Multiscale Analysis of Pangenomes Enables Improved Representation of Genomic Diversity For Repetitive And Clinically Relevant Genes

Chen-Shan Chin[1,2,*], Sairam Behera[3], Asif Khalak[2], Fritz J Sedlazeck[3,4], Peter H Sudmant[5], Justin Wagner[6], Justin M. Zook[6]

1. GeneDX Holdings Corp, Stamford, CT 06902, USA
2. Foundation of Biological Data Science, Belmont CA 94002, USA
3. Human Genome Sequencing Center, Baylor College of Medicine, One Baylor Plaza, Houston, TX 77030, USA
4. Department of Computer Science, Rice University, 6100 Main Street, Houston, TX, 77005, USA
5. Department of Integrative Biology, University of California Berkeley, Berkeley, USA
6. Material Measurement Laboratory, National Institute of Standards and Technology, Gaithersburg, MD, 20899, USA

*To whom correspondence should be addressed
[+]Authors are listed in alphabetical order except the corresponding author

Abstract

Advancements in sequencing technologies and assembly methods enable the regular production of high-quality genome assemblies characterizing complex regions. However, challenges remain in efficiently interpreting variation at various scales, from smaller tandem-repeats to megabase re-arrangements, across many human genomes. We present a pangenome research toolkit (PGR-TK) enabling analyses of complex pangenome structural- and haplotype-variation at multiple scales. To demonstrate PGR-TK , we apply the graph decomposition methods to the class II major histocompatibility complex demonstrating the importance of the human pangenome for analyzing complicated regions. Moreover, we investigate the Y-chromosome gene, *DAZ1*/*DAZ2*/*DAZ3*/*DAZ4*, of which structural variants have been linked to male infertility, and X-chromosome genes *OPN1LW* and *OPN1MW* linked to eye disorders. We further showcase PGR-TK across 395 complex repetitive medically important genes. This highlights the power of PGR-TK to resolve complex variation in regions of the genome that were previously too complex to analyze.

Introduction

Studying genomes, the fundamental information contained in all living beings, is the foundation for understanding the biology and evolution of all organisms, as well as the genetic diseases of humans. Despite the millions of human genomes that have been sequenced since the onset of the Human Genome Project[1,2], and the dramatic reduction in the cost of short-read (~150bp) DNA sequencing, there is still fundamental information yet to be revealed in genomics[3]. While it is important to recognize successes to date, including small variant surveys, genome-wide association studies[4–7], and the development of routine lab tests for genetic-based precision medicine [8–11], there remain fundamental biological questions that involve structures at greater length scales that can only be captured using long-range information accessible by long-read technologies and diploid phased assemblies[12–14].

With the possibility of resolving variants at multiple scales, small and large, researchers now can fully characterize previously inaccessible regions by focusing on SNPs and small indels alone[15,16]. Examples of such previously inaccessible regions include centromere, telomeres, and complex repeat regions. Recent results with a pangenome-scale de novo human assemblies and the CHM13 telomere to telomere assembly have already shown the potential for revealing biological insights [3,17–20], which are the foundation for understanding complex genetic diseases.

A concept that becomes powerful in such analyses is that of the pangenome -- that is, a characterization of both the genetic structure and the genetic variation across diverse individuals of a species. However, such complexity and diversity generate interpretive challenges that require more advanced tools. A graph representing many genome assemblies at once provides a way to visualize and analyze complicated structural variations among different haplotypes[21–28]. Previously, distinct approaches to generate graphs representing pangenome structures have been proposed for various applications For example, variant graph[27,29] and PanGenie[26] focus on improving variant calling and genotyping with pangenome references. Cactus graphs[25], Progressive Cactus graphs[30], PGGB[19], and cactus-mingraph[19,24] build pangenome graphs aiming for large-scale structural rearrangement comparisons. Stringomics graph with "stringlet"[21], Seqwish[22] and de Bruijn graph based approaches[26,31] provide algorithms and data structures for improving storage and query efficiency and reduce bias caused by the alignment processes. These tools provide more accessible pictures for researchers to understand repeats and rearrangements than using computational intensive and visually complicated multiple sequence alignments (MSA)[30,32,33]. The traditional MSA view is typically represented as a big matrix where each row represents a different genome and the columns represent the bases. With MSA, the relationships between sequences are not obvious when there are complicated repeats or structural variations. Instead of per base alignment, a pangenome graph effectively condenses the homologous regions and can express the relationship between those different regions through graph edge connections that are easier to trace. Meanwhile, although a graph is an elegant data structure for gathering information from pan-genomic assemblies, there remains a gap in projecting the underlying linear sequences onto a graph at various scales to reveal and compare features of many different haplotypes[23].

To address this gap, we present a generalized graph framework as a software package, PanGenome Research Tool Kit (PGR-TK, https://github.com/GeneDx/pgr-tk), that is scalable to rapidly represent multiple samples at varying resolution levels by adopting different parameters

to facilitate exploratory analysis. PGR-TK is able to resolve and visualize the most complex regions of the human genome that often impact multiple medical important phenotypes (e.g. LPA, HLA, etc).

