Building pangenome graphs

Erik Garrison[1*†], Andrea Guarracino[1,2†], Simon Heumos[3,4], Flavia Villani[1], Zhigui Bao[5], Lorenzo Tattini[6], Jörg Hagmann[7], Sebastian Vorbrugg[8], Santiago Marco-Sola[9,10], Christian Kubica[8], David G. Ashbrook[1], Kaisa Thorell[11], Rachel L. Rusholme-Pilcher[12], Gianni Liti[6], Emilio Rudbeck[13], Agnieszka A. Golicz[14], Sven Nahnsen[3,4], Zuyu Yang[15], Moses N. Mwaniki[16], Franklin L. Nobrega[17], Yi Wu[17], Hao Chen[18], Joep de Ligt[15], Peter H. Sudmant[19], Nicole Soranzo[20,21,22,23,2], Vincenza Colonna[1,24], Robert W. Williams[1] and Pjotr Prins[1]

[1]Department of Genetics, Genomics and Informatics, University of Tennessee Health Science Center, 71 S Manassas St, Memphis, 38163, Tennessee, USA. [2]Fondazione Human Technopole, Viale Rita Levi Montalcini, 20157 Milan, Italy. [3]Quantitative Biology Center (QBiC) Tübingen, University of Tübingen, Tübingen, Germany. [4]Biomedical Data Science, Dept. of Computer Science, University of Tübingen, Tübingen, Germany. [5]Shenzhen Branch, Guangdong Laboratory of Lingnan Modern Agriculture, Genome Analysis Laboratory of the Ministry of Agriculture and Rural Affairs, Agricultural Genomics Institute at Shenzhen, Chinese Academy of Agricultural Sciences, Buxin Road 97, Shenzhen, 518120, Guangdong, China. [6]Université Côte d'Azur, CNRS, INSERM, IRCAN, Nice, France. [7]Computomics GmbH, Eisenbahnstr. 1, 72072 Tübingen, Germany. [8]Department of Molecular Biology, Max Planck Institute for Biology, Max-Planck-Ring 9, 72076 Tübingen, Baden-Wuerttemberg, Germany. [9]Computer Sciences Department, Barcelona Supercomputing Center, Barcelona 08034, Spain. [10]Departament d'Arquitectura de Computadors i Sistemes Operatius, Universitat Autonoma de Barcelona, Barcelona 08193, Spain. [11]Chemistry and Molecular Biology, Faculty of Science, University of Gothenburg, Sweden. [12]Earlham Institute, Norwich Research Park, Colney Lane, Norwich, Norfolk, NR4 7UZ. UK. [13]Clinical Genomics Gothenburg, Bioinformatics and Data Centre, University of Gothenburg, Sweden. [14]Department of Plant Breeding, Justus Liebig University Giessen, Giessen, Germany. [15]The Institute of Environmental Science and Research, New Zealand. [16]Department of Computer Science, University of Pisa. [17]School of Biological Sciences, Faculty of Environmental and Life Sciences, University of Southampton, Southampton, UK. [18]Department of Pharmacology, Addiction Science and Toxicology, University of Tennessee Health Science Center, Memphis, TN. [19]Department of Integrative Biology, University of California Berkeley, Berkeley, CA. [20]Wellcome Sanger Institute, Genome Campus, Hinxton CB10 1SA, UK. [21]National Institute for Health Research Blood and Transplant Research Unit in Donor Health and Genomics, University of Cambridge, Cambridge, UK. [22]Department of Haematology, Cambridge Biomedical Campus, Cambridge CB2 0AW, UK. [23]British Heart Foundation Centre of Research Excellence, University of Cambridge, Cambridge, UK. [24]Institute of Genetics and Biophysics, National Research Council, Naples 80111, Italy.

*Corresponding author. E-mail: egarris5@uthsc.edu;
†These authors contributed equally to this work.

Abstract

Pangenome graphs can represent all variation between multiple genomes, but existing methods for constructing them are biased due to reference-guided approaches. In response, we developed the PanGenome Graph Builder (PGGB), a reference-free pipeline for constructing unbiased pangenome graphs. PGGB uses all-to-all whole-genome alignments and learned graph embeddings to build and iteratively refine a model in which we can identify variation, measure conservation, detect recombination events, and infer phylogenetic relationships.

Keywords: pangenomes, genome alignment, variant detection, comparative genomics, chromosome evolution, phylogenetics, population genetics

Pangenome graphs compactly represent complete genomes, their homologies, and all forms of variation between them [1–4]. They allow us to identify variation, measure conservation, detect recombination events, and infer phylogenetic relationships, making them valuable tools for studying sequence evolution and variation in diverse species [5, 6]. However, existing methods for constructing pangenome graphs [7, 8] are biased due to their reference and tree-guided approaches [5, 9], which can lead to incomplete and unstable representations of genetic variation [10]. Inductive biases result from techniques to mitigate computational complexity [5, 7], or from a goal to structure the resulting graphs so that they are easier to use during read alignment [8]. The unbiased construction of pangenome graphs implies all-to-all comparisons that scale quadratically with the number of included genomes [10]. Although approaches for unbiased pangenome graph construction have been proposed [10, 11], these have been limited to the graph induction step, and specialized techniques for pangenome alignment and refinement are required to avoid systematic biases and obtain high-quality pangenome builds [12].

To overcome these limitations, we propose the PanGenome Graph Builder (PGGB), a reference-free pipeline to construct unbiased pangenome graphs. Its output presents a base-level representation of the pangenome, including variants of all scales from single nucleotide polymorphisms (SNPs) to structural variants (SVs). The graph is unbiased, i.e., all genomes are treated equivalently, regardless of input order or phylogenetic dependencies, and any genome may be used as a frame of reference in downstream analysis. PGGB makes no assumptions about phylogenetic relationships, orthology groups, or evolutionary histories, allowing data to speak for itself without risks of implicit biases that may affect inferences made from the graph. PGGB is implemented as a modular shell script, integrating independent components via standard text-based file formats, which provides a template for future pangenome construction methods. The method is practical, scalable to hundreds of genomes, and has been proven accurate through years of development in the Human Pangenome Reference Consortium (HPRC) [12, 16] and in the broader bioinformatics community [17–20]. Here, we describe the specific innovations in the three main phases of the algorithm: alignment, graph creation, and graph normalization. We then use cross-validation with MUMMER4 [21] to demonstrate the accuracy of our approach across a wide range of species and scales.

