Genomics, Proteomics & Bioinformatics, 2026, **24(2)**, qzag003
https://doi.org/10.1093/gpbjnl/qzag003
Published: 3 February 2026
Review

GPB Genomics, Proteomics & Bioinformatics
OXFORD

Approaches to Studying Viral Pangenome Variation Graphs

Tim Downing[1,2,*]

[1]Dublin City University, Dublin, D09 A260, Ireland
[2]The Pirbright Institute, Surrey, GU24 0NF, UK
*Corresponding author: Tim.Downing@pirbright.ac.uk (Downing T).
Handling Editor: Ting Wang

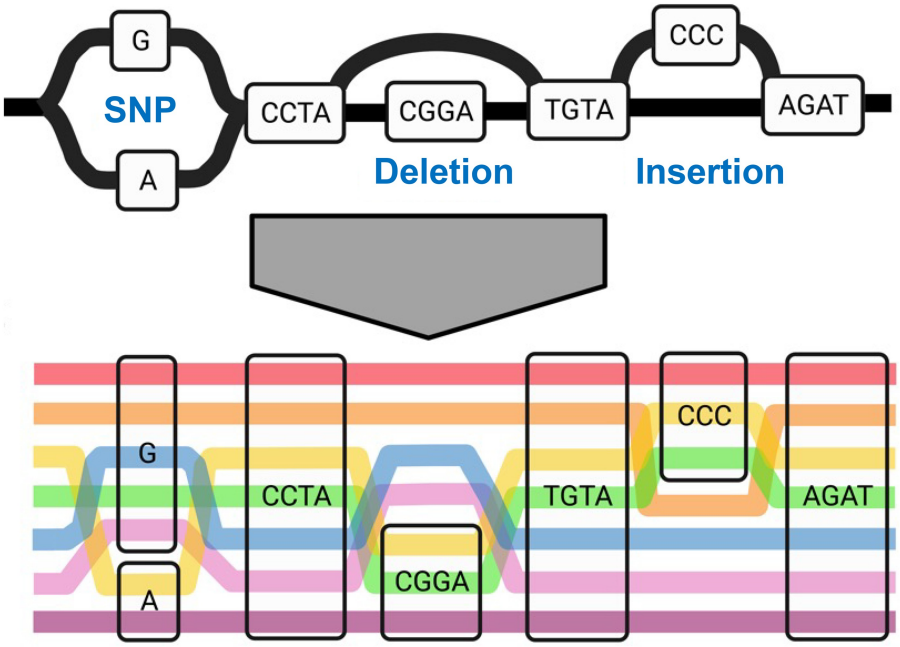
Abstract

Pangenome variation graphs (PVGs) allow for the representation of genetic diversity in a more nuanced way than traditional reference-based approaches. Here, I focus on how PVGs are a powerful tool for studying genetic variation in viruses, offering insights into the complexities of viral quasispecies, mutation rates, and population dynamics. PVGs originated in human genomics and hold great promise for viral genomics. Previous work has been constrained by small sample sizes and gene-centric methods, whereas PVGs enable a more comprehensive approach to studying viral diversity. Large viral genome collections should be used to make PVGs, which offer significant advantages. Here, I outline accessible tools to achieve their construction. These tools span PVG construction, PVG file formats, PVG manipulation and analysis, PVG visualisation, PVG openness measurement, and read mapping to PVGs. Additionally, the development of PVG-specific formats for mutation representation and personalised PVGs that reflect specific research questions will further enhance PVG applications. Challenges remain, particularly in managing nested variants, optimising error detection, optimising k-mer/minimizer-based approaches for AT-rich genomes, incorporating long-read sequencing data, and developing scalable visualisation approaches. Nevertheless, PVGs offer a new opportunity for viral population genomics, and a testing ground for tool development prior to application to larger eukaryotic genomes. These advances will enable more accurate and comprehensive detection of viral mutations, contributing to a deeper understanding of viral evolution and genotype–phenotype associations.

Key words: Genomics; Pangenomics; Virus; Genome evolution; Pangenome graph.

Received: 6 December 2024. **Revised:** 1 October 2025. **Accepted:** 13 January 2026.

Graphical abstract



What are pangenome variation graphs?

Pangenomes facilitate the investigation of broader patterns across a sample collection or lineage. Traditionally, a pangenome typically refers to the set of all genes in a sample collection (first coined by Sigaux in 2000 [1]). This could also be considered as the number of genetic elements [2]. Recently, gene-based pangenomes have been supplanted by more highly resolved nucleotide-level pangenomes, where each base can act as a genetic element of interest. These are known as pangenome variation graphs (PVGs) or sometimes pangenome graphs; the former term is used here.

A PVG is a graphical representation of the set of all mutations in a sample collection (**Figure 1**). Graphs can be used to represent genomic data in de Bruijn graph (DBG), string-based graph, and other formats. Such graphs consist of nodes (or vertices), which represent sequences of varying lengths, and edges that link these nodes derived from the same underlying sequence. One can then traverse through the PVG along the "path" associated with a single sample (or haplotype). Paths in the graph reflect the nodes observed in one or more samples in a specific order, and a traversal like this can be referred to as a "walk" (**Table 1**). For example, walking along a DBG starts with a single path that may split and merge at mutation sites that create bifurcating paths. These splits are termed bubbles that indicate alternate genetic sequences that split before merging later [3]. Superbubbles are an extension of this where multiple paths split, meander, and connect later (all at the same node). The diversity of these bubbles and superbubbles is a function of the input data [3].

Why should we use PVGs?

Traditional methods for characterising samples map read libraries to existing linear reference genomes. A reference genome is a haploid sequence that does not contain any diversity, and thus is prone to the reference allele bias when reads are mapped to it. This is because a read will not map to the correct reference genome region if the number of nucleotide differences is high, resulting in missed mutations. Neither using a consensus reference sequence nor using a multi-sample linear reference addresses these limitations in viruses [4,5]. Mutation detection power is a function of genetic distance of the sample to the reference, resulting in impaired mutation detection for samples distantly related to the reference. As a result, accuracy decreases for divergent samples that align poorly to a single reference genome [2]. This is particularly limiting when examining reassortment or novel specimens. Such reference bias is a known limitation and means that variation at hypervariable loci is more difficult to resolve [6]. As a result, novel mutations, structural variants (SVs), and copy number variants (CNVs) may be missed [7]. Novel pathogen isolates genetically distinct from known references will be characterised less accurately [8,9]. Such missed or misinterpreted mutations result in imprecise inference of phylogenetic relationships and the molecular bases of phenotype changes [10–12].

Mapping reads to multiple linear reference genomes is one possible solution; however, this approach underperforms compared to using PVGs, which enable higher read mapping efficiency [13]. Another potential solution is *de novo* genome assembly, which assists in profiling novel specimens in principle.

