Subgroup. The Sequence Alignment/Map format and SAMtools. Bioinformatics. 2009 25(16):2078-9. doi: 10.1093/bioinformatics/btp352.

Lischer HEL, Shimizu KK. Reference-guided de novo assembly approach improves genome reconstruction for related species. BMC Bioinformatics. 2017 18(1):474. doi: 10.1186/s12859-017-1911-6

Manni M, Berkeley MR, Seppey M, Simão FA, Zdobnov EM. BUSCO Update: Novel and Streamlined Workflows along with Broader and Deeper Phylogenetic Coverage for Scoring of Eukaryotic, Prokaryotic, and Viral Genomes. Mol Biol Evol. 2021 38(10):4647-4654.

Moshiri N, Fisch KM, Birmingham A, DeHoff P, Yeo GW, Jepsen K, Laurent LC, Knight R. The ViReflow pipeline enables user friendly large scale viral consensus genome reconstruction. Sci Rep. 2022 12(1):5077. doi: 10.1038/s41598-022-09035-w.

Ondov BD, Treangen TJ, Melsted P, Mallonee AB, Bergman NH, Koren S, Phillippy AM. Mash: fast genome and metagenome distance estimation using MinHash. Genome Biol. 2016 17(1):132. doi: 10.1186/s13059-016-0997-x.

Paradis E, Schliep K. ape 5.0: an environment for modern phylogenetics and evolutionary analyses in R. Bioinformatics. 2019 35(3):526-528. doi: 10.1093/bioinformatics/bty633.

Parmigiani L, Garrison E, Stoye J, Marschall T, Doerr D. Panacus: fast and exact pangenome growth and core size estimation. Bioinformatics. 2024 40(12):btae720. doi: 10.1093/bioinformatics/btae720.

Paten B, Eizenga JM, Rosen YM, Novak AM, Garrison E, Hickey G. Superbubbles, Ultrabubbles, and Cacti. J Comput Biol. 2018 25(7):649-663. doi: 10.1089/cmb.2017.0251.

R Core Team (2025). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/

Rakocevic G, et al Fast and accurate genomic analyses using genome graphs. Nat. Genet. 2019 51, 354–362.

Revll LJ. phytools 2.0: an updated R ecosystem for phylogenetic comparative methods (and other things). PeerJ 12:e16505 https://doi.org/10.7717/peerj.16505

Sayers EW, et al. Database resources of the National Center for Biotechnology Information. Nucleic Acids Res. 2021 49(D1):D10-D17. doi: 10.1093/nar/gkaa892.

Schliep KP. phangorn: phylogenetic analysis in R. Bioinformatics. 2011 27(4):592-3. doi: 10.1093/bioinformatics/btq706.