We demonstrate the ability of PGR-TK to visualize and enable deeper insights into complex variants in repetitive genes, including a gene within nested palindromic and tandem repeats (AMY1A), the MHC region including the complex HLA Class II locus[34], the GIAB challenging medically relevant gene list[35], and chrX and chrY ampliconic genes[36]. Many genome wide studies including GIAB have excluded many of these genes from their analyses because they are challenging to represent in VCF and it is challenging to compare differing representations[37]. To understand how PGR-TK can help with the challenge of variant calling, variant representation and comparison across these genes and genomic loci we utilize the Human Pangenome Reference Consortium[19,38] year one 47 human genome assemblies (94 diverse haplotypes). With the ability to survey a large set of genes swiftly with PGR-TK, we hope to understand how to better provide a broader benchmark set for challenging genes utilizing HPRC assemblies in the future. We examine OPN1LW and OPN1MW on chromosome X and *DAZ1/2/3/4* on chromosome Y in detail to understand how the limit due to complicated large scale genome rearrangement impacts the current methodology of generating variant call benchmarks. Our initial analysis of the GIAB Clinical And Medically Important Genes(CMRG) with a pangenome graph approach will help the research community to adapt the pangenome resource for clinical and medical genetic applications. Tools for visualizing and analyzing complicated re-arrangement loci such as PGR-TK will be essential for better variant calling and understanding the related mechanism for the community.

Results

Pangenome Research Toolkit

The PGR-TK has several different components to facilitate rapid pangenome analysis. The general scope and design of the PGR-TK is illustrated in **Figure 1a**. PGR-TK applies the computation techniques and data structures initially developed for fast genome assemblers[39–41] to pangenomics analysis tasks. Instead of building a whole genome graph at once, which can be computationally expensive. PGR-TK provides tools for building an indexed sequence database, fetching and querying sequences of interest (e.g. genes or regions with large scale structural variations) from the database to create pan-genomics graphs accordingly. It uses minimizer anchors to generate pangenome graphs at different scales without more computational intensive sequence-to-sequence alignment or explicitly calling variants with respect to a reference. The generation step of the pangenome graph considers all input sequences equivalently without a preferential reference. Note that the sequence fetching step using a query sequence may introduce bias due to missing or incorrect alignments. We also developed an algorithm to decompose tangled pangenome graphs to more manageable units (principal bundles). With such decomposition, we can easily project the linear genomics sequence onto the principal bundles. It can provide more straightforward visualization to generate insight by revealing the contrast of the repeat and rearrangement variations among the haplotypes. Such pangenome-level graph decomposition provides utilities similar to the A-de Bruijn graph approach for identifying repeats and conserved segmental duplications[42–45] , but for the whole human pangenome collection at once.

PGR-TK employs the Assembly Genome Compressor [46] for storing pangenome assembly contigs and includes binary for creating the **s**parse **hi**erarchical **m**ini**m**iz**er** (SHIMMER) index. For the HPRC year-one data release (94 fully assembled haplotypes from 47 samples), it takes 18 minutes to create the index file on an AWS c5.12xlarge instance, with the default parameters. This is significantly faster than building an alignment index for query with tools such as mimimap2 (**Supplementary Table 1**). Although PGR-TK was designed to retrieve homologous sequences from the database, rather than finding the best alignments, our evaluations indicate that the query results are generally consistent with other alignment tools (**Supplementary Table 2 & 3**).

Once the index is built, it can be loaded into memory within minutes. As shown in **Figure 1a**, there are three main functional modules utilizing the index: (1) fetching homologous regions and sequences of the pangenome database given a query sequence, (2) creating Minimizer Anchored Pangenome Graph (MAP-graph), and (3) command line tools and a software library for interactive analysis and visualization on the generated graph and the underlying sequences. One of the major applications of PGR-TK is for deconvolving large regions of the human genome to reveal complex variations. It offers a set of efficient command line tools for various tasks, but also allows for more interactive and in-depth analysis through its integration with Jupyter Lab and other data science tools. This makes it a valuable resource for researchers seeking to uncover insights from their genomic data."

The source code and library can be downloaded from https://github.com/GeneDX/pgr-tk. The documentation of the Python APIs is at https://genedx.github.io/pgr-tk/.

Sparse Hierarchical Minimizer Index

Sparse Hierarchical Minimizer (SHIMMER) is a data structure extending the minimizer for more efficient indexing over larger regions. Additional minimizer reduction steps to generate sparse minimizers are applied to the minimizer sequences instead of the original base-pair sequences in a hierarchical way[39]. Such sparse minimizers can serve as natural “anchors” or "markers" on genomics sequences without an explicit reference coordinate system. We utilize the SHIMMERs for quick sequence queries as initially proposed by Roberts, M. et al.[47]. PGR-TK identifies all neighboring pairs of SHIMMERs and indexes all the sequence segments between the pairs. **Figure 1b** shows a cartoon of the SHIMMERs identified on each sequence. Then, the pairs of the neighboring SHIMMERs are used for indexing the corresponding sequence segments within the paired SHIMMERs. After that, we build a look-up table of all pairs of SHIMMERs to all segments with the same pair at both ends (**Figure 1c**). For the query, we compute the neighboring SHIMMER pairs from a query sequence and search the database for all segments indexed by the same pairs. Finally, we can fetch all target segments stored in the database to get all related sequence information for further analysis. PGR-TK provides functions to refine the raw query results and filter out spurious alignments likely caused by repeats outside the region of interest. With the set of sequences homologous to the query sequence, we can quickly perform downstream analysis work, e.g., variant discovery by aligning the sequences to each other. Furthermore, we can generate a local pan-genomics (MAP-graph) for comparing the sequences in the pangenome dataset at various scales by adjusting parameters to fit different analysis tasks.