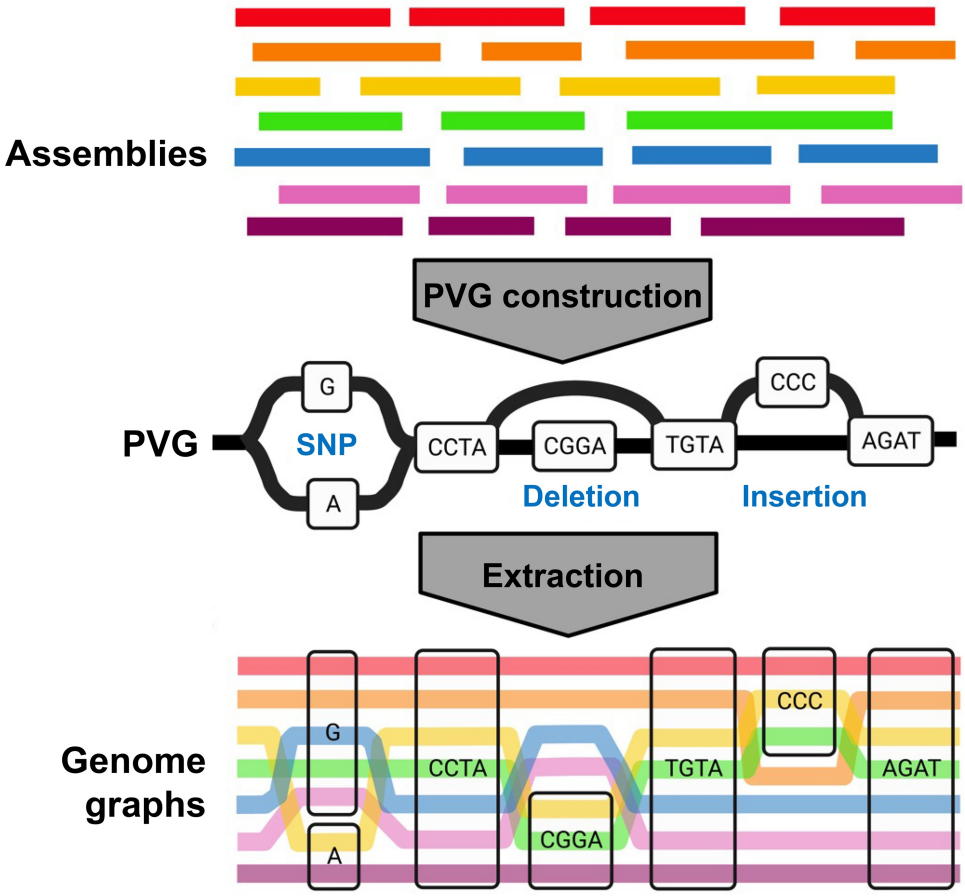


Figure 1 PVG construction and interpretation
A set of assemblies (top) for seven different samples (represented by different colours) can be used to construct a PVG (middle) to include all variations observed in this collection, such as SNPs, deletions, or insertions. Not all samples possess each of these mutations, so the genome graphs per sample (bottom) depict differing paths. PVG, pangenome variation graph; SNP, single-nucleotide polymorphism.

In practice, this approach is inadequate: errors in reference genomes, AT/GC content biases, host and vector contamination, short contigs, and repetitiveness can result in incomplete and biased genomes [14,15]. Moreover, contigs often need to be aligned to existing databases either as sequences or k-mers [16,17], and many double-stranded DNA (dsDNA) viruses require target enrichment or long-range or tiling PCR amplification, which require known reference genome(s). Although long-read technologies are helping to address this genome assembly issue thanks to new approaches yielding lower error rates, the cost remains high. In addition, there are substantial numbers of high-quality genome assemblies and short-read libraries for many important viruses, which can be examined effectively using PVG-based approaches. In this way, PVGs represent a cheaper approach to profile a large number of published genomes and short-read libraries.

A further consideration is linear reference genome quality. Some viral genomes lack high resolution of complex repeat structures, terminal inverted repeats (TIRs), regions with secondary structures, and polynucleotide tracts. This affects mutation detection at such regions, resulting in false positive and missed changes. PVGs reduce this bias by combining multiple genomes into one graph, thus allowing reads to map to paths that more closely reflect their true structures.

PVGs integrate genetic variation from an entire sample collection, capturing diversity that better reflects wider populations [6]. By representing alleles as distinct paths within the PVG, PVGs enable improved read mapping compared to traditional methods (Figure 1). These paths encode various genomic features, such as alleles, haplotypes, alignments, and alternate sequence paths that correspond to genetic variation [3].

PVGs are similar to DBGs: they merge shared sequences into conserved nodes where the PVG node lengths can vary widely [6]. PVGs are bidirected graphs where nodes can be traversed in either direction. However, PVGs' functional annotation and coding sequence (CDS) orientation provide directionality across nodes for biological interpretation. Rare nodes in PVGs can represent sequencing errors: these are typically removed during graph construction to ensure accuracy and allow easier manipulation of complex graphs. Such errors originate as sequencing artefacts: for example, the median error rate at bases generated by older Illumina MiSeq sequencing is $\sim$ 0.5% [18]. A sequence graph is a bidirected set of connected nodes containing sequences. A genome graph is a set of annotated paths in a sequence graph that reflect a set of complete genomes [19].

Table 1 Common terms in PVG analysis

Term	Definition
PVG	A graphical representation of all mutations across a sample collection
DBG	A type of graph representing sequences by overlapping k-mers, where splits represent mutations
Bubble	A PVG structure where a path splits and rejoins, indicating alternate genetic sequences
Superbubble	A bubble structure where multiple paths diverge and reconverge, indicating complex mutations
Walk	A path through a graph corresponding to the sequence of a single sample or haplotype
Sequence graph	A single genome extracted from a graph as a connected set of nodes (a walk), often bidirected
Genome graph	A path through a graph representing a full genome, factoring in gene orientation and annotation
GBWT	A compressed index structure representing many paths through a graph for efficient mapping
GFA	A generic interpretable format for representing DBGs
PAF	A file format for recording pairwise alignments
GAF	A generalisation of PAF format for use with PVGs in GFA
PVG in .vg format	A bidirectional graph for representing PVGs containing HashGraph or PackedGraph data
PVG in .pg format	A memory-efficient, static version of a PVG with PackedGraph data
PVG in .xg format	A static index of a PVG's paths, nodes and edges for queries
GCSA	An index of suffixes in a graph used for efficient sequence location queries
snarl	A subgraph representing allelic or structural differences (like a bubble) between paths
Minimizer	A subsequence of a k-mer representing a sequence region; used for fast mapping
GAM	A graph-based representation of read alignments on a PVG, equivalent to BAM/SAM files
PACK file	The read support for candidate mutations in an augmented PVG
Surjection	The conversion of a GAM file into a BAM file

Note: PVG, pangenome variation graph; DBG, de Bruijn Graph; GBWT, graph Burrows-Wheeler transform; GFA, Graphical Fragment Assembly; PAF, Pairwise Alignment Format; GAF, Graph Alignment Format; GCSA, Generalised Compressed Suffix Array; GAM, Graph Alignment Map; BAM, Binary Alignment Map; SAM, Sequence Alignment Map.

The primary advantage of using PVGs is a higher resolution of intrasample diversity, including insertions and deletions (indels), SVs, and CNVs. SVs include translocations, inversions, and larger mutations, and typically have lengths $>$ 50 bp, whereas small indels are $<$ 50 bp. This benefit of graphical formats is especially important when assessing paired-end short reads, where the total fragment size may limit power to detect large SVs, particularly compound SVs [7]. For instance, reads whose maximum fragment length is 900 bp cannot properly resolve SVs with lengths $>$ 900 bp, because there are be no reads that fully span the SV length, rendering the exact length of the SV unclear. Although long reads possess better SV resolving power, this inferential capacity is still a function of read length that can be improved by PVGs. A secondary advantage of graphical formats is that they enable computationally scalable methods [20], allowing more thorough analyses [2]. As a consequence, PVGs have better sensitivity and accuracy [4,21,22].

Examples of SVs uniquely detectable using PVG-based methods include those linked to rare human diseases [23], genome-wide association studies (GWAS) in humans [24], diversity in sheep [25], and variation in tomatoes [26]. Over two-thirds of SVs were missed due to the usage of short reads mapped to a single reference genome in humans [27]. Moreover, PVGs found an average of 64,000 additional mutations per human sample, because 10% more reads were perfectly mapped to the PVG [28]: these hard-to-resolve regions/mutations are often strongly associated with phenotypic changes [29].

Consequently, PVGs are effective approaches for understanding genetic variation at the subspecies level, such as populations, clades, or any genetically related grouping lacking substantial structural genomic changes. Highly divergent samples or collections will prevent a PVG from being formed properly. An example of this would be examining all African swine fever virus (ASFV) genomes, whose plastic genomes form tandem gene arrays with high CNV levels. Incomplete PVGs possess low linearity and high levels of structural changes, such as bubbles and nested paths. An unanswered challenge in this area is to quantify how diverse informative PVGs can be.