PGGB begins with sequence alignment (Figure 1A). To avoid reference and order bias, PGGB uses an all-to-all alignment of the input sequences. This approach aligns sequences directly to each other, enabling each sequence in the pangenome to serve as a potential reference to describe variation. To obtain alignments, PGGB uses WFMASH [22]. WFMASH first applies an extension of MASHMAP [23] to obtain high-level homology mappings, by default using seeds of 5kbp to find homologies of 25kbp or more at 90% average nucleotide identity. WFMASH then uses a generalization of the bidirectional Wavefront Algorithm (BiWFA) [24, 25] that aligns the sequences by comparing segments of 256bp rather than single characters. This algorithm, BiWFλ—so named

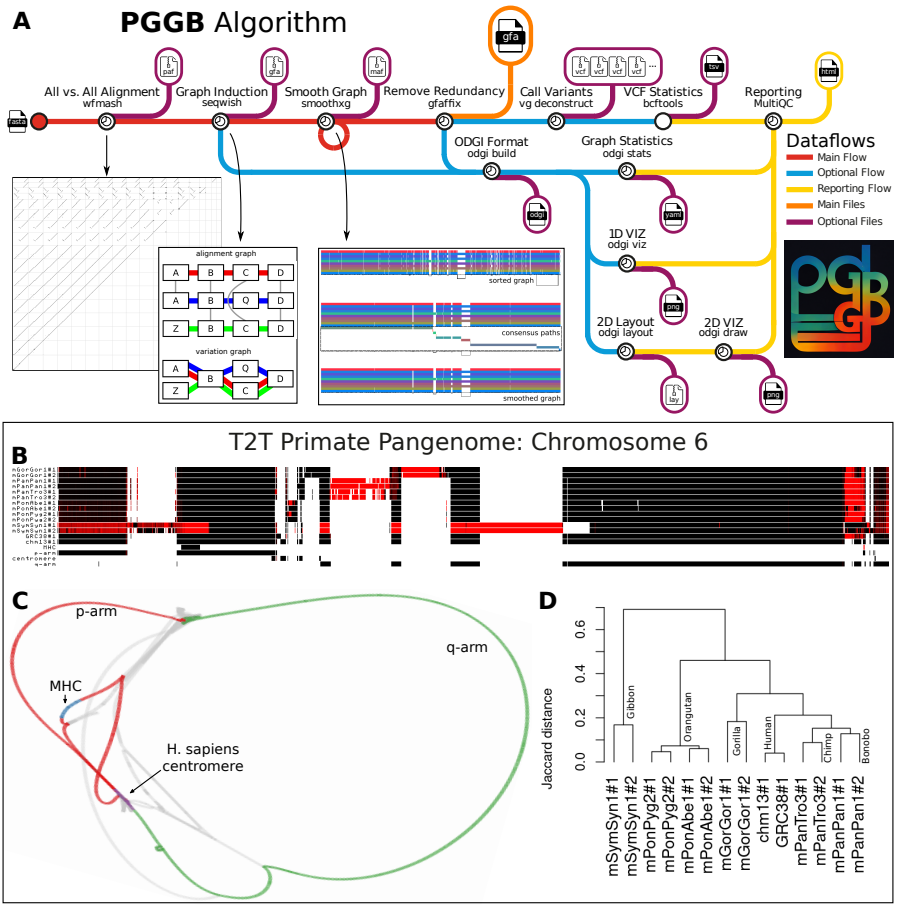

Fig. 1: **PGGB and applications.** (A) PGGB's algorithms/data flows. Primary flow (red) proceeds from FASTA to alignment, graph induction, smoothing, to normalization with GFAF-FIX, ending with final graph (orange). Optional outputs (blue): statistics, variant calls, and 1D/2D graph visualizations. (B) 1D pangenome graph visualization using 14 haplotype-resolved primate assemblies homologous to human chromosome 6. T2T-CHM13 annotations (Major Histocompatibility Complex, p-arm, centromere, q-arm) are shown. The p-arm region with the MHC is inverted in Gibbon. Centromeric regions appear largely non-homologous among many species, with the exception of between chimpanzee and bonobo and between the orangutans. (C) 2D visualization, rendered with the same human chromosomal annotations in GFAESTUS [13], shows possible circularization due to structural variation or homogenization between subtelomeric regions. (D) Using ODGI [6], we extract a pairwise distance matrix based on in-graph Jaccard metrics over shared base pairs. This distance matrix yields a phylogenetic tree that matches previous results based on SNPs [14]. We posit that the greater phylogenetic distances reflect the inclusion of the centromeres—which feature low rates of recombination and tend to diversify rapidly by near-clonal evolution [15]—in our distance computation.

because it replaces character match with a callback function λ that matches segments—obtains a final base level alignment by splicing together "incepted" alignments over the 256bp segment pairs that lie in the optimal alignment path. Our use of WFMASH ensures that the alignments which structure the graph feature long-range collinearity that is insensitive to repetitive, shorter homologies found between transposons and satellite sequences. Although WFMASH alignments have an ideal structure for its operation, PGGB can build the graph using any set of user-defined alignments in PAF format.

The second step—pangenome graph induction—converts a collection of genomes and pairwise alignments between them into an equivalent variation graph. We achieve this with SEQWISH [10], a tool specifically designed to scale graph induction to whole pangenomes in low memory. At a high level, SEQWISH merges all DNA base-pairs that are matched together in the alignments into a single character in the output graph. This process also compresses transitively-matched base-pairs. For example, if A, B, and C are characters in input sequences and $\rightarrow$ represents a character match, $A \rightarrow B \rightarrow C$ would result in a single character in the output graph that also implies the transitive match $A \rightarrow C$. SEQWISH evokes this transitive closure while retaining the mapping between input sequences and the output graph, allowing it to losslessly embed the input sequences as paths through the resulting graph. Any single input genome is faithfully and fully embedded in the graph and can be completely extracted by tracing labeled paths through the nodes. The SEQWISH graph thus recovers transitive homology relationships that may not be present in the initial alignment set. This property allows us to apply random sparsification to reduce the complexity of very large alignment problems.

For large inputs, PGGB can use a heuristic based on the Erdős–Rényi model of random graphs to set a sparsification threshold for initial mappings. This model leads us to expect a giant component, or connected subgraph that contains a significant portion of the nodes in the graph, to arise in a random graph of N nodes when the probability of edges between two nodes is $P_{critical} = 1/(N-1)$ [26]. Considering the SEQWISH alignment graph (Figure 1A), where nodes correspond to subsequences and edges to mappings, we seek to ensure that a giant component exists for all homologous collections of nodes in all regions of the pangenome. This will let us reconstruct all transitive relationships in the variation graph without needing to directly compute all pairwise alignments. We thus set a sparsification parameter that uses a hash of each mapping record to filter mappings with a probability $P_{sparse} \gg P_{critical}$, allowing us to avoid the expected $O(N^2)$ costs implied when $P = 1$. This allows us to dramatically reduce the runtime of alignment and graph induction with negligible effect on accuracy (Table 1), e.g. 10× increase in the number of genomes requires only 20× increase in runtime—rather than 100×.