Minimizer Anchored Pangenome Graph (MAP-graph)

PGR-TK provides tools to generate the "Minimizer Anchored Pangenome(MAP) graph" from a set of homologous sequences. The vertices in a MAP-graph are labeled with the neighboring SHIMMER pairs representing a set of sequence segments in the database (**Figure 1c, 1d, and Methods**). The edges in the MAP-graph are induced when at least one sequence connects the two fragments. Thus, each sequence naturally corresponds to a path in the graph, and the vertices in the path also contain the segments of other sequences in the database that share the same SHIMMER pair label. Please see the **Methods** for a precise mathematical definition of a MAP-graph. The deployment of minimizer- or minhash- based approaches successfully in sequence comparison[39,40,48] indicates that sequence segments with the same minimizer labels are also likely to be highly homologous. The homology between sequences can be further confirmed by explicit sequence alignment of the segment inside a MAP-graph vertex. However, the computation intensive base-to-base alignment is not required for building the MAP-graph.

The MAP-graph construction in pgr-tk is highly efficient, as it does not rely on traditional sequence-to-sequence alignment. This is demonstrated by the fast graph construction for a set of 147 major histocompatibility complex class II region sequences from the pangenome reference, which was completed in under 5 seconds of wall-clock time using PGR-TK, compared to the 3.5 minutes using seqwish[22] and 13 minutes using minigraph[24] (**Supplementary Table 4**).

The size of vertices in the MAP-graph, which represents sequence segments in the pangenome, can be adjusted by adjusting the parameters that determine the distance between minimizers. This allows us to study genomic features at different length scales and generate pangenome graphs with varying levels of detail. This is particularly useful when analyzing features that vary in size, such as tandem repeats in the human genome, which can range from a few hundred base pairs to 1-2 kilobases (**Supplementary Table 5**). By generating pangenome graphs at different levels of detail, we can gain a more comprehensive understanding of complex variation patterns within populations and focus on specific features of interest.

The analysis of pan-genomic structure can be adjusted by controlling the parameters of minimizer window size (w), minimizer size (k), and hierarchical reduction factor (r), along with an auxiliary parameter min_span, which sets the minimum distance between minimizers in the construction of the SHIMMER index and MAP-graph (see **Methods**). The length of each vertex in the MAP-graph, representing a sequence segment, can be modified by adjusting these parameters. This allows us to study genomics features at different length scales.

Supplementary Figure 1 illustrates the vertex length distribution for different parameter sets using chromosome 1 of CHM13 assembly. An increase in either w or r results in longer sequences being represented by each vertex, enabling a more sparse sampling of the pangenome. The choice of parameters depends on the length of the region of interest and the size of relevant biological features, such as repeat sizes and distances. For example, when studying large-scale differences, bigger w and r values are preferable to generate a sparse index that can efficiently capture large-scale differences. Conversely, to compare small-scale differences, smaller w or r values should be used. Determining the optimal parameters for the pangenome graph generation step can be challenging if the underlying interesting features are

less understood. In light of this, we have found that the best initial parameter choice is determined by the length of the sequences of interest to ensure comprehensible results. Based on this, we provide a simple formula for selecting the parameters (**Supplementary Table 6**, and see **Figures 2** and **Supplementary Figure 3** for related examples).

Deriving the Principal Bundle Decomposition from the MAP-graph for Visualizing and Analyzing Large-scale Pangenome Variation Structures

A pangenome graph can serve as a cornerstone for analyzing repeat structure variation in population[17,19]. It is usually hard to compare multiple sequences with complicated repeat structure by examining pairwise sequence alignments directly. The traditional visualization technique "dot-plot[49]" allows us to perceive the complexity of the repeats but it does not provide insights into the repeat structures as linear representations across each individual sequence directly. Furthermore, only two sequences can be compared with a dot-plot.

As an example, we use PGR-TK to investigate the repeat structure of the *AMY1A* gene (Alpha-amylase 1, an enzyme for the first step of catalyzing starch and glycogen in saliva) locus. We pick *AMY1A* as it has various numbers of copies caused by larger scale structure variation related to the repeat surrounding the gene. The dot-plots from randomly picking 36 sequences of a 400 kb region around *AMY1A* in the first year HPRC assemblies to the GRCh38 *AMY1A* reference sequence are shown in **Supplementary Figure 2a**. Visual inspections of the dot plots show there are various numbers of copies of forward repeats and inverted repeats forming palindrome sequences at the scales of 100kb, and from zero up to 5 palindrome units. Still, only pairwise comparisons are enabled with dotplots, and thus, we lack a comprehensive assessment.

For comparison, we generate the *AMY1A* MAP-graphs at two different scales (**Figure 2)** from the HPRC year one assemblies (47 samples). These can be generated with PRG-TK in less than three minutes from indexed sequence data. In additional to the MAP-graph, we provide tools analyzing a MAP-graph to "re-linearize" the graph into a set of "principal bundles." We design the algorithm to generate the principal bundles representing those consensus paths which are most likely corresponding to repeat units in the pangenomes. The algorithm searches the paths that the majority of the pangenome sequences go through without branching as the principal bundles. This is analogous to identifying the contigs[50,51] in genome assembly algorithms.