PVG construction

A large number of PVG construction tools have been developed, applying the optimal approach of graph construction from genome assemblies (or sometimes reads) rather than from compiled mutation files, such as Variant Call Format (VCF) files [7]. These mutation files are insufficient because they were originally created based on comparisons of samples to a linear reference genome. As a result, they encode information specific to that comparison alone, omitting population-wide data associated with complex mutations.

Chief among the PVG construction tools is PanGenome Graph Builder (PGGB) [30], which creates PVGs using three main steps: genome-wide pairwise alignment, PVG induction, and PVG refinement. The alignment stage, implemented by wfmash [31], examines 256-bp segments within regions of 1–25 kb to cluster sequences sharing at least 90% similarity. This alignment step is optimised, so that as the number of genomes becomes large, the compute time remains manageable without genome partitioning. The induction phase uses Seqwish, which constructs a PVG from the aligned sequences through an intermediate tree-based process [9]. Refinement is handled by Smoothxg [32], which splits the graph into blocks on which local realignment is completed to make partial order alignment (POA) graphs. Smoothxg is dependent on the input parameters, such as k-mer length (*e.g.*, 19 bp) and window size (*e.g.*, 27 bp). PGGB performs well across diverse datasets, including eukaryotic, prokaryotic, and viral genomes, and is available as part of the Nextflow nf-core [33]. It also provides extensive data exploration and visualisation options. However, its use of an all-to-all alignment approach means that scaling to large sample numbers can be challenging.

Another widely used tool is Minigraph, which can both construct PVGs and map reads to them [34]. Minigraph requires an initial PVG or reference genome and iteratively maps each sequence to the PVG, gradually reshaping it to incorporate observed diversity. It achieves this by identifying co-linear minimizer chains using Minimap2 [35], generating linear chains that are iteratively connected and incorporated into the PVG based on mapping quality and length. The resulting graph structure depends on the input order of the genome assemblies analysed. Minigraph was originally designed for large vertebrate genomes. Related to this is the Minigraph-Cactus pipeline [36], which combines Progressive Cactus alignment approaches [37] with Minigraph [34]. This pipeline was developed for larger genomes and is part of the Cactus software package.

A related tool, Scaffold Graph Toolkit [38], enables PVG construction in the form of scaffold graphs in formats such as FASTQ or Graphical Fragment Assembly (GFA). It supports associated analyses and interactive visualisation through Cytoscape. Input data can include FASTQ, FASTA, Sequence Alignment Map (SAM), Binary Alignment Map (BAM), or contig files.

An important consideration for PVG construction is scalability, which has been enhanced by indexing a PVG as a graph Burrows-Wheeler transform (GBWT) by Optimized Dynamic Genome/Graph Implementation (ODGI) [39] and Giraffe [20] without the need to search across the graph itself. Additionally, indexes like the r-index scale better with large datasets [40]. Nonetheless, scalability is still a function of viral genomic diversity, length, and repetitiveness. Existing approaches align sequence fragments to one another to iteratively build a PVG that can be pruned [30,36,38]. However, viruses with high diversity will create a complex PVG with numerous paths, nested variants, bubbles, and superbubbles. Thus, considerable thought must be given to the taxonomic unit of interest (population, subclade, clade, serotype, or species) such that the PVG can be interpreted effectively.

Pangenome graph file formats are also an essential consideration, with GFA being the primary format (https://github.com/GFA-spec/GFA-spec). However, GFA currently lacks coordinate support. Reference GFA (rGFA) [34] includes reference paths in the form of walks (*i.e.*, sets of connected nodes) but may not fully accommodate the representation of complex mutations. There are two commonly used GFA versions: GFA1.0, used in PGGB-related methods, and GFA1.1, associated with Minigraph workflows. A third version, GFA2, is less widely used at present (https://gfa-spec.github.io/GFA-spec/). Methods for each GFA type are typically specific to its specification, with limited interoperability between GFA1.0 and GFA1.1. Tools like PGGB generate pairwise alignment files in the Pairwise Alignment Format (PAF) format [30], which has been extended to the Graph Alignment Format (GAF) for PVGs in GFA format. GFA files can be converted to GAF or PAF with minichain [41], where GAF generalises the text-based PAF format. Lastly, a PVG Format (PGVF), derived from GFA1.0, for graph-to-graph comparisons has been proposed but remains unsupported thus far (https://github.com/pangenome/pgvf-spec).

Table 2 PVG mapping tools suitable for viral PVG analyses

Tool	Description	Ref.
GraphTyper2	Use short reads to genotype mutations with a full PVG	[52]
Paragraph	Use short reads to genotype structural variants with a local PVG	[51]
PanGenie	Use short reads' k-mers to genotype with a haplotype-resolved PVG	[53]
Giraffe	Map short reads to allelic variation	[20]
VG-MAP	Map short reads to known haplotypes	[7]
VG-MPMAP	Map short reads to multiple paths	[7]
V-MAP	Map short or long reads to a PVG	[55]
Minigraph	Map short or long reads to a PVG and can construct PVGs	[34]
GraphAligner	Map long reads to a PVG	[54]
PanAligner	Map long reads to a PVG	[56]

Note: VG, Variation Graph; V-MAP, Variant Map.

PVG analysis, manipulation, and visualisation

PVG analysis and visualisation tools are essential for deciphering both simple and complex SVs [42]. These tools extend and integrate with PVG construction tools but often have significant back-end complexity or lack command-line interfaces. Among the most prominent is ODGI, which enables computationally efficient PVG examination and is an indispensable tool for PVG analysis with (currently) 44 functions [39]. ODGI provides extensive functionality, including PVG manipulation, node trimming, sorting, exporting, and various refinements, along with visualisation capabilities. By default, ODGI works with GFA1.0 format but can be toggled via Minigraph-Cactus to support GFA1.1 using the Variation Graph (VG) toolkit's convert function. ODGI works with PVGs in .og format and has a partial Python equivalent called mygfa (https://github.com/cucapra/pollen/tree/main/mygfa).

Numerous other PVG analysis tools complement ODGI. Gretl (https://github.com/MoinSebi/gretl) provides rapid summaries of GFA statistics, core metrics, and path similarity metrics, including rates per node and sequence amounts. Gfatools [43] enables subgraph extraction from a GFA and conversion of GFA files to FASTA or Browser Extensible Data (BED) format. Gaftools (https://github.com/marschall-lab/gaftools) offers a wide range of GAF-related functions, including GFA-to-GAF indexing, GAF viewing, subsetting, alignment sorting, realignment, subsequence extraction from node paths, PVG ordering, phasing using tabular data, and numerical summarisation of GAF files. Gfakluge can sort PVGs in GFA format, extract FASTA sequences from GFA, convert between GFA1.0 and GFA1.1, and merge GFA pairs. Similarly, Gfapy can perform conversions between GFA1.0 and GFA1.1 [44]. Impg allows the extraction of a PVG region based on a region of interest in a particular sample/path (https://github.com/pangenome/impg). Another tool, Gafpack, computes coverage metrics for GAF files based on GFAs (https://github.com/pangenome/gafpack). Chrom_mini_graph creates a coloured minimizer PVG based on FASTQ queries (https://github.com/gaojunxuan/chrom_mini_graph). Gfalace usefully allows the merging of different PVGs (https://github.com/pangenome/gfalace).

In terms of PVG quality, there are four main options. The first is the pangenome graph evaluator (PGGE; https://github.com/pangenome/pgge), which assesses PVG accuracy and optimises PVG construction across various approaches. The second is the pangenome single copy and universal k-mer counter (PG-SCUnK), which assesses PVG quality using single-copy and universal k-mers based on the principle that single-copy k-mers are orthologous across samples (https://github.com/cumtr/PG-SCUnK/) [45]. The third is ODGI validate, which compares the PVG topology to its constituent paths [39]. The fourth is VG validate, which can examine PVG nodes, node structures, path connectedness across edges, and also the compatibility of a Graph Alignment Map (GAM) file with a PVG [22].