Graph building completes with SMOOTHXG (Figure 1A), an iterative post-processing step specifically designed for PGGB that locally compresses and simplifies the pangenome graph. Although the SEQWISH graph presents a complete, lossless model of the input genomes and their homologies, in our

experience it often presents complex local motifs that can cause problems for diverse types of downstream analysis. A key issue is that pairwise alignments derived across input sequences are not mutually normalized, leading to different representations of small variants like indels in low-complexity sequences, which in turn generate complex looping motifs that are difficult to process. We mitigate this issue by removing short matches from SEQWISH's input alignments. This reduces complexity, but also creates a graph that can be locally "under-aligned" and does not represent all local pairwise relationships. To resolve this, we apply a local realignment kernel, partial order alignment (POA) [27–29], across all parts of the graph. By default, we do so at a scale of around 1kbp, which is smaller than most nonlinear patterns of structural variation found in genomes [12, 30]. This allows the PGGB graph to represent complex structurally-variable loci as simple loops through a single copy of duplicated sequences [12]. The kernel is applied to regions that are extracted from a 1-dimensional graph embedding [6]. This embedding orders nodes in the graph so that their distance in the order best-approximates their distance in the genomic paths of the graph. SMOOTHXG first learns this embedding, then obtains partially overlapping segments of the graph (blocks) to which it then applies POA. The realigned blocks are "laced" back into a complete variation graph. We iterate the entire SMOOTHXG step multiple times (3 by default) to limit edge effects that can occur near block boundaries, progressively refining the learned graph embedding. As a final normalization step, we

Pangenome	Size (bp)	Compr.	Time (m)	Mem. (GB)	SNVs	F-score
athaliana7.chr1	210174177	5.12	28.51	9.71	129374	0.877267
ecoli50	249520474	12.56	89.35	12.97	56915	0.947041
ecoli500*	2572341327	23.99	1816.66	134.59	58259	0.936551
hsapiens90.chr6	15508376475	81.17	1183.33	135.52	143972	0.971475
mouse17.chr19	994731502	11.52	203.80	29.48	223951	0.907288
primate14.chr6	2635610277	6.18	1742.37	61.38	2886064	0.909077
scerevisiae8	96255507	6.47	8.78	3.53	53742	0.968729
scerevisiae142	1702093905	55.29	1021.81	112.98	62796	0.955988
scerevisiae142*	1702093905	41.41	562.89	75.91	62580	0.955650
soy37.chr18	2240871558	20.50	599.66	29.17	101907	0.907878
tomato23.chr2	1280460312	20.69	78.53	43.84	39243	0.948173

Table 1: **Performance of PGGB with pangenomes across species.**
For each pangenome, we report its size, the compression ratio (pangenome sequence length divided by graph size), the PGGB runtime, the maximum memory usage of PGGB, the average number of SNVs (across all haplotypes except the one used as reference) identified with MUMMER4 that we used to evaluate SNVs identified with PGGB, and the average F-score (across all haplotypes except the one used as reference) computed using MUMMER4's SNVs as ground truth. The name of each pangenome indicates the species and the number of haplotypes. All runs were performed on machines equipped with AMD EPYC 7402P 24-Core, 378 GB of RAM, and a 1 TB Solid-State Drive. All PGGB runs were executed with 48 threads. *Erdős–Rényi random sparsification activated.

apply GFAFFIX to compress redundant nodes [31] and use ODGI to make a final sort for the modified graph [6].

PGGB provides outputs that support immediate interpretation, quality control, and downstream applications. Using ODGI, it produces basic graph statistics, such as size, node count, and base content. ODGI creates 1D and 2D visualizations that provide intuition about the structure of the entire graph, with the 1D view showing the relative alignment of paths into the graph structure, and the 2D view showing high-level features of the graph topology. Both can be applied at the scale of multi-gigabasepair graphs. Optionally, PGGB provides graph statistics and diagnostic plots in a MultiQC report [32]. We also provide an option to call variants [1, 33] from the graph to produce a phased description of embedded haplotypes in variant call format (VCF). Variants called directly from the graph can include large nested genetic sites, leading to incompatibility with many applications. To address this, PGGB decomposes complex nested variation into a minimal reference-relative representation using BiWFA [34]. This allows PGGB to provide input to analyses based on small variants, leading to compatibility with numerous downstream applications based on genomic variation. PGGB is thus a multi-sample variant caller for whole-genome assemblies.

PGGB has been applied and validated at large scale in projects in the HPRC [12], where it additionally has provided the first sequence-based evidence for systematic recombination between heterologous acrocentric human chromosomes [16]. Here, to demonstrate PGGB's broad utility, we present results from its application to a variety of diverse pangenome and comparative genomic contexts (Table 1). We provide information on runtime and resource requirements, showing that even for hundreds of small genomes, PGGB can provide a variation graph within hours. Larger genomes in general require partitioning to maximize parallelism and ensure that total compute requirements fit in standard commodity servers (Section A.2). Due to lack of ground truth, quality evaluation on real data can be difficult. For validation, we compare PGGB's output with SNPs detected by MUMMER4 [21], a current standard approach for pairwise whole genome alignment. These show cross validation F-scores >90% across all tested contexts except *athaliana7.chr1*, indicating that the method performs equivalently to existing standards. However, while MUMMER4 provides only pairwise comparisons with a target reference, PGGB yields a full all-to-all comparison between genomes that leads to completely new bioinformatic analysis modalities.

Many downstream applications that are typically based on polarization of variants (e.g. SNPs) relative to a single reference genome may be directly implemented in the space of variation graphs built with PGGB and similar methods. This follows from two basic concepts: in the variation graph, nodes are alleles, while genomes can be simply understood as vectors of allele counts. Methods based in this vector space allow us to simultaneously consider *all* classes of variation in downstream analyses, without reference bias, an objective, which to our knowledge has never been achieved before in bioinformatics

with the practical generality provided by PGGB. As proof of principle, we put forward a phylogenetic tree constructed directly from distances measured within a pangenome graph of 14 complete assemblies of chromosome six from the great ape family (Figure 1D), which matches established phylogenies of the *Hominoidea* clade based on manually curated sets of SNPs that exclude structurally variable regions [14].

PGGB is a new, modular, and conceptually straightforward approach to understanding sequence relationships between many complete genomes in both pangenomic and comparative genomics settings. Our approach provides a general framework for genome graph building techniques which we expect researchers will upgrade and extend in the future. By making it easy to build variation graphs, PGGB opens the door to diverse downstream population and evolutionary genetic methods that can consider all classes of sequence variation simultaneously. This will allow us to develop a comprehensive understanding of the links between sequence variation, phenotype, and evolution in an era where the complete assembly of genomes becomes routine.