Figure 2 shows the principal bundle decomposition of the *AMY1A* region MAP-graph in two different scales for comparison. The smaller choice of "r" generates a MAP-graph with more vertices in the graph and each vertex only represents a smaller portion of the pangenome. This leads to a finer scale of principal bundle decompositions.

The linear representation derived from the MAP-graph allows for efficient identification and classification of repeat structures, which are otherwise challenging to characterize. Seven genomes were selected for analysis in **Figure 2**, each demonstrating distinct repeat structures. For example, HG00438#2 mostly lacks repetitive sequences. The GRCh38 one has one relatively simple invert repeat (forming a palindrome region). HG02145#1 has 3 copies of non-inverted repeats (labeled as repeat 1,2, and 3). HG02257#2 has 3 palindromic repeats (labeled as P1, P2, and P3) . The two haplotypes from HG002 have similar structure to the GRCh38 except there is

an inversion in the middle of the palindrome repeats in one of the two haplotypes. A full plot with all 96 repeat structures, including a hierarchical clustering tree identifying the similarities, is shown in the **Supplementary Figure 2b**. This decomposition approach can be utilized by researchers to effectively classify the repeat structures of regions of interest.

Visualizing and Analyzing the Highly Polymorphism HLA Class 2 Locus

The major histocompatibility complex (MHC) region in human genomes is highly polymorphic. The genomic sequences of the MHC are fundamental for understanding a human's adaptive immune system and autoimmune diseases[52,53]. Due to its complexity and polymorphic nature, it has been challenging to get a complete picture of the MHC genomics in the human population and to benchmark variant calling in the most variable regions[34,54]. The HPRC assemblies provide a new opportunity to analyze the MHC sequences with nearly fully assembled sequences of the region.

To showcase the effectiveness of PGR-TK in analyzing complicated human haplotype structures and sequences, we applied it on the HLA class II locus (GRCh38, chr6:32,313,513-32,992,088). We fetched the HPRC HLA class II haplotype sequences by anchoring them with more conserved flanking regions. Our dataset consists of a total of 105 full-length sequences, ranging from 650kbp to 800kbp. **Figure 3a** illustrates the MAP-graph of the 105 sequences. The tangled region in the MAP-graph represents highly polymorphic haplotypes in the human population. By generating the MAP-graph and the principal bundle decomposition of the MHC Class II, we can uncover the combinatorial nature of haplotype variation in relation to the newly released human pangenome references.

We constructed a hierarchical dendrogram on top of the principal bundle decomposition to study the relation of highly polymorphic haplotypes in the MHC region of the human population. The PGR-TK provides a command line tool to compute a distance metric derived from pair-wise sparse alignment of the bundles between two sequences, and generate a dendrogram from all pairwise distances. Known HLA Class II gene sequences were also mapped to 105 pangenome sequences. The full annotated principal bundle decomposition, which was annotated with HLA Class II genes and the hierarchical clustering dendrogram, is shown in **Figure 3b**. The highly polymorphic region in the MAP-graph was found to correspond to the *DRB-1/3/4/5*, *DOB*, and *DOA1/2* region. By clustering the full set of haplotype sequences, we can identify the combination of bundles in the entire region that correspond to the gene combinations of each cluster. Our approach employs the principal bundles to classify the complete sequence, rather than relying solely on gene fragments. This classification has the potential to facilitate new applications for improved genotyping or haplotyping of larger population data in this complex region. The results could offer valuable insights into the relationship between haplotype sequences and gene combinations in the context of genetic variation and disease susceptibility.

We can also use the vertices in the MAP-graph to conduct a principal component analysis (PCA) of the MHC class II regions. We collected all vertices in the MAP-graph to form the basis of vectors. Then, we constructed a binary vector for each haplotype path by indicating whether or not the path of the haplotype passes through the vertex. **Figure 3c** displays the principal component plot of the haplotype path vectors, along with the ethnic groups. The dotted line

connects the two haplotypes of an individual in the sample. We have highlighted four different groups based on the HLA class II gene combinations (excluding the subtype). Each group contains 10-49 haplotypes. With the current dataset of a limited number of haplotypes, we have not found any statistically significant patterns yet. However, as additional data will be released from HPRC in the coming years, we anticipate that the MAP-graph can be utilized to systematically analyze this region and better understand its impact on human disease within ethnic group structure.

Visualizing and Analyzing the Large Scales Nested Inversion of Medically Relevant Amplicon Genes

GIAB is using the fully assembled HG002 chrX and chrY from T2T to form new small variant and structural variant benchmarks. The assembly fully resolves the medically relevant ampliconic genes[36,55,56] *OPN1LW*/*OPN1MW*/*OPN1MW2*/*OPN1MW3* and *DAZ1*/*DAZ2*/*DAZ3*/*DAZ4*, but the variation in these genes is too complex for current approaches to make reliable variant calls compatible with current benchmarking tools.

For example, the genes *OPN1MW* and *OPN1MW2* are inside a 74 kb deletion in HG002 relative to GRCh38, so HG002 contains only 2 of the 4 copies of the array in GRCh38 - *OPN1LW* and one copy of *OPN1MW*/*OPN1MW2*/*OPN1MW3*. Dipcall[57] can call the 74 kb deletion and variants in the other gene copies, but it may be possible to align the assembly to GRCh38 in alternative ways. The visualization from PGR-TK makes clear the varying number of genes in this array in each haplotype in **Figure 4a**, which is important for some phenotypes like color blindness, since seeing full color requires OPN1LW and at least one copy of *OPN1MW*/*OPN1MW2*/*OPN1MW3* [55,58].