Despite the availability of analysis tools, effective PVG visualisation tools for non-human organisms remain limited. One of the best options is SequenceTubeMap [46], which visualises PVGs in .pg or .vg format via a web interface or Docker installation. It supports GAM files and/or haplotype data, as well as BED file annotations. However, SequenceTubeMap imposes a 5 MB upload limit, which may require down-sampling of certain GAM files. Other visualisation tools include ODGI's view function; gfaestus for 2D visualisation (https://github.com/chfi/gfaestus); the VG toolkit's view function [22]; pvg (https://github.com/w-gao/pgv); the Assembly Graph Browser (AGB) [47]; Waragraph (https://github.com/chfi/waragraph) for generating 1D and 2D plots; Bandage, which has been effective for graphical visualisation for some time [48]; and gfaviz, which creates 2D plots, scaffold graphs, alignment graphs, and PVG images [49]. A newer option is wally, which visualises the mutations in mapped short reads, long reads, contigs, and PVGs using BAM or Compressed Reference-oriented Alignment Map (CRAM) input (https://github.com/tobiasrausch/wally) [50].

PVG read mapping and mutation detection

A wide variety of tools exist for constructing and analysing PVGs based on multiple sequence alignment (MSA) and read mapping (**Table 2**). Notably, read mapping to a PVG demonstrates computational efficiency comparable to mapping against linear references [2]. Earlier methods focused on mapping short reads to DBGs, but modern tools enhance efficiency using k-mer-based strategies, suffix trees, and Burrows-Wheeler transforms (BWTs). VG's map function (VG-MAP) mapped reads to a PVG in the same manner as mapping to a linear reference where a read could map to the range of haplotypes in the PVG [22]. However, this has been superseded by VG's giraffe function (Giraffe). To map with Giraffe, VG uses a BWT for efficient PVG indexing and read mapping using POA to create GBWT and GBZ indices [7]. Giraffe maps reads to the combination of alleles across haplotypes, using minimizer and genetic distance data as additional input. Consequently, Giraffe maps reads to synthetic haplotypes absent in the original PVG. For this reason, Giraffe is much more effective than VG-MAP and VG-MPMAP, as the latter two map reads to multiple paths [20]. Effective SV detection tools using graph include Paragraph [51], VG [22], and GraphTyper2 [52]. Paragraph and GraphTyper2 use different approaches to re-map reads at known mutations in the reference graph to improve the original detection based on a linear reference [51,52]. Lastly, PanGenie uses a k-mer-based approach to support genotyping, and has high specificity in detecting mutations and sub-consensus changes in closely related specimens [53].

In addition, some tools can now map long reads to reference graphs, such as Giraffe [20], GraphAligner [54], Minigraph [34], Variant Map (V-Map) [55], and PanAligner [56]. GraphAligner maps long reads and may be better thanks to banded sequence alignment, whereas V-Map tolerates both short and long reads [54,55]. Other options do exist, but these are either old or designed with human data in mind.

Like ODGI above, PVG analysis and mapping need to include the VG toolkit [7]. VG can integrate information from VCFs and assemblies, and create PVGs representing the full spectrum of known diversity in a sample collection. Consequently, VG is an analysis tool as well as one for read mapping. VG has a range of file conversion capabilities, as well as ones to view PVG

components. Although VG's construct function can be used to make a PVG based on VCF data, this is not advised if the mutation data come from linear reference mapping. Gfautil (https://crates.io/crates/gfautil) and VG's deconstruct function allow conversion in the other direction (from GFA to VCF).

For PVG read mapping, there are more complex file format options that are akin to SAMtools [57] faidx indexing. VG's convert function can convert PVGs into the bidirectional .vg (HashGraph or PackedGraph) format, allowing the PVG to be modified by VG (**Figure 2**). Optionally, long nodes can be reduced in size using VG's mod function for large PVGs. To create a compressed haplotype index, VG's gbwt function can convert GFA files into GBZ or GBWT format. Another common PVG file format is .xg (XG), which is a static sequence-free PVG index for querying. The PVG .xg format can be converted into .pg (PackedGraph) format with VG's convert function. PVGs can also be indexed in Generalised Compressed Suffix Array (GCSA) format, which contains a suffix array of sequence locations on the PVG [58].

The next step is PVG pruning: the removal of rare or discontinuous nodes. For small genome sizes such as those of viruses, this may not be needed, but for large or complex datasets this is crucial. Following this, VG's snarls function can be used to create a snarl index from the PVG in .pg format [3]. A snarl is a subgraph representing allelic or structural difference between paths, sometimes also called a bubble.

VG's deconstruct function can take the PVG in .xg format and the GBWT index to create a VCF of the PVG's diversity. VG's autoindex function is useful because it automatically detects the appropriate indexing approach based on the input data. Lastly, VG's minimizer function can use the GBWT index (and a set of genetic distances) to create a set of minimizers to support other analyses.

In relation to read mapping file formats, GAM is analogous to SAM and BAM [22]: they encode the coordinates of the reads' matches on the PVG. GAM files are also equivalent to GAF files. VG's stat function can generate mapping metrics on a GAM file. VG augments reads in a GAM file and the PVG in .pg format to create an augmented PVG: the latter integrates the mutations in the GAM file with that PVG. VG pack uses this augmented PVG to make a PACK file, generating read support for each candidate mutation. VG call can make a VCF file for that sample's reads using this PACK file, the augmented PVG, and a snarls file. In parallel, VG depth takes the PVG in .vg format and the PACK file to create a file containing the read depth across the PVG. VG can surject GAM files into BAM format for processing with widely used tools such as Freebayes [59], DeepVariant [60], BCFtools [61], and GATK [62].

PVG openness

The number of samples required to create a representative PVG is specific to the research question and sample collection involved. This can be assisted using Heap's Law to model the rates of new mutations as the number of samples included increases [63]. If new mutations (on average) continue to be discovered as more samples are added to the PVG, the PVG is considered open. Conversely, if no additional mutations are found, it is considered closed. The number of new mutations in N genomes sharing n mutations can be estimated from the power-law regression $n = \kappa N^{\text{gamma}}$ where κ is a scalar [64] (**Figure 3**A). The rate of new mutation (Δn) can be calculated from $\Delta n = kN^{-\text{alpha}}$ where k is another scalar, and an open PVG has alpha < 1 while a closed one has alpha > 1 [65] (Figure 3B).

There are three main approaches for estimating PVG openness. The first is Pangrowth, which estimates this based on the rate of increase in the number of unique k-mers as more genomes are added to the PVG [66]. The second is Panacus, which uses the rate of growth in nodes in the PVG as the genome number increases [67]. These are important because they allow computation of the sample size needed to reflect the population's broader diversity based on the sample taken (Figure 3A). As the relative rate of new k-mers or nodes becomes asymptotically close to zero, additional samples may not yield novel variation that informs on genomic or evolutionary questions in the dataset. Such a PVG would be closed with alpha > 1 (Figure 3B).

In a gene-based pangenome, the core genome is composed of genes shared by (usually) $>$ 99% samples, and the remaining genes at lower frequencies (typically $<$ 95%) represent the accessory genome [63]. The precise thresholds used here ($>$ 99%,

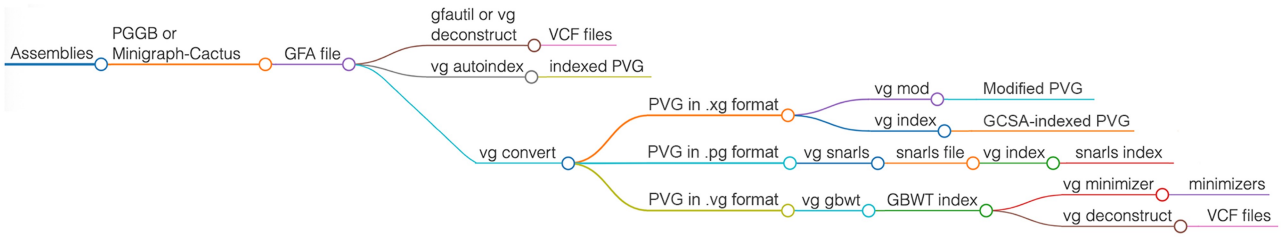


Figure 2 PVG file formats and tool functions

A set of assemblies can be converted into GFA format using tools such as PGGB or Minigraph-Cactus. Then the GFA file can be converted into VCF files by VG's deconstruct function or Gfautil, and indexed with VG's autoindex function. The GFA file can be converted by VG's convert function into .xg, .pg, or .vg formats. The PVG in .xg format can be modified by VG's mod function, and VG's index function can create a GCSA-indexed PVG from it. The PVG in .pg format can be used to create a snarls file using VG's snarls function, which can then be converted into a snarls index using VG's index function. The PVG in .vg format can be used to generate a GBWT index using VG's gbwt function. This index can be used to (1) create VCF files using VG's deconstruct function and (2) generate minimizers using VG's minimizer function (if given a set of genetic distances as additional input). GFA, Graphical Fragment Assembly; VCF, Variant Call Format; VG, Variation Graph; GCSA, Generalised Compressed Suffix Array; GBWT, graph Burrows-Wheeler transform.