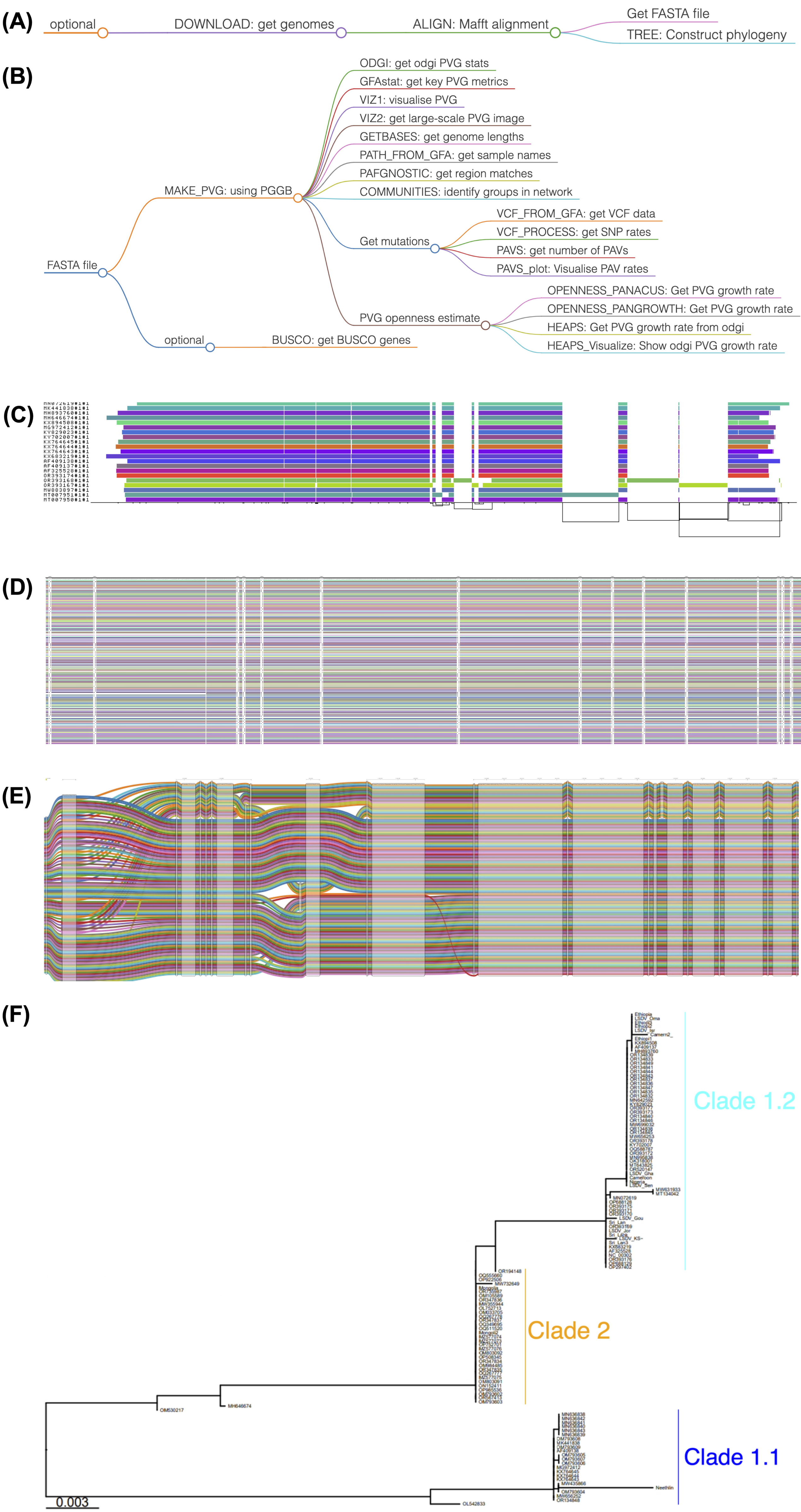
Valiente-Mullor C, et al. One is not enough: On the effects of reference genome for the mapping and subsequent analyses of short-reads. PLoS Comput Biol. 2021 17(1):e1008678. doi: 10.1371/journal.pcbi.1008678.

Wang LG, et al. Treeio: An R Package for Phylogenetic Tree Input and Output with Richly Annotated and Associated Data. Mol Biol Evol. 2020 37(2):599-603. doi: 10.1093/molbev/msz240.

Wickham H. 2016. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York. ISBN 978-3-319-24277-4, https://ggplot2.tidyverse.org.

Xie S, Cui L, Liao Z, Zhu J, Ren S, Niu K, Li H, Jiang F, Wu J, Wang J, Wu J, Song B, Wu W, Peng C. Genomic analysis of lumpy skin disease virus asian variants and evaluation of its cellular tropism. NPJ Vaccines. 2024 9(1):65. doi: 10.1038/s41541-024-00846-8.

Yu G, Smith D, Zhu H, Guan Y, Lam TT. ggtree: an R package for visualization and annotation of phylogenetic trees with their covariates and other associated data. 2017. Methods in Ecology and Evolution, 8, 28-36. doi:10.1111/2041-210X.12628.


(A)
optional
DOWNLOAD: get genomes
ALIGN: Mafft alignment
Get FASTA file
TREE: Construct phylogeny
(B)
FASTA file
MAKE_PVG: using PGGB
ODGI: get odgi PVG stats
GFAstat: get key PVG metrics
VIZ1: visualise PVG
VIZ2: get large-scale PVG image
GETBASES: get genome lengths
PATH_FROM_GFA: get sample names
PAFGNOSTIC: get region matches
COMMUNITIES: identify groups in network
Get mutations
VCF_FROM_GFA: get VCF data
VCF_PROCESS: get SNP rates
PAVS: get number of PAVs
PAVS_plot: Visualise PAV rates
PVG openness estimate
OPENNESS_PANACUS: Get PVG growth rate
OPENNESS_PANGROWTH: Get PVG growth rate
HEAPS: Get PVG growth rate from odgi
HEAPS_Visualize: Show odgi PVG growth rate
optional
BUSCO: get BUSCO genes
(C)
(D)
(E)
(F)
Clade 1.2
Clade 2
Clade 1.1
0.003