Another important gene family *DAZ1*/*DAZ2*/*DAZ3*/*DAZ4* are in a set of nested palindromic repeats. It has been reported that partial deletions in this region may cause male infertility[56]. It would be useful to understand the natural distribution of non-pathogenic structural variants across this ampliconic gene cluster. *DAZ1* and *DAZ2* are ~1.5 Mbp from *DAZ3* and *DAZ4*, and HG002 has a 1 to 2 Mbp inversion relative to GRCh38 with breakpoints in the segmental duplications that contain the *DAZ* genes (**Figure 4b**). In addition to the large inversion, the *DAZ* genes contain structural variants, including a ~10 kb deletion in *DAZ2*, 2 deletions in *DAZ4*, and 2 insertions in *DAZ3* of sequences that are only in *DAZ1* and *DAZ4* in GRCh38. PGR-TK's ability to color and visualize variation with the principal bundle decomposition algorithm at multiple scales enables intuitive understanding of this type of very complex variation, which would be very difficult to represent and understand as simple structural variant calls in VCF format.

The efficiency of PGR-TK makes it suitable for analyzing complex variants at isolated loci, as well as a set of regions of interest. For example, GIAB has identified a set of 395 challenging but medically relevant genes. Analyzing this set in the pangenome references will provide insight into the related complicated variation at the population level.

Using PGR-TK, we extracted all sequences of the 395 genes from the HPRC year one release (94 haplotype assemblies), CHM13 v1.1, GRCh38, and hg19 of all 385 CMRG. We generated a MAP-graph for each gene and output it in GFA format (as seen in the **Supplementary Material**). For each graph, we derived two metrics to estimate (1) the degree of polymorphism among the

pangenomes, and (2) the repeat content taking into account the variations of the pangenomes (as shown in Figure 4c). These two measurements provide independent assessments of the MAP-graph structures of these genes. We found, as expected, that highly repetitive genes (such as *LPA* and *KATNAL2*) are more difficult to create a reliable variant benchmark call set. Many highly repetitive genes are excluded from the current CMRG benchmark set. We did not observe a correlation between higher entropy and the reduction of the gene in the benchmark set. We found that the high entropy genes also span larger regions in the genome. While entropy can indicate the complexity of variations in the population, we observed different clustering structures of the top entropy genes. (Please see the comparison of the MAP-graph PCA plots of *SNTG2* and *KMT2C* in **Supplementary Figure 4**.)

Discussion

With the advance in DNA sequencing technologies, more comprehensive human genomes at, or close to, telomere to telomere will be collected and made available in the coming years. It will enable researchers to study and characterize those previously inaccessible complex, but likely relevant, regions. The current Human Pangenome Reference Consortium assembly release has significantly impacted our understanding of the human genome architecture. It will also be essential for building applications for clinical and medical tests and diagnostics soon. Flexible and scalable computational tools for analyzing pangenome level genome assemblies will be part of the vital task of improving the practice of precision medicine with rich genomic data such as those from HPRC.

Many of the recently developed pangenome analysis tools allow graph analysis at the whole genome level[22–24]. Meanwhile, the richness of diversity of human genomes over the repetitive regions poses unique challenges for analysis. In our work developing PGR-TK, we focus on providing a flexible library of useful algorithms. Furthermore, it enables analyzing the genome assemblies such that a developer or a researcher can rapidly access certain complex regions by adjusting parameters for visualization and integrating with subsequent analysis.

Each main building unit (the vertex) of the MAP-graph represents a set of closely related sequence fragments. This is more analogous to the stringomics method proposed by Ferragina[21] than other methods building graphs on top of MSA or variant calls. Such approaches combined with the sparse minimizers is efficient to reduce the computation complexity (fewer vertices) to represent larger-scale scale structures. Complementary to that, PGR-TK provides an interface to fetch the sequences within each vertex such that it is possible to combine a MAP-graph with other graph analysis approaches, e.g. Cactus graph[25] and A-de Bruijn Graph[42] for base level analysis, e.g. variant calling, genotyping, and point mutation analysis, with recursive hybrid graph data structures.

We demonstrate how to use the PGR-TK for studying and characterizing the repetitive region *AMY1A* and the highly polymorphic HLA Class II region. We present a tool in PGR-TK backed by a novel pangenome graph traversal algorithm, re-linearizing tangled graphs caused by repetitive sequences to principal bundles for visualization. With the principal bundle decomposition, we can automatically visualize the repetitive and non-repetitive components of haplotype assembly contig. The PGR-TK can provide intuitive qualitative information about different genome arrangement architectures with decomposition and associated visualizations. For example, it

enables visualization of both the very large inversion in the *DAZ* locus and much smaller complex structural variation within the genes. The *OPN1LW*/*OPN1MW* gene array enables visualization of copy number of the subtly different *OPN1LW* and *OPN1MW* genes, which can affect vision, as well as nearby structural variants. We also utilize PGR-TK to survey a set of regions of interest across the whole genome. We derive two metrics for measuring the polymorphism and repetitiveness in human pangenome to more systematically survey complexity of a large set of medically and clinically relevant genes. In the future, we aim to extend the PGR-TK library to provide more quantitative and base-level analysis for both fundamental and translational research utilizing pangenome resources.