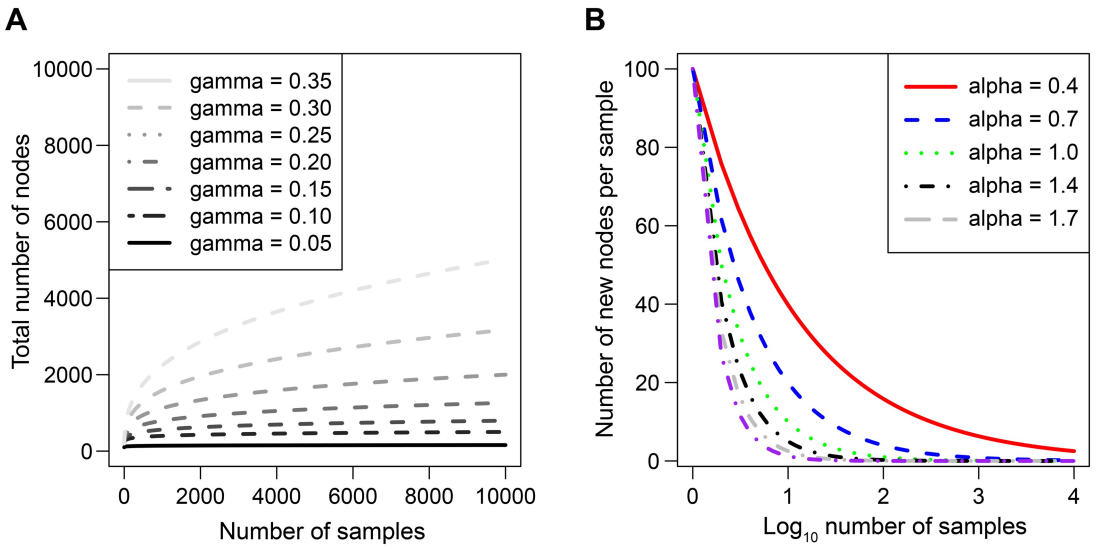


Figure 3 Modelling PVG openness

A. Total number of nodes as the number of samples increases. The grey lines show differing values of gamma such that the total number of nodes in N samples is κN^{gamma} where $\kappa = 100$. **B**. Number of new nodes per sample as the number of samples increases. The coloured lines show varying levels of alpha such that the number of new nodes is $kN^{-\text{alpha}}$ where $k = 100$.

$>$ 95%, and $<$ 95%) for core, accessory, and other strata depend on the genomes being examined. ODGI's heaps function computes the number of bases in a PVG as the number of genomes in the PVG increases and tells us more about the PVG-based terms for core and accessory pangenomes [39]. Firstly, the shared PVG from ODGI heaps reflects the sum of bases at nodes found in all genomes, analogous to the core genome. Secondly, the sum of bases at all node is the complete PVG, *i.e.*, every single base in the PVG. Thirdly, the sum of nodes in 50% of samples indicates the median PVG size, which may estimate the median genome size in the sample set.

Viral haplotype reconstruction from PVGs

Viral haplotype reconstruction is closely linked to PVGs: both processes aim to represent the genetic variation within a population of viral genomes. The frequencies of major and minor haplotypes can vary across different samples and genomic regions. Viral haplotype reconstruction is a form of strain-aware metagenome assembly, and it is particularly complex due to the varying viral effective population sizes and mutation rates, which lead to significant differences in within-host sequence diversity [68]. For viruses with segmented genomes, an additional layer of complexity arises from the difficulty in resolving the linkage between segments. These linkages are often challenging and may not be fully resolvable in some cases. Current haplotyping methods, which typically rely on *de novo* assembly or read mapping to a reference genome, show varied outcomes and can struggle when applied to highly variable genomic data [68]. Furthermore, the depth of sequencing reads is an important factor in detecting rare minor alleles. Most read mapping tools have maximum mismatch rates, which are set based on the error rates of the sequencing technology. These constraints can limit the detection of rare variants, which may be crucial for understanding viral diversity within a population.

Viral haplotypes can be reconstructed from the paths in the PVG, where the inter-node coverage correlation plays a significant role. In most cases, these resolved viral haplotypes correspond to paths in the PVG, and since they are largely composed of long unitigs, most of the nodes are relatively stable with few mutations. However, regions of the genome with low complexity can complicate this process, leading to cycles and bubbles in the potential paths. Typically, directed acyclic graphs (DAGs) are sufficient when representing homogeneous viral genomes because they assume a start and an end.

However, complex ones might need bidirected graphs, which model contig orientation and SVs more accurately. One approach to resolving these complexities is to build a contig PVG using initial *de novo* assembly contigs and sequencing read libraries. From there, the relative change in allele frequencies across alternate contig paths is minimized [69]. This method uses the coverage of mapped reads to help determine the relative frequencies of each haplotype across the contigs, which is essential for resolving the viral haplotypes. Once the haplotypes are resolved, they constitute individual genome graphs that, when aggregated, form the overall PVG.

Other graphical representations of viral quasispecies have been developed to handle mixed sample metagenomes. Tools like SAVAGE [12], Virus-VG [70], and VG-Flow [69], which use flow variation graphs, offer solutions for representing haplotype abundance and further improving the accuracy of viral haplotype reconstruction in complex viral populations. For all approaches, read error correction before assembly can be beneficial to improve minor allele detection. Tools that perform this correction assume that sequencing errors are rare k-mers in a graph and not rare haplotypes [19]. This approach is particularly useful because PVGs, due to their structure, may highlight rare haplotypes that might otherwise be missed.

Potential for PVGs applied to viral genomes

PVGs originated in human genomics and there are significant opportunities for their use to illuminate viral genetic variation. No PVG analyses on large viral sample collections have been conducted yet. Pangenomic approaches have been applied to viruses, including 59 hepatitis B read libraries [13], 8 *Pithoviridae* genomes [71], 160 Ebolavirus genomes [72], 42 ASFV genomes [73], 46 ASFV genomes [74], and 36 *Chlorovirus* genomes [75]. Excluding hepatitis B and Ebolavirus, these studies have been constrained by gene-centric methods and small sample sizes ($n < 50$).

Animal and plant pathogenic viruses typically have genome lengths < 30 kb [76], potentially rendering PVG-based approaches less informative. The genome sizes of dsDNA viruses are usually larger (range: 2.47–4.7 Mb), whereas single-stranded DNA (ssDNA) viruses are usually smaller (range: 0.86–10.9 kb) [76]. PVG-style approaches could be relatively accessible for more than 715 DNA and RNA viral genera whose genome is a single molecule, which include all dsDNA and most ssDNA viruses. Forty-six additional genera have a segmented genome contained within a single particle, and a further 45 have multipartite genomes where each segment is packaged in a separate particle. For the latter 91 genera, PVG construction is more complex, but can draw on approaches from vertebrates.