Online content. PGGB is available at https://github.com/pangenome/pggb. Code used for experiments can be accessed at https://github.com/pangenome/pggb-paper. Pangenomes are available at https://doi.org/10.5281/zenodo.7937947.

Acknowledgments

The authors thank members of the HPRC Pangenome Working Group for their insightful discussion and feedback, and members of the HPRC production teams for their development of resources used in our exposition.

Funding

The authors gratefully acknowledge support from National Institutes of Health/NIDA U01DA047638 (E.G.), National Institutes of Health/NIGMS R01GM123489 (E.G. and P.P.), National Institutes of Health/NIGMS R35GM142916 (P.H.S), and NSF PPoSS Award #2118709 (E.G. and P.P.), and the Center for Integrative and Translational Genomics (E.G.). S.H. acknowledges funding from the Central Innovation Programme (ZIM) for SMEs of the Federal Ministry for Economic Affairs and Energy of Germany. This work was supported by the BMBF-funded de.NBI Cloud within the German Network for Bioinformatics Infrastructure (de.NBI) (031A532B, 031A533A, 031A533B, 031A534A, 031A535A, 031A537A, 031A537B, 031A537C, 031A537D, 031A538A). A.A.G. acknowledges the Alexander von Humboldt Foundation in the framework of Sofja Kovalevskaja Award and German Research Foundation (DFG) project number 497667402.

Conflict of interest

Author J.H. is employed by Computomics GmbH.

Author contributions

Project conception: Erik Garrison
Project guidance: Erik Garrison, Sven Nahnsen, Nicole Soranzo, Vincenza Colonna, Robert W. Williams, Pjotr Prins
Software development: Erik Garrison, Andrea Guarracino, Simon Heumos, Santiago Marco-Sola, Mwaniki N. Moses
Paper editing: Erik Garrison, Andrea Guarracino, Simon Heumos, Vincenza Colonna, Robert W. Williams, Pjotr Prins
Experimental design: Erik Garrison
Quality evaluation: Erik Garrison, Andrea Guarracino, Lorenzo Tattini
Testing: Erik Garrison, Andrea Guarracino, Simon Heumos, Flavia Villani, Zhigui Bao, Lorenzo Tattini, Jörg Hagmann, Sebastian Vorbrugg, Christian Kubica, Kaisa Thorell, Rachel L. Rusholme-Pilcher, Agnieszka A. Golicz, Sven Nahnsen, Zuyu Yang, Mwaniki N. Moses, Franklin L. Nobrega, Hao Chen, Joep de Ligt, Peter H. Sudmant
Experimental execution: Andrea Guarracino
Documentation: Andrea Guarracino, Simon Heumos
Mus musculus and Rattus Norvegicus: Flavia Villani, David G. Ashbrook, Hao Chen, Vincenza Colonna
Tomato pangenome: Zhigui Bao
S. cerevisiae and S. paradoxus: Lorenzo Tattini, Gianni Liti
Soy G.max: Jörg Hagmann
A. thaliana: Sebastian Vorbrugg, Christian Kubica
Parameter settings: Sebastian Vorbrugg
Algorithm development: Santiago Marco-Sola
Helicobacter pylori: Kaisa Thorell, Emilio Rudbeck
V. fava: Agnieszka A. Golicz
Neisseria mingitidis: Zuyu Yang, Joep de Ligt
SARS-CoV-2 dataset: Mwaniki N. Moses
E. coli and Coliphages: Franklin L. Nobrega, Yi Wu
Primate pangenome: Peter H. Sudmant
High Performance Computing management: Pjotr Prins

Appendix A Methods

Here we provide details about components which are not described in other publications. Our primary focus is on SMOOTHXG, which is the main "glue" that ties together the PGGB pipeline into a coherent system. Through a series of passes over the pangenome, SMOOTHXG reshapes the graph to reduce local complexity and underalignment. This resolves key problems encountered

in earlier attempts to implement all-vs-all alignment based graph construction [10, 35], which typically resulted in very complex, looping, graph motifs at small scales, and graph redundancy or loss of alignment sensitivity caused by match filtering. We additionally provide a description of the evaluation method we use in our cross-validation experiments where PGGB graphs are compared with SNPs determined by MUMMER4.

A.1 SMOOTHXG

SMOOTHXG requires a GFA pangenome graph as input, for example output from SEQWISH. The raw SEQWISH graph is globally unsorted and might be locally unaligned. SMOOTHXG sorts and normalizes the graph preserving nonlinear complex global structural variation. Detailed steps are described subsequently.

Preprocessing. A Path-Guided Stochastic Gradient Descent (PG-SGD) algorithm optimizes the one-dimensional (1D) node order of the graph to best match the nucleotide positions in the embedded paths. A grooming step ensures that for each node, the node orientation follows the consensus node orientation of all path steps visiting the node. A 1D topological sorting of the graph completes the overall sorting steps. Finally, the graph is chopped so that each node does contain a relatively little number of nucleotides (SMOOTHXG default: 100), preserving node topology and order. This prepares the graph for more exact cut points during the block creation process described in the next section.

Create blocks. The smoothable blocks are discovered by iterating over all nodes following the previously calculated order. A node is added to a block if its addition does not exceed the: 1. total path length limit of a block, 2. the maximum edge jump limit of a block, or 3. the maximum block length. Blocks are broken at likely Variable Number Tandem Repeat (VNTR) boundaries and to ensure that the path ranges within each block do not exceed the maximum sequence input size for the POA step described in the next section.

Smooth each block. For each block, padding extends each block to the left and right. This improves the local alignment at the boundaries of each block. The k-mer plus min-hash approach ensures that only unique sequences are passed to the POA step, which can significantly reduce runtime. POA is applied to each block. Optionally, this step generates a multiple sequence alignment in MAF format for each block. The padding is removed, and the block is saved for later integration into a full graph.

Lace blocks into the smoothed graph. The smoothed blocks are laced together to the final pangenome graph following their initial input order. As a final step, the graph is unchopped, preserving the maximum possible node lengths in the graph.

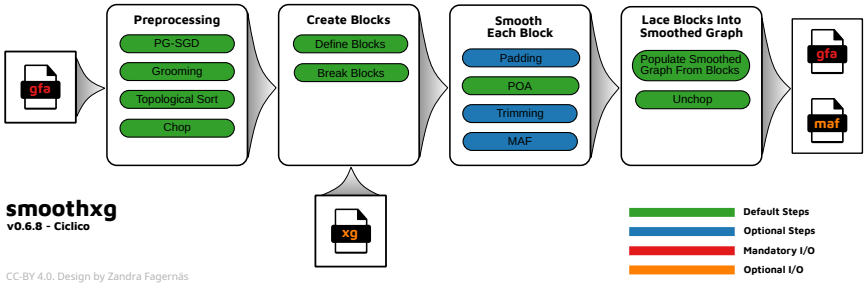

Fig. A1: Overview of the algorithmic steps in SMOOTHXG.