Methods

Sequence Database and SHIMMER Index construction

To generate the SHIMMER index, each sequence is scanned and the symmetrical minimizers were generated with the specific minimizer window size "w" and kmer size "k". We call this first level of minimizer. Given a reduction factor "r" > 1, additional levels of minimizer sets[39] are generated to increase the span between the minimizers by a reduction step to facilitate pan-genomics analysis. Even with the reduction step, in some simple sequence context, e.g. long or short tandem repeats, two minimizers can remain too close to each other. A parameter "min_span" can be applied to eliminate a pair of minimizers that are too close. We use a heuristic algorithm to eliminate those minimizers that are within the distance of "min_span" to each other. This helps to reduce the minimizer density when detailed analysis for those simple context regions is not desired. Setting "min_span" to zero and "r" to one will generate the standard minimizers for each sequence.

Each pair of the reduced minimizers (SHIMMERs) are used as the key to build a hashmap to the sequence id, and coordinates and matching orientation of the minimizer pairs on the sequences.

We also present examples of API calling and command line usage in the Supplementary Material.

Generate Minimizer Anchored Pan-genomics Graph (MAP-graph)

The MAP-graph is constructed by scanning through each sequence in the database. The vertices are simply the set of the tuples of neighboring minimizers (minimizer anchored segments). The edges are constructed by connecting minimizer anchored segments as a bi-directed graph. One can consider this as an extension of the string graph[51], where the overlaps are the minimizers at both ends. However, in the pangenome graph, each vertex includes a set of sequence segments from multiple genomes rather than one sequence.

As the MAP-graph can be constructed by scanning the SHIMMER pairs through the sequences. For a given set of n sequences $S = \{s_i \mid i = 0..n-1\}$, the vertices of the MAP-graph are

$V = \{ (m^{(i)}_p, m^{(i)}_{p+1}) \mid m^{(i)}_p$, and $m^{(i)}_{p+1}$ are the p-th and (p+1)-th minimizers of a sequence s_i, in all s_i in S}.

We can assign a weight w_i of a vertex $v_i=(m_a, m_b)$ as the total number of observed (m_a, m_b)-pairs in S.

The edges of the MAP-graph are

$E = \{ (v_i, v_j) \mid v_i = (m^{(i)}_p, m^{(i)}_{p+1})$ and $v_i = (m^{(i)}_{p+1}, m^{(i)}_{p+2})$ for all $(m^{(i)}_p, m^{(i)}_{p+1}, m^{(i)}_{p+2})$ in all s_i in S}

Identify The Principal Bundles in a MAP-graph

To decompose a MAP-graph into principal bundles for downstream analysis, we apply a variation of depth first search[59] (DFS) to build the traversal trees from the graph. Our DFS prioritized vertices with high "weight" (defined as the number of sequence segments contained in a vertex) and taking account the bi-directed nature of the MAP-graph.

The DFS traversal through the graph is then converted to a tree structure internally. The leaf nodes in the tree are typically when the depth first searches are terminated by no out edge from a node or a bubble or a loop is found. As we prioritize the weights of the vertices during DFS, long paths are usually corresponding to the "common" paths that most sequences in the data would go through. Rarer haplotypes typically correspond to short bubble paths in the MAP-graph. Thus, they can be identified as short branches in the DFS traversal tree. We use the tree to remove those vertices in the MAP-graph if those are shorter than pre-specified length in the DFS tree. In general, the vertices in the principal bundle represent more sequences in the set of input sequences (**Supplementary Figure 6**) and are likely more conservative in the pangenome.

After removing the vertices corresponding to the short branches, we further remove vertices in the MAP-graph that have more than three out edges after converting the MAP-graph as an undirected graph. After such removal, the graph will only consist of simple paths and we output those paths as the principal bundles.

In summary, here is the sketch of the algorithm:

(1) Build a DFS traversal tree with a deep first search for a given MAP-graph. To capture paths that are more conserved among the pangenome sequences of interest as the principal bundles, our DFS search prioritizes high-weight vertices when constructing the DFS traversal tree.
(2) In MAP-graph, remove vertices which corresponds to nodes in short branches of the DFS traversal tree
(3) Remove branching vertices in MAP-graph (by considering it as an undirected graph)
(4) Output the simple paths from the resulting graph as the principal bundles.

Principal component plot for the HLA Class II locus

To generate the principal component of the pangenome HLA Class II sequences, we convert each of the haplotype sequences to a binary vector. The binary vector has the same length of the total number of vertices of all principal bundles. Let's call these vertices $V = \{v_i \mid v_i \text{ in principle bundles}, i = 0..n-1\}$, where n is the total number of vertices in the principal bundles. For each sequence s, we construct a binary vector $w_s = \{b_0, b_1, ..., b_{n-1}\}$ where $b_i = 1$ if the sequence s contains the vertex v_i, and $b_i = 0$ if not. Then, we perform the standard principal component transformation with the binary vectors of all sequences from the HLA Class II region.

General Workflow for Analyzing a Region of Interest

Here we outline the general workflow on how to use PGR-TK to generate MAP-graph and the principal bundle decomposition

(1) For a region or sequences of interest, put the sequences as a fasta file for quering the PGR-TK pangnome sequence database. (PGR-TK provides a command line tool "`pgr-fetch`" and python APIs to fetch such sequence from the PGR-TK sequence database.)