A paradigm in this area is the PVG investigation of the 3.2-kb hepatitis B virus genome [13]. This genome is unusual: it is a small, partially dsDNA, and circular, yet can replicate independently in hosts in part thanks to four partially overlapping open reading frames (ORFs). Existing linear hepatitis B reference genomes showed wide heterogeneity of mapping outcomes for real and simulated read libraries, and demonstrated that no such reference performed as well as a representative PVG [13]. Additionally, this accuracy was a function of genetic divergence, such that PVGs contributed more information for samples with higher diversity [13]. This was particularly acute in more variable regions, which may be associated with $> 8\%$ of reads that did not map to linear references but did map to a PVG [13]. Although there were instances where the PVG *vs.* linear mapping rate difference was small, predicting this in advance of any analysis of an uncharacterised sample would be challenging.

As an example of where PVGs could be useful, ASFV is paramount due to its dynamic genome. ASFV genomes are dsDNA molecules of 170.1–193.9 kb in length with 150–167 ORFs on both strands [73,74,77]. ASFV is in the nucleocytoplasmic large dsDNA virus (NCLDV) superfamily and the Asfarvirus family. Notably, the ASFV genome contains nested genes within larger ORFs that are embedded in polycistronic arrays, rather than a monocistronic one [78]. Across all genotypes but with high sampling of genotypes I and II, ASFV has an open pangenome suggesting that broader sampling of ASFV populations will identify additional novel genes. This is based on a study of 42 genomes that had an open pangenome (alpha = 0.62) with 102 core and 168 accessory genes [73], which was extended by a study of 46 genomes that had 86 core and 324 accessory gene clusters [74]. The variation in these estimates is understandable given ASFV's high diversity and the comparatively small sample sizes in each investigation.

Based on the information above and with a view to identifying the most straightforward path for viral PVG studies, suggested key tools are listed to enable researchers new to PVGs to navigate this complex area (**Table 3**). The first four stages (PVG creation, analysis, visualisation, and openness) refer to PVGs themselves, whereas the final stage of read mapping is an application of PVGs to mutation detection. New PVG tools are emerging rapidly [79], and the following website maintains an updated list: https://github.com/colindaven/awesome-pangenomes.

Future prospects

PVGs should serve as the cornerstone for many population genomics studies, including multi-omics ones [80]. While PVG analysis and visualisation are an emerging field, significant software development and formalisation are needed to reduce reliance on bespoke scripts. One limitation of current PVG approaches is the continued use of linear VCF formats to represent mutations. This practice limits the output's utility, suggesting the need for the development of a dedicated PVG-based format for VCF files [36]. Moreover, there is a growing need for enhanced capacity to handle nested variants and optimise error detection across large sample collections and diverse sequencing technologies. Linked to this is the opportunity to create a wider array of PVG visualisation options that are scalable and rapid, present layers of information effectively, and permit non-experts to view mutations of interest. The small size of viral genomes means that they offer an efficient sandbox for PVG

Table 3 Key tools suggested for viral PVG analyses

Tool	Stage	Task	Ref.
PGGB	Creation	PVG construction	[9]
ODGI	Analysis	PVG analysis and manipulation	[39]
SequenceTubeMap	Visualisation	PVG visualisation	[46]
Panacus	PVG openness	PVG openness	[67]
Pangrowth	PVG openness	PVG openness	[66]
ODGI heaps	PVG openness	PVG size	[40]
VG	PVG openness	PVG indexing	[9]
Giraffe	Read mapping	Short-read mapping	[20]
GraphAligner	Read mapping	Long-read mapping	[54]

Note: PGGB, PanGenome Graph Builder; ODGI, Optimized Dynamic Genome/Graph Implementation.

visualisation development. These all would significantly improve the effectiveness of PVGs in advancing scientific investigations.

One promising avenue to overcome some of these limitations is the use of k-mer-based methods. Such approaches are more efficient and scalable than alignment-based ones, so they offer a powerful way to quantify diversity and rates of genetic change in large collections. At the same time, k-mer-based methods risk lower accuracy and sensitivity in detailed analyses such as detecting specific mutations due to shared k-mers and the value of k used [81]. For the latter, alignment-based approaches are more effective. Emerging k-mer tools, such as PanKmer [64], enable reference-free k-mer-based analyses, while other methods index all k-mers in the input reads to construct the PVG [82]. These k-mer-based approaches can be computationally more efficient, making them more scalable [83]. An example is the PVG indexing method called Maximal Exact Match Ordered (MEMO), which uses k-mers to allow rapid k-mer matching queries at a variety of scales [84]. As sequencing error rates continue to decline, these methods will also become more accurate [18].

A key challenge in PVG analysis is managing the sheer volume of mutations included in the graph, which can place significant strain on computational resources [85]. This has led to the development of reference PVGs [13,28,34]. However, the reference set of genomes used to create a PVG should be carefully selected to align with the specific scientific question being addressed. A promising solution is the use of "personalised" PVGs, where haplotypes are chosen based on patterns of k-mer count matching in read data. These personalised PVGs outperform those based on common variants or the entire sample set [86]. Ideally, these PVGs would be composed of hybrid assemblies generated from both short- and long-read sequencing technologies. Fortunately, viruses present fewer computational and phasing challenges for this k-mer matching process, making them well-suited for these advanced approaches. Importantly, methods that can handle the small lengths ($<$ 1 kb) of certain viral sequences not requiring large numbers of computational threads have emerged [87] and have been applied to viral data [88]. At the same time, many viral genomes are complex (*e.g.*, circular) or may be AT-rich, thus requiring a different approach when considering "personalised" PVGs.

A final note is that PVGs may not be effective for genetically divergent novel samples without a closely related comparator genome. Fortunately, the error rates of viral long-read sequencing using Oxford Nanopore Technology (ONT) and Pacific Biosciences (PacBio) are continuing to decrease, suggesting that this may be a useful option. This can be integrated with PVG-based approaches: it is now possible to augment a PVG to incorporate a new sample's long-read library (https://github.com/ldenti/palss). Moreover, genotyping may be enhanced by the incorporation of high-quality long-read assemblies, for which PVGs are effective visualisation aids.

CRediT author statement

Tim Downing: Conceptualisation, Investigation, Writing – original draft, Writing – review & editing. The author has read and approved the final manuscript.

Competing interests

The author declares no competing interests.

Acknowledgments

The Pirbright Institute receives support from UK Research and Innovation (UKRI) Biotechnology and Biological Sciences Research Council (BBSRC) (Grant Nos. BBS/E/PI/230002A, BBS/E/PI/230002B, BBS/E/PI/230002C, BBS/E/PI/23NB0003, and BBS/E/PI/23NB0004). I thank the anonymous reviewers for their constructive comments and feedback.

ORCID

0000-0002-8385-6730 (Tim Downing)

References

[1] Sigaux F. Génome du cancer ou de la construction des cartes d'identité moléculaire des tumeurs. Bull Acad Natl Med 2000;184:1441–7.
[2] Eizenga JM, Novak AM, Sibbesen JA, Heumos S, Ghaffaari A, Hickey G, et al. Pangenome graphs. Annu Rev Genomics Hum Genet 2020;21:139–62.
[3] Paten B, Eizenga JM, Rosen YM, Novak AM, Garrison E, Hickey G. Superbubbles, ultrabubbles, and cacti. J Comput Biol 2018;25:649–63.
[4] Chen NC, Solomon B, Mun T, Iyer S, Langmead B. Reference flow: reducing reference bias using multiple population genomes. Genome Biol 2021;22:8.
[5] 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;21:41–9.
[6] Rakocevic G, Semenyuk V, Lee WP, Spencer J, Browning J, Johnson IJ, et al. Fast and accurate genomic analyses using genome graphs. Nat Genet 2019;51:354–62.
[7] Hickey G, Heller D, Monlong J, Sibbesen JA, Sirén J, Eizenga J, et al. Genotyping structural variants in pangenome graphs using the vg toolkit. Genome Biol 2020;21:35.
[8] Boehm E, Kronig I, Neher RA, Eckerle I, Vetter P, Kaiser L, et al. Novel SARS-CoV-2 variants: the pandemics within the pandemic. Clin Microbiol Infect 2021;27:1109–17.
[9] Garrison E, Guarracino A. Unbiased pangenome graphs. Bioinformatics 2023;39:btac743.
[10] Rick JA, Brock CD, Lewanski AL, Golcher-Benavides J, Wagner CE. Reference genome choice and filtering thresholds jointly influence phylogenomic analyses. Syst Biol 2024;73:76–101.
[11] Valiente-Mullor C, Beamud B, Ansari I, Francés-Cuesta C, García-González N, Mejía L, 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:e1008678.
[12] Baaijens JA, Aabidine AZE, Rivals E, Schönhuth A. *De novo* assembly of viral quasispecies using overlap graphs. Genome Res 2017;27:835–48.