A.2 Sequence partitioning

Pangenome graphs can represent the mutual alignments within a collection of sequences. Nonetheless, it is unreasonable to expect that all sequences can be pairwise aligned together and obtain a graph with well-separated connected components. Instead, a large connected component is likely to form, along with a few smaller ones, due to inaccurate mappings or false homologies. This can result in increased computational complexity and hinder downstream analysis. Consequently, before applying PGGB, we can perform community detection with the input sequences to uncover the underlying structure of their mutual relationships and gather group of sequences that share common properties. For example, the identified communities (or groups of sequences) may correspond to the distinct chromosomes present within the input genomes. Indeed, by incorporating reference genomes into the set of input sequences, we can leverage the reference annotations to effectively partition sequences by chromosome. Once these groups of sequences have been identified, we can then partition the sequences by community and apply PGGB on each group separately.

To simplify the sequence partitioning process for users, we provide such a workflow as a dedicated shell script called "partition-before-pggb". Here is a summary of its key steps:

1. **Homology detection.** We perform pairwise mapping of the input sequences with WFMASH [22] to identify homologus regions, specifically focusing on their location, size and estimated identity. We do not require base-level alignment; by omitting this step, we accelerate the homology detection.
2. **Mapping graph construction.** We use the homology map to build a mapping graph. In contrast to a variation graph, each node in a mapping graph represents an input sequence, with edges representing mappings between these sequences. The edges are weighted, with each weighting factor given by the product of the length of the mapping and its estimated sequence identity.
3. **Community detection.** We apply the Leiden algorithm [36] to detect the underlying communities present in the mapping graph. In the context of community detection, a community is a group of nodes within a larger

graph that are more densely connected to each other than they are to the rest of the graph. The Leiden algorithm aims to maximize modularity by partitioning a graph into distinct, densely connected sets of nodes. Modularity is a measure to quantify the strength of the division of a graph into communities. It evaluates the density of links within communities compared to the density of links between communities. A higher modularity indicates a stronger and more defined community structure within the graph. By using a weighted mappping graph, we place more emphasis on the strongest homologies for community detection.

4. **Command generation.** For each community, we generate a complete PGGB command line in order to run it on each set of sequences separately. This eases the analysis and reduces the computational burden of building pangenome graphs.

It is important to highlight that if it is already known that the input sequences present particular rearrangements, such as rare chromosomal translocations, it may be advisable to skip the sequence partitioning and conduct the analysis with the full set of sequences.

A.3 Validation experiments

To evaluate the accuracy and reliability of our pangenome graph construction and variant calling methods, we designed a cross-validation approach that allowed us to compare the results obtained from the graph-based method (PGGB) against those generated by the widely-used pairwise alignment tool, NUCMER, in MUMMER4 [21].

The cross-validation process begins with the extraction of FASTA sequences from the pangenome graph GFA and preparation of reference sequences. Next, variants are identified using both PGGB (with VG) and nucmer (via a MUMMER4 script), generating a VCF file for each haplotype to ease comparison using the RealTime Genomics toolkit.

These variant files are then compared and evaluated, focusing on regions where both methods are able to call variants, an aspect that we found to be important in the HPRC cross-validation studies, wherein DIPCALL was used to find consistently-alignable regions in which comparisons were conceptually sound [12]. Finally, we collect metrics and statistics for further analysis and visualization. To simplify reproducibility, here we provide a detailed summary of the evaluation process:

1. **Extract sequences in FASTA**: The script extracts the reference paths in the GFA file and creates a new FASTA file containing these sequences.
2. **Identify variants with PGGB**: The script then identifies variants in the pangenome graph using the `vg deconstruct` tool with appropriate options for haplotype-based variant calling from the graph and complex allele decomposition with BiWFA and VCFLIB. The final variants are saved in a VCF format file.
3. **Pre-process the PGGB-based VCF files**: For compatibility with NUCmer, we pre-process the VCF files, normalizing alleles, removing

insertions and deletions larger than 1 base pair, and removing the ALT allele if it is not present in the haplotype.

4. **Identify variants with NUCmer**: The script performs a pairwise sequence alignment between the reference and each contig in the pangenome using NUCmer. The script extracts SNPs from the NUCmer delta file using the `show-snps` command and generates VCF files for each aligned contig.
5. **Merge variants by haplotype**: The script then merges all VCF files for each haplotype generated by NUCmer to create a multi-haplotype VCF file.
6. **Variant evaluation**: RTG Tools' `vcfeval` is used to evaluate the performance of PGGB-based variants and NUCmer-based variants by comparing true positives, false positives, and false negatives in shared callable regions. This is done for both "non-waved" and "waved" (BiWFA-normalized) PGGB-based VCF files, allowing for a direct comparison of the performance of these variant calling methods.
7. **Collect statistics**: The script computes summary statistics, such as precision, recall, and F-scores for each haplotype and writes them to TSV files. It also calculates the total number of variants called and the ratio of evaluated variants for both NUCmer and PGGB-based methods.
8. **Organize output**: Finally, the script organizes the output data, including VCF files, evaluation results, and statistics, into a specified output directory.

Although imperfect due to our lack of ground truth in the context of whole-genome alignment, this method provides a way to approximately compare the existing standard for whole-genome pairwise alignment, MUMMER4, with PGGB. We focus on SNPs and omit comparison of structural variation for diverse reasons. First, we found extracting SVs from MUMMER4 output to be problematic and poorly-supported. Second this issue remains difficult in genomics due to the multiple near-equivalent representations that a given structural variant allele may have. However, we have addressed these topics in the context of the HPRC paper [12], where significant resources were available to drive an independent evaluation of PGGB and other graph building methods.