(2) Query the whole pangenome database to get initial hits which match the query sequence with the command line tool "`pgr-query`" or using the Python APIs.

(3) Filter the hits to remove unwanted matches that do match a user's analysis objectives. With the command line tool "`pgr-query`", it generats a summary table of the hits for filtering.

(4) With the filtered sequences, using "`pgr-pbundle-decomp`" command line tool to generate the MAP-graph in GFA format and principal bundle decompostion in the BED format. Beside analyzing the generated data, the generated bed file of the decomposition can be rendered by the pgr-pbundle-bed2svg to generate visualization. Python APIs are also provided for more scripting to resolve complicated analysis cases.

(5) Optionally, we can take the fetched sequences from "`pgr-query`" to use with other third party tools, for example, calling variant with dipcall, create multiple sequence alignment, or building other local pangenomic graphs with minigraph, or pggb.

(6) Re-adjust the parameters (w, k, r, min_span) and repeat (3), (4) and additional analysis on the results if necessary

Acknowledgments

Certain commercial equipment, instruments, or materials are identified to specify adequately experimental conditions or reported results. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the equipment, instruments, or materials identified are necessarily the best available for the purpose. FJS and SB are supported by NIH grants (UM1HG008898 and 1U01HG011758-01)

Author contributions

Conceptualization and design: CSC, AK, FS, PS, JW, JZ
Algorithm, code and documentation development of PGR-TK: CSC
Manuscript writing: CSC, AK, FS, JZ
Manuscript revision, data management and code validation: SB, FS, PS, JW

Competing Interests

CSC is an employee and shareholder of GeneDX, Inc. FJS obtains research support from Illumina, PacBio and Oxford Nanopore.

References

1. Lander, E. S. *et al.* Initial sequencing and analysis of the human genome. *Nature* **409**, 860–921 (2001).
2. Venter, J. C. *et al.* The sequence of the human genome. *Science* **291**, 1304–1351 (2001).
3. Nurk, S. *et al.* The complete sequence of a human genome. *Science* **376**, 44–53 (2022).
4. Siva, N. 1000 Genomes project. *Nat. Biotechnol.* **26**, 256 (2008).
5. Canela-Xandri, O., Rawlik, K. & Tenesa, A. An atlas of genetic associations in UK Biobank. *Nat. Genet.* **50**, 1593–1599 (2018).
6. Bycroft, C. *et al.* The UK Biobank resource with deep phenotyping and genomic data. *Nature* **562**, 203–209 (2018).
7. The 'All of Us' Research Program. *N. Engl. J. Med.* **381**, 668–676 (2019).
8. Voelkerding, K. V., Dames, S. A. & Durtschi, J. D. Next-generation sequencing: from basic research to diagnostics. *Clin. Chem.* **55**, 641–658 (2009).
9. Rehder, C. *et al.* Next-generation sequencing for constitutional variants in the clinical laboratory, 2021 revision: a technical standard of the American College of Medical Genetics and Genomics (ACMG). *Genet. Med.* **23**, 1399–1415 (2021).
10. Yohe, S. & Thyagarajan, B. Review of Clinical Next-Generation Sequencing. *Arch. Pathol. Lab. Med.* **141**, 1544–1557 (2017).
11. Green, E. D., Rubin, E. M. & Olson, M. V. The future of DNA sequencing. *Nature* **550**, 179–181 (2017).
12. Mahmoud, M. *et al.* Structural variant calling: the long and the short of it. *Genome Biology* vol. 20 Preprint at https://doi.org/10.1186/s13059-019-1828-7 (2019).
13. Sedlazeck, F. J., Lee, H., Darby, C. A. & Schatz, M. C. Piercing the dark matter: bioinformatics of long-range sequencing and mapping. *Nat. Rev. Genet.* **19**, 329–346 (2018).
14. De Coster, W., Weissensteiner, M. H. & Sedlazeck, F. J. Towards population-scale long-read sequencing. *Nat. Rev. Genet.* **22**, 572–587 (2021).