[13] Duchen D, Clipman SJ, Vergara C, Thio CL, Thomas DL, Duggal P, et al. A hepatitis B virus (HBV) sequence variation graph improves alignment and sample-specific consensus sequence construction. PLoS One 2024; 19:e0301069.
[14] Alser M, Rotman J, Deshpande D, Taraszka K, Shi H, Baykal PI, et al. Technology dictates algorithms: recent developments in read alignment. Genome Biol 2021;22:249.
[15] Lischer HEL, Shimizu KK. Reference-guided *de novo* assembly approach improves genome reconstruction for related species. BMC Bioinformatics 2017;18:474.
[16] Bradley P, den Bakker HC, Rocha EPC, McVean G, Iqbal Z. Ultrafast search of all deposited bacterial and viral genomic data. Nat Biotechnol 2019;37:152–9.
[17] Ondov BD, Treangen TJ, Melsted P, Mallonee AB, Bergman NH, Koren S, et al. Mash: fast genome and metagenome distance estimation using MinHash. Genome Biol 2016;17:132.
[18] Stoler N, Nekrutenko A. Sequencing error profiles of Illumina sequencing instruments. NAR Genom Bioinform 2021;3:lqab019.
[19] Baaijens JA, Bonizzoni P, Boucher C, Della Vedova G, Pirola Y, Rizzi R, et al. Computational graph pangenomics: a tutorial on data structures and their applications. Nat Comput 2022;21:81–108.
[20] Sirén J, Monlong J, Chang X, Novak AM, Eizenga JM, Markello C, et al. Pangenomics enables genotyping of known structural variants in 5202 diverse genomes. Science 2021;374:abg8871.
[21] Garrison E, Sirén J, Novak AM, Hickey G, Eizenga JM, Dawson ET, et al. Variation graph toolkit improves read mapping by representing genetic variation in the reference. Nat Biotechnol 2018;36:875–9.
[22] Novak AM, Garrison E, Paten B. A graph extension of the positional Burrows-Wheeler transform and its applications. Algorithms Mol Biol 2017;12:18.
[23] Groza C, Schwendinger-Schreck C, Cheung WA, Farrow EG, Thiffault I, Lake J, et al. Pangenome graphs improve the analysis of structural variants in rare genetic diseases. Nat Commun 2024;15:657.
[24] Gupta PK. GWAS for genetics of complex quantitative traits: genome to pangenome and SNPs to SVs and k-mers. Bioessays 2021;43:e2100109.
[25] Li R, Gong M, Zhang X, Wang F, Liu Z, Zhang L, et al. A sheep pangenome reveals the spectrum of structural variations and their effects on tail phenotypes. Genome Res 2023;33:463–77.
[26] Gao L, Gonda I, Sun H, Ma Q, Bao K, Tieman DM, et al. The tomato pan-genome uncovers new genes and a rare allele regulating fruit flavor. Nat Genet 2019;51:1044–51.
[27] Zhao X, Collins RL, Lee WP, Weber AM, Jun Y, Zhu Q, et al. Expectations and blind spots for structural variation detection from long-read assemblies and short-read genome sequencing technologies. Am J Hum Genet 2021;108:919–28.
[28] Liao WW, Asri M, Ebler J, Doerr D, Haukness M, Hickey G, et al. A draft human pangenome reference. Nature 2023;617:312–24.
[29] Haga IR, Shih BB, Tore G, Polo N, Ribeca P, Gombo-Ochir D, et al. Sequencing and analysis of Lumpy skin disease virus whole genomes reveals a new viral subgroup in West and Central Africa. Viruses 2024;16:557.
[30] Garrison E, Guarracino A, Heumos S, Villani F, Bao Z, Tattini L, et al. Building pangenome graphs. Nat Methods 2024; 21:2008–12.
[31] Guarracino A, Mwaniki N, Marco-Sola S, Garrison E. wfmash: whole-chromosome pairwise alignment using the hierarchical wavefront algorithm. Zenodo 2024.
[32] Garrison E, Guarracino A, Heumos S, Villani F, Bao Z, Tattini L, et al. smoothxg: normalization of variation graphs with local partial order realignment. Zenodo 2022.
[33] Heumos S, Heuer ML, Hanssen F, Heumos L, Guarracino A, Heringer P, et al. Cluster-efficient pangenome graph construction with nf-core/pangenome. Bioinformatics 2024; 40:btae609.
[34] Li H, Feng X, Chu C. The design and construction of reference pangenome graphs with minigraph. Genome Biol 2020;21:265.
[35] Li H. Minimap2: pairwise alignment for nucleotide sequences. Bioinformatics 2018;34:3094–100.
[36] Hickey G, Monlong J, Ebler J, Novak AM, Eizenga JM, Gao Y. Pangenome graph construction from genome alignments with Minigraph-Cactus. Nat Biotechnol 2024;42:663–73.
[37] Armstrong J, Hickey G, Diekhans M, Fiddes IT, Novak AM, Deran A, et al. Progressive Cactus is a multiple-genome aligner for the thousand-genome era. Nature 2020; 587:246–51.
[38] Kunyavskaya O, Prjibelski AD. SGTK: a toolkit for visualization and assessment of scaffold graphs. Bioinformatics 2019;35:2303–5.
[39] Guarracino A, Heumos S, Nahnsen S, Prins P, Garrison E. ODGI: understanding pangenome graphs. Bioinformatics 2022;38:3319–26.
[40] Moeckel C, Mareboina M, Konnaris MA, Chan CSY, Mouratidis I, Montgomery A, et al. A survey of k-mer methods and applications in bioinformatics. Comput Struct Biotechnol J 2024;23:2289–303.
[41] Chandra G, Gibney D, Jain C. Haplotype-aware sequence alignment to pangenome graphs. Genome Res 2024; 34:1265–75.
[42] Audano PA, Sulovari A, Graves-Lindsay TA, Cantsilieris S, Sorensen M, Welch AE, et al. Characterizing the major structural variant alleles of the human genome. Cell 2019; 176:663–75.e19.
[43] Li H. Gfatools: tools for manipulating sequence graphs in the GFA and rGFA formats. GitHub 2024.
[44] Gonnella G, Kurtz S. GfaPy: a flexible and extensible software library for handling sequence graphs in Python. Bioinformatics 2017;23:btx398.
[45] Cumer T, Milia S, Leonard AS, Pausch H. PG-SCUnK: measuring pangenome graph representativeness using single-copy and universal K-mers. BMC Bioinformatics 2025;27:29.
[46] Beyer W, Novak AM, Hickey G, Chan J, Tan V, Paten B, et al. Sequence tube maps: making graph genomes intuitive to commuters. Bioinformatics 2019;35:5318–20.
[47] Mikheenko A, Kolmogorov M. Assembly Graph Browser: interactive visualization of assembly graphs. Bioinformatics 2019;35:3476–8.
[48] Wick RR, Schultz MB, Zobel J, Holt KE. Bandage: interactive visualization of *de novo* genome assemblies. Bioinformatics 2015;31:3350–2.