References

[1] Garrison, E., Sirén, J., Novak, A.M., Hickey, G., Eizenga, J.M., Dawson, E.T., Jones, W., Garg, S., Markello, C., Lin, M.F., Paten, B., Durbin, R.: Variation graph toolkit improves read mapping by representing genetic variation in the reference. Nature Biotechnology **36**(9), 875–879 (2018). https://doi.org/10.1038/nbt.4227

[2] Paten, B., Novak, A.M., Eizenga, J.M., Garrison, E.: Genome graphs and the evolution of genome inference. Genome Res. **27**(5), 665–676 (2017). https://doi.org/10.1101/gr.214155.116

[3] Garrison, E.: Graphical pangenomics (2019). https://doi.org/10.17863/CAM.41621

[4] Eizenga, J.M., Novak, A.M., Sibbesen, J.A., Heumos, S., Ghaffaari, A., Hickey, G., Chang, X., Seaman, J.D., Rounthwaite, R., Ebler, J., Rautiainen, M., Garg, S., Paten, B., Marschall, T., Sirén, J., Garrison, E.: Pangenome graphs. Annual Review of Genomics and Human Genetics **21**(1), 139–162 (2020). https://doi.org/10.1146/annurev-genom-120219-080406

[5] Armstrong, J., Hickey, G., Diekhans, M., Fiddes, I.T., Novak, A.M., Deran, A., Fang, Q., Xie, D., Feng, S., Stiller, J., Genereux, D., Johnson, J., Marinescu, V.D., Alföldi, J., Harris, R.S., Lindblad-Toh, K., Haussler, D., Karlsson, E., Jarvis, E.D., Zhang, G., Paten, B.: Progressive cactus is a multiple-genome aligner for the thousand-genome era. Nature **587**(7833), 246–251 (2020). https://doi.org/10.1038/s41586-020-2871-y

[6] Guarracino, A., Heumos, S., Nahnsen, S., Prins, P., Garrison, E.: ODGI: understanding pangenome graphs. Bioinformatics (2022). https://doi.org/10.1093/bioinformatics/btac308

[7] Li, H., Feng, X., Chu, C.: The design and construction of reference pangenome graphs with minigraph. Genome Biology **21**(1) (2020). https://doi.org/10.1186/s13059-020-02168-z

[8] Hickey, G., Monlong, J., Ebler, J., Novak, A.M., Eizenga, J.M., Gao, Y., Abel, H.J., Antonacci-Fulton, L.L., Asri, M., Baid, G., Baker, C.A., Belyaeva, A., Billis, K., Bourque, G., Buonaiuto, S., Carroll, A., Chaisson, M.J.P., Chang, P.-C., Chang, X.H., Cheng, H., Chu, J., Cody, S., Colonna, V., Cook, D.E., Cook-Deegan, R.M., Cornejo, O.E., Diekhans, M., Doerr, D., Ebert, P., Ebler, J., Eichler, E.E., Fairley, S., Fedrigo, O., Felsenfeld, A.L., Feng, X., Fischer, C., Flicek, P., Formenti, G., Frankish, A., Fulton, R.S., Garg, S., Garrison, E., Garrison, N.A., Giron, C.G., Green, R.E., Groza, C., Guarracino, A., Haggerty, L., Hall, I.M., Harvey, W.T., Haukness, M., Haussler, D., Heumos, S., Hoekzema, K., Hourlier, T., Howe, K., Jain, M., Jarvis, E.D., Ji, H.P., Kenny, E.E., Koenig, B.A., Kolesnikov, A., Korbel, J.O., Kordosky, J., Koren, S., Lee, H., Lewis, A.P., Liao, W.-W., Lu, S., Lu, T.-Y., Lucas, J.K., Magalhães, H., Marco-Sola, S., Marijon, P., Markello, C., Marschall, T., Martin, F.J., McCartney, A., McDaniel, J., Miga, K.H., Mitchell, M.W., Mountcastle, J., Munson, K.M., Mwaniki, M.N., Nattestad, M., Nurk, S., Olsen, H.E., Olson, N.D., Pesout, T., Phillippy, A.M., Popejoy, A.B., Porubsky, D., Prins, P., Puiu, D., Rautiainen, M., Regier, A.A., Rhie, A., Sacco, S., Sanders, A.D., Schneider, V.A., Schultz, B.I., Shafin, K., Sibbesen, J.A., Sirén, J., Smith, M.W., Sofia, H.J., Tayoun, A.N.A., Thibaud-Nissen, F., Tomlinson, C., Tricomi, F.F., Villani, F., Vollger, M.R.,

Wagner, J., Walenz, B., Wang, T., Wood, J.M.D., Zimin, A.V., Zook, J.M., Marschall, T., Li, H., Paten, B.: Pangenome graph construction from genome alignments with minigraph-cactus. Nat Biotechnol (2023). https://doi.org/10.1038/s41587-023-01793-w

[9] Noll, N., Molari, M., Shaw, L.P., Neher, R.A.: PanGraph: scalable bacterial pan-genome graph construction (2022). https://doi.org/10.1101/2022.02.24.481757

[10] Garrison, E., Guarracino, A.: Unbiased pangenome graphs. Bioinformatics **39**(1) (2022). https://doi.org/10.1093/bioinformatics/btac743

[11] Minkin, I., Pham, S., Medvedev, P.: TwoPaCo: an efficient algorithm to build the compacted de bruijn graph from many complete genomes. Bioinformatics **33**(24), 4024–4032 (2016). https://doi.org/10.1093/bioinformatics/btw609

[12] Liao, W.-W., Asri, M., Ebler, J., Doerr, D., Haukness, M., Hickey, G., Lu, S., Lucas, J.K., Monlong, J., Abel, H.J., Buonaiuto, S., Chang, X.H., Cheng, H., Chu, J., Colonna, V., Eizenga, J.M., Feng, X., Fischer, C., Fulton, R.S., Garg, S., Groza, C., Guarracino, A., Harvey, W.T., Heumos, S., Howe, K., Jain, M., Lu, T.-Y., Markello, C., Martin, F.J., Mitchell, M.W., Munson, K.M., Mwaniki, M.N., Novak, A.M., Olsen, H.E., Pesout, T., Porubsky, D., Prins, P., Sibbesen, J.A., Sirén, J., Tomlinson, C., Villani, F., Vollger, M.R., Antonacci-Fulton, L.L., Baid, G., Baker, C.A., Belyaeva, A., Billis, K., Carroll, A., Chang, P.-C., Cody, S., Cook, D.E., Cook-Deegan, R.M., Cornejo, O.E., Diekhans, M., Ebert, P., Fairley, S., Fedrigo, O., Felsenfeld, A.L., Formenti, G., Frankish, A., Gao, Y., Garrison, N.A., Giron, C.G., Green, R.E., Haggerty, L., Hoekzema, K., Hourlier, T., Ji, H.P., Kenny, E.E., Koenig, B.A., Kolesnikov, A., Korbel, J.O., Kordosky, J., Koren, S., Lee, H., Lewis, A.P., Magalhães, H., Marco-Sola, S., Marijon, P., McCartney, A., McDaniel, J., Mountcastle, J., Nattestad, M., Nurk, S., Olson, N.D., Popejoy, A.B., Puiu, D., Rautiainen, M., Regier, A.A., Rhie, A., Sacco, S., Sanders, A.D., Schneider, V.A., Schultz, B.I., Shafin, K., Smith, M.W., Sofia, H.J., Tayoun, A.N.A., Thibaud-Nissen, F., Tricomi, F.F., Wagner, J., Walenz, B., Wood, J.M.D., Zimin, A.V., Bourque, G., Chaisson, M.J.P., Flicek, P., Phillippy, A.M., Zook, J.M., Eichler, E.E., Haussler, D., Wang, T., Jarvis, E.D., Miga, K.H., Garrison, E., Marschall, T., Hall, I.M., Li, H., Paten, B.: A draft human pangenome reference. Nature **617**(7960), 312–324 (2023). https://doi.org/10.1038/s41586-023-05896-x

[13] Fischer, C., Garrison, E.: chfi/gfaestus: a pangenome graph browser. Zenodo (2022). https://doi.org/10.5281/ZENODO.6954035. https://zenodo.org/record/6954035

[14] Cagan, A., Theunert, C., Laayouni, H., Santpere, G., Pybus, M., Casals, F., Prüfer, K., Navarro, A., Marques-Bonet, T., Bertranpetit, J., Andrés, A.M.: Natural selection in the great apes. Mol Biol Evol **33**(12), 3268–3283 (2016). https://doi.org/10.1093/molbev/msw215

[15] Logsdon, G.A., Vollger, M.R., Hsieh, P., Mao, Y., Liskovykh, M.A., Koren, S., Nurk, S., Mercuri, L., Dishuck, P.C., Rhie, A., de Lima, L.G., Dvorkina, T., Porubsky, D., Harvey, W.T., Mikheenko, A., Bzikadze, A.V., Kremitzki, M., Graves-Lindsay, T.A., Jain, C., Hoekzema, K., Murali, S.C., Munson, K.M., Baker, C., Sorensen, M., Lewis, A.M., Surti, U., Gerton, J.L., Larionov, V., Ventura, M., Miga, K.H., Phillippy, A.M., Eichler, E.E.: The structure, function and evolution of a complete human chromosome 8. Nature **593**(7857), 101–107 (2021). https://doi.org/10.1038/s41586-021-03420-7

[16] Guarracino, A., Buonaiuto, S., de Lima, L.G., Potapova, T., Rhie, A., Koren, S., Rubinstein, B., Fischer, C., Abel, H.J., Antonacci-Fulton, L.L., Asri, M., Baid, G., Baker, C.A., Belyaeva, A., Billis, K., Bourque, G., Carroll, A., Chaisson, M.J.P., Chang, P.-C., Chang, X.H., Cheng, H., Chu, J., Cody, S., Cook, D.E., Cook-Deegan, R.M., Cornejo, O.E., Diekhans, M., Doerr, D., Ebert, P., Ebler, J., Eichler, E.E., Eizenga, J.M., Fairley, S., Fedrigo, O., Felsenfeld, A.L., Feng, X., Flicek, P., Formenti, G., Frankish, A., Fulton, R.S., Gao, Y., Garg, S., Garrison, N.A., Giron, C.G., Green, R.E., Groza, C., Haggerty, L., Hall, I., Harvey, W.T., Haukness, M., Haussler, D., Heumos, S., Hickey, G., Hoekzema, K., Hourlier, T., Howe, K., Jain, M., Jarvis, E.D., Ji, H.P., Kenny, E.E., Koenig, B.A., Kolesnikov, A., Korbel, J.O., Kordosky, J., Lee, H., Lewis, A.P., Li, H., Liao, W.-W., Lu, S., Lu, T.-Y., Lucas, J.K., Magalhães, H., Marco-Sola, S., Marijon, P., Markello, C., Marschall, T., Martin, F.J., McCartney, A., McDaniel, J., Miga, K.H., Mitchell, M.W., Monlong, J., Mountcastle, J., Munson, K.M., Mwaniki, M.N., Nattestad, M., Novak, A.M., Nurk, S., Olsen, H.E., Olson, N.D., Paten, B., Pesout, T., Popejoy, A.B., Porubsky, D., Prins, P., Puiu, D., Rautiainen, M., Regier, A.A., Sacco, S., Sanders, A.D., Schneider, V.A., Schultz, B.I., Shafin, K., Sibbesen, J.A., Sirén, J., Smith, M.W., Sofia, H.J., Tayoun, A.N.A., Thibaud-Nissen, F., Tomlinson, C., Tricomi, F.F., Villani, F., Vollger, M.R., Wagner, J., Walenz, B., Wang, T., Wood, J.M.D., Zimin, A.V., Zook, J.M., Gerton, J.L., Phillippy, A.M., Colonna, V., Garrison, E.: Recombination between heterologous human acrocentric chromosomes. Nature **617**(7960), 335–343 (2023). https://doi.org/10.1038/s41586-023-05976-y

[17] Crysnanto, D., Leonard, A., Pausch, H.: Comparison of methods for building pangenome graphs. In: Proceeding of 12th World Congress on Genetics Applied to Livestock Production (WCGALP) Technical and Species Orientated Innovations in Animal Breeding, and Contribution of Genetics to Solving Societal Challenges, pp. 1066–1069 (2022). Wageningen Academic

Publishers

[18] Zhang, H., Wu, S., Aluru, S., Li, H.: Fast sequence to graph alignment using the graph wavefront algorithm. arXiv preprint arXiv:2206.13574 (2022)

[19] Leonard, A.S., Crysnanto, D., Mapel, X.M., Bhati, M., Pausch, H.: Graph construction method impacts variation representation and analyses in a bovine super-pangenome (2022). https://doi.org/10.1101/2022.09.17.508368

[20] Zhou, Y., Zhang, Z., Bao, Z., Li, H., Lyu, Y., Zan, Y., Wu, Y., Cheng, L., Fang, Y., Wu, K., Zhang, J., Lyu, H., Lin, T., Gao, Q., Saha, S., Mueller, L., Fei, Z., Städler, T., Xu, S., Zhang, Z., Speed, D., Huang, S.: Graph pangenome captures missing heritability and empowers tomato breeding. Nature **606**(7914), 527–534 (2022). https://doi.org/10.1038/s41586-022-04808-9

[21] Marçais, G., Delcher, A.L., Phillippy, A.M., Coston, R., Salzberg, S.L., Zimin, A.: MUMmer4: A fast and versatile genome alignment system. PLoS Computational Biology **14**(1), 1005944 (2018). https://doi.org/10.1371/journal.pcbi.1005944

[22] 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

[23] Jain, C., Koren, S., Dilthey, A., Phillippy, A.M., Aluru, S.: A fast adaptive algorithm for computing whole-genome homology maps. Bioinformatics **34**(17), 748–756 (2018). https://doi.org/10.1093/bioinformatics/bty597