[49] Gonnella G, Niehus N, Kurtz S. GfaViz: flexible and interactive visualization of GFA sequence graphs. Bioinformatics 2019;35:2853–5.
[50] Rausch T, Snajder R, Leger A, Simovic M, Giurgiu M, Villacorta L, et al. Long-read sequencing of diagnosis and post-therapy medulloblastoma reveals complex rearrangement patterns and epigenetic signatures. Cell Genom 2023; 3:100281.
[51] Chen S, Krusche P, Dolzhenko E, Sherman RM, Petrovski R, Schlesinger F, et al. Paragraph: a graph-based structural variant genotyper for short-read sequence data. Genome Biol 2019;20:291.
[52] Eggertsson HP, Kristmundsdottir S, Beyter D, Jonsson H, Skuladottir A, Hardarson MT, et al. GraphTyper2 enables population-scale genotyping of structural variation using pangenome graphs. Nat Commun 2019;10:5402.
[53] Ebler J, Ebert P, Clarke WE, Rausch T, Audano PA, Houwaart T, et al. Pangenome-based genome inference allows efficient and accurate genotyping across a wide spectrum of variant classes. Nat Genet 2022;54:518–25.
[54] Rautiainen M, Marschall T. GraphAligner: rapid and versatile sequence-to-graph alignment. Genome Biol 2020;21:253.
[55] Vaddadi K, Srinivasan R, Sivadasan N. Read mapping on genome variation graphs. In: Huber KT, Gusfield D, editors. Proceedings of the 19th International Workshop on Algorithms in Bioinformatics (WABI 2019). Dagstuhl: Schloss Dagstuhl–Leibniz-Zentrum für Informatik; 2019, p.7.
[56] Rajput J, Chandra G, Jain C. Co-linear chaining on pangenome graphs. Algorithms Mol Biol 2024;19:4.
[57] Li H, Handsaker B, Wysoker A, Fennell T, Ruan J, Homer N, et al. The Sequence Alignment/Map format and SAMtools. Bioinformatics 2009;25:2078–9.
[58] Sirén J, Välimäki N, Mäkinen V. Indexing graphs for path queries with applications in genome research. IEEE/ACM Trans Comput Biol Bioinform 2014;11:375–88.
[59] Garrison E, Marth G. Haplotype-based variant detection from short-read sequencing. arXiv 2012;1207.3907.
[60] Poplin R, Chang PC, Alexander D, Schwartz S, Colthurst T, Ku A, et al. A universal SNP and small-indel variant caller using deep neural networks. Nat Biotechnol 2018;36:983–7.
[61] Danecek P, Bonfield JK, Liddle J, Marshall J, Ohan V, Pollard MO, et al. Twelve years of SAMtools and BCFtools. Gigascience 2021;10:giab008.
[62] McKenna A, Hanna M, Banks E, Sivachenko A, Cibulskis K, Kernytsky A, et al. The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. Genome Res 2010;20:1297–303.
[63] Tettelin H, Masignani V, Cieslewicz MJ, Donati C, Medini D, Ward NL, et al. Genome analysis of multiple pathogenic isolates of *Streptococcus agalactiae*: implications for the microbial "pan-genome". Proc Natl Acad Sci U S A 2005; 102:13950–5.
[64] Aylward AJ, Petrus S, Mamerto A, Hartwick NT, Michael TP. PanKmer: k-mer-based and reference-free pangenome analysis. Bioinformatics 2023;39:btad621.
[65] Decano AG, Downing T. An *Escherichia coli* ST131 pangenome atlas reveals population structure and evolution across 4,071 isolates. Sci Rep 2019;9:17394.
[66] Parmigiani L, Wittler R, Stoye J. Revisiting pangenome openness with k-mers. Peer Community J 2024;4:e47.
[67] Parmigiani L, Garrison E, Stoye J, Marschall T, Doerr D. Panacus: fast and exact pangenome growth and core size estimation. Bioinformatics 2024;40:btae720.
[68] Eliseev A, Gibson KM, Avdeyev P, Novik D, Bendall ML, Pérez-Losada M, et al. Evaluation of haplotype callers for next-generation sequencing of viruses. Infect Genet Evol 2020;82:104277.
[69] Baaijens JA, Stougie L, Schönhuth A. Strain-aware assembly of genomes from mixed samples using flow variation graphs. In: Schwartz R, edtor. Research in Computational Molecular Biology. RECOMB 2020. Cham: Springer; 2020:12074.
[70] Baaijens JA, Van der Roest B, Köster J, Stougie L, Schönhuth A. Full-length *de novo* viral quasispecies assembly through variation graph construction. Bioinformatics 2019;35:5086–94.
[71] Queiroz VF, Carvalho JVRP, de Souza FG, Lima MT, Santos JD, Rocha KLS, et al. Analysis of the genomic features and evolutionary history of Pithovirus-like isolates reveals two major divergent groups of viruses. J Virol 2023; 97:e0041123.
[72] Dziadkiewicz P, Dojer N. Getting insight into the pangenome structure with PangTree. BMC Genomics 2020; 21:274.
[73] Wang Z, Jia L, Li J, Liu H, Liu D. Pan-genomic analysis of African swine fever virus. Virol Sin 2020;35:662–5.
[74] Wang L, Luo Y, Zhao Y, Gao GF, Bi Y, Qiu HJ. Comparative genomic analysis reveals an 'open' pan-genome of African swine fever virus. Transbound Emerg Dis 2020; 67:1553–62.
[75] Rodríguez-Pérez R, Fernández L, Marco S. Overoptimism in cross-validation when using partial least squares-discriminant analysis for omics data: a systematic study. Anal Bioanal Chem 2018;410:5981–92.
[76] Campillo-Balderas JA, Lazcano A, Becerra A. Viral genome size distribution does not correlate with the antiquity of the host lineages. Front Ecol Evol 2015;3:143.
[77] Truong QL, Nguyen TL, Nguyen TH, Shi J, Vu HLX, Lai TLH, et al. Genome sequence of a virulent African swine fever virus isolated in 2020 from a domestic pig in Northern Vietnam. Microbiol Resour Announc 2021; 10:e00193-21.
[78] Torma G, Tombácz D, Csabai Z, Moldován N, Mészáros I, Zádori Z, et al. Combined short and long-read sequencing reveals a complex transcriptomic architecture of African swine fever virus. Viruses 2021;13:579.
[79] Xiao J, Zhang Z, Wu J, Yu J. A brief review of software tools for pangenomics. Genomics Proteomics Bioinformatics 2015;13:73–6.
[80] Downing T, Angelopoulos N. A primer on correlation-based dimension reduction methods for multi-omics analysis. J R Soc Interface 2023;20:20230344.
[81] Denti L, Bonizzoni P, Brejova B, Chikhi R, Krannich T, Vinar T, et al. Pangenome graph augmentation from unassembled long reads. bioRxiv 2025;637057.
[82] Dilthey A, Cox C, Iqbal Z, Nelson MR, McVean G. Improved genome inference in the MHC using a population reference graph. Nat Genet 2015;47:682–8.

[83] Derelle R, von Wachsmann J, Mäklin T, Hellewell J, Russell T, Lalvani A, et al. Seamless, rapid, and accurate analyses of outbreak genomic data using split k-mer analysis. Genome Res 2024;34:1661–73.
[84] Hwang S, Brown NK, Ahmed OY, Jenike KM, Kovaka S, Schatz MC, et al. MEM-based pangenome indexing for k-mer queries. Algorithms Mol Biol 2025;20:3.
[85] Pritt J, Chen NC, Langmead B. FORGe: prioritizing variants for graph genomes. Genome Biol 2018;19:220.
[86] Sirén J, Eskandar P, Ungaro MT, Hickey G, Eizenga JM, Novak AM, et al. Personalized pangenome references. Nat Methods 2024;21:2017–23.
[87] Tennakoon C, Freville T, Downing T. Panalyze: automated virus pangenome variation graph construction, analysis and annotation. Bioinform Adv 2026;6:vbag071.
[88] Downing T, Tennakoon C, Lasecka-Dykes L, Wright C. Using pangenome variation graphs to improve mutation detection in a large DNA virus. Microb Genom 2026;12:001698.