[24] Marco-Sola, S., Moure, J.C., Moreto, M., Espinosa, A.: Fast gap-affine pairwise alignment using the wavefront algorithm. Bioinformatics (2020). https://doi.org/10.1093/bioinformatics/btaa777

[25] Marco-Sola, S., Eizenga, J.M., Guarracino, A., Paten, B., Garrison, E., Moreto, M.: Optimal gap-affine alignment in o(s) space. Bioinformatics **39**(2) (2023). https://doi.org/10.1093/bioinformatics/btad074

[26] Bollobás, B.: The evolution of random graphs—the giant component. In: Random Graphs, vol. 184, pp. 130–59 (2001). Cambridge university press Cambridge

[27] Lee, C., Grasso, C., Sharlow, M.F.: Multiple sequence alignment using partial order graphs. Bioinformatics **18**(3), 452–464 (2002). https://doi.org/10.1093/bioinformatics/18.3.452

[28] Vaser, R., Sović, I., Nagarajan, N., Šikić, M.: Fast and accurate de novo genome assembly from long uncorrected reads. Genome Res. **27**(5), 737–746 (2017). https://doi.org/10.1101/gr.214270.116

[29] Gao, Y., Liu, Y., Ma, Y., Liu, B., Wang, Y., Xing, Y.: abPOA: an SIMD-based c library for fast partial order alignment using adaptive band. Bioinformatics **37**(15), 2209–2211 (2020). https://doi.org/10.1093/bioinformatics/btaa963

[30] Vollger, M.R., Dishuck, P.C., Harvey, W.T., DeWitt, W.S., Guitart, X., Goldberg, M.E., Rozanski, A.N., Lucas, J., Asri, M., Abel, H.J., Antonacci-Fulton, L.L., Baid, G., Baker, C.A., Belyaeva, A., Billis, K., Bourque, G., Buonaiuto, S., Carroll, A., Chaisson, M.J.P., Chang, P.-C., Chang, X.H., Cheng, H., Chu, J., Cody, S., Colonna, V., Cook, D.E., Cook-Deegan, R.M., Cornejo, O.E., Diekhans, M., Doerr, D., Ebert, P., Ebler, J., Eizenga, J.M., Fairley, S., Fedrigo, O., Felsenfeld, A.L., Feng, X., Fischer, C., Flicek, P., Formenti, G., Frankish, A., Fulton, R.S., Gao, Y., Garg, S., Garrison, E., Garrison, N.A., Giron, C.G., Green, R.E., Groza, C., Guarracino, A., Haggerty, L., Hall, I.M., Haukness, M., Haussler, D., Heumos, S., Hickey, G., Hourlier, T., Howe, K., Jain, M., Jarvis, E.D., Ji, H.P., Kenny, E.E., Koenig, B.A., Kolesnikov, A., Korbel, J.O., Kordosky, J., Koren, S., Lee, H., Li, H., Liao, W.-W., Lu, S., Lu, T.-Y., Lucas, J.K., Magalhães, H., Marco-Sola, S., Marijon, P., Markello, C., Marschall, T., Martin, F.J., McCartney, A., McDaniel, J., Miga, K.H., Mitchell, M.W., Monlong, J., Mountcastle, J., Mwaniki, M.N., Nattestad, M., Novak, A.M., Nurk, S., Olsen, H.E., Olson, N.D., Paten, B., Pesout, T., Phillippy, A.M., Popejoy, A.B., Prins, P., Puiu, D., Rautiainen, M., Regier, A.A., Rhie, A., Sacco, S., Sanders, A.D., Schneider, V.A., Schultz, B.I., Shafin, K., Sibbesen, J.A., Sirén, J., Smith, M.W., Sofia, H.J., Tayoun, A.N.A., Thibaud-Nissen, F., Tomlinson, C., Tricomi, F.F., Villani, F., Vollger, M.R., Wagner, J., Walenz, B., Wang, T., Wood, J.M.D., Zimin, A.V., Zook, J.M., Munson, K.M., Lewis, A.P., Hoekzema, K., Logsdon, G.A., Porubsky, D., Paten, B., Harris, K., Hsieh, P., Eichler, E.E.: Increased mutation and gene conversion within human segmental duplications. Nature **617**(7960), 325–334 (2023). https://doi.org/10.1038/s41586-023-05895-y

[31] Doerr, D., Marijon, P., Marschall, T.: GFAffix identifies walk-preserving shared affixes in variation graphs and collapses them into a non-redundant graph structure (2023). https://github.com/marschall-lab/GFAffix

[32] Ewels, P., Magnusson, M., Lundin, S., Käller, M.: Multiqc: summarize analysis results for multiple tools and samples in a single report. Bioinformatics **32**(19), 3047 (2016). https://doi.org/10.1093/bioinformatics/btw354

[33] Paten, B., Eizenga, J.M., Rosen, Y.M., Novak, A.M., Garrison, E., Hickey, G.: Superbubbles, ultrabubbles, and cacti. Journal of Computational Biology **25**(7), 649–663 (2018). https://doi.org/10.1089/cmb.2017.0251

[34] Garrison, E., Kronenberg, Z.N., Dawson, E.T., Pedersen, B.S., Prins, P.: A spectrum of free software tools for processing the VCF variant call format: vcflib, bio-vcf, cyvcf2, hts-nim and slivar. PLoS Computational Biology **18**(5), 1009123 (2022). https://doi.org/10.1371/journal.pcbi.1009123

[35] Llamas, B., Narzisi, G., Schneider, V., Audano, P.A., Biederstedt, E., Blauvelt, L., Bradbury, P., Chang, X., Chin, C.-S., Fungtammasan, A., Clarke, W.E., Cleary, A., Ebler, J., Eizenga, J., Sibbesen, J.A., Markello, C.J., Garrison, E., Garg, S., Hickey, G., Lazo, G.R., Lin, M.F., Mahmoud, M., Marschall, T., Minkin, I., Monlong, J., Musunuri, R.L., Sagayaradj, S., Novak, A.M., Rautiainen, M., Regier, A., Sedlazeck, F.J., Siren, J., Souilmi, Y., Wagner, J., Wrightsman, T., Yokoyama, T.T., Zeng, Q., Zook, J.M., Paten, B., Busby, B.: A strategy for building and using a human reference pangenome. F1000Res **8**, 1751 (2021). https://doi.org/10.12688/f1000research.19630.2

[36] Traag, V.A., Waltman, L., van Eck, N.J.: From louvain to leiden: guaranteeing well-connected communities. Scientific Reports **9**(1) (2019). https://doi.org/10.1038/s41598-019-41695-z