15. Olson, N. D. *et al.* PrecisionFDA Truth Challenge V2: Calling variants from short and long reads in difficult-to-map regions. *Cell Genomics* vol. 2 100129 Preprint at https://doi.org/10.1016/j.xgen.2022.100129 (2022).
16. Aganezov, S. *et al.* A complete reference genome improves analysis of human genetic variation. *Science* **376**, eabl3533 (2022).
17. Wang, T. *et al.* The Human Pangenome Project: a global resource to map genomic diversity. *Nature* **604**, 437–446 (2022).
18. Sirén, J. *et al.* Genotyping common, large structural variations in 5,202 genomes using pangenomes, the Giraffe mapper, and the vg toolkit. Preprint at https://doi.org/10.1101/2020.12.04.412486.
19. Liao, W.-W. *et al.* A Draft Human Pangenome Reference. *bioRxiv* 2022.07.09.499321 (2022) doi:10.1101/2022.07.09.499321.
20. Jarvis, E. D. *et al.* Automated assembly of high-quality diploid human reference genomes. *bioRxiv* 2022.03.06.483034 (2022) doi:10.1101/2022.03.06.483034.
21. Ferragina, P. & Mishra, B. Algorithms in Stringomics (I): Pattern-Matching against 'Stringomes'. *bioRxiv* 001669 (2014) doi:10.1101/001669.
22. Garrison, E. & Guarracino, A. Unbiased pangenome graphs. *bioRxiv* 2022.02.14.480413 (2022) doi:10.1101/2022.02.14.480413.
23. Eizenga, J. M. *et al.* Pangenome Graphs. *Annu. Rev. Genomics Hum. Genet.* **21**, 139–162 (2020).
24. Li, H., Feng, X. & Chu, C. The design and construction of reference pangenome graphs with minigraph. *Genome Biol.* **21**, 265 (2020).
25. Paten, B. *et al.* Cactus graphs for genome comparisons. *J. Comput. Biol.* **18**, 469–481 (2011).
26. Ebler, J. *et al.* Pangenome-based genome inference allows efficient and accurate genotyping across a wide spectrum of variant classes. *Nat. Genet.* **54**, 518–525 (2022).
27. Garrison, E. *et al.* Variation graph toolkit improves read mapping by representing genetic variation in the reference. *Nat. Biotechnol.* **36**, 875–879 (2018).
28. Computational Pan-Genomics Consortium. Computational pan-genomics: status, promises and

challenges. *Brief. Bioinform.* **19**, 118–135 (2018).

29. Hickey, G. *et al.* Genotyping structural variants in pangenome graphs using the vg toolkit. *Genome Biol.* **21**, 35 (2020).
30. Armstrong, J. *et al.* Progressive Cactus is a multiple-genome aligner for the thousand-genome era. *Nature* **587**, 246–251 (2020).
31. Beller, T. & Ohlebusch, E. A representation of a compressed de Bruijn graph for pan-genome analysis that enables search. *Algorithms for Molecular Biology* vol. 11 Preprint at https://doi.org/10.1186/s13015-016-0083-7 (2016).
32. Darling, A. C. E., Mau, B., Blattner, F. R. & Perna, N. T. Mauve: multiple alignment of conserved genomic sequence with rearrangements. *Genome Res.* **14**, 1394–1403 (2004).
33. Guarracino, A., Heumos, S., Nahnsen, S., Prins, P. & Garrison, E. ODGI: understanding pangenome graphs. *Bioinformatics* (2022) doi:10.1093/bioinformatics/btac308.
34. Chin, C.-S. *et al.* A diploid assembly-based benchmark for variants in the major histocompatibility complex. *Nat. Commun.* **11**, 4794 (2020).
35. Wagner, J. *et al.* Curated variation benchmarks for challenging medically relevant autosomal genes. *Nat. Biotechnol.* **40**, 672–680 (2022).
36. Bhowmick, B. K., Satta, Y. & Takahata, N. The origin and evolution of human ampliconic gene families and ampliconic structure. *Genome Res.* **17**, 441–450 (2007).
37. Zook, J. M. *et al.* A robust benchmark for detection of germline large deletions and insertions. *Nat. Biotechnol.* **38**, 1347–1355 (2020).
38. Cody, S. Year 1 Sequencing data release. *Human Pangenome Reference Consortium* https://humanpangenome.org/year-1-sequencing-data-release/ (2021).
39. Chin, C.-S. & Khalak, A. Human Genome Assembly in 100 Minutes. *bioRxiv* 705616 (2019) doi:10.1101/705616.
40. Ekim, B., Berger, B. & Chikhi, R. Minimizer-space de Bruijn graphs: Whole-genome assembly of long reads in minutes on a personal computer. *Cell Syst* **12**, 958–968.e6 (2021).

41. Li, H. Minimap and miniasm: fast mapping and de novo assembly for noisy long sequences. *Bioinformatics* **32**, 2103–2110 (2016).

42. Pevzner, P. A., Tang, H. & Tesler, G. De novo repeat classification and fragment assembly. *Genome Res.* **14**, 1786–1796 (2004).

43. Jiang, Z. *et al.* Ancestral reconstruction of segmental duplications reveals punctuated cores of human genome evolution. *Nat. Genet.* **39**, 1361–1368 (2007).

44. Wang, M., Ye, Y. & Tang, H. A *de Bruijn* Graph Approach to the Quantification of Closely-Related Genomes in a Microbial Community. *Journal of Computational Biology* vol. 19 814–825 Preprint at https://doi.org/10.1089/cmb.2012.0058 (2012).

45. Pu, L., Lin, Y. & Pevzner, P. A. Detection and analysis of ancient segmental duplications in mammalian genomes. *Genome Res.* **28**, 901–909 (2018).

46. Deorowicz, S., Danek, A. & Li, H. AGC: Compact representation of assembled genomes. *bioRxiv* 2022.04.07.487441 (2022) doi:10.1101/2022.04.07.487441.

47. Roberts, M., Hayes, W., Hunt, B. R., Mount, S. M. & Yorke, J. A. Reducing storage requirements for biological sequence comparison. *Bioinformatics* **20**, 3363–3369 (2004).

48. Ondov, B. D. *et al.* Mash: fast genome and metagenome distance estimation using MinHash. *Genome Biol.* **17**, 132 (2016).

49. Gibbs, A. J. & McIntyre, G. A. The diagram, a method for comparing sequences. Its use with amino acid and nucleotide sequences. *Eur. J. Biochem.* **16**, 1–11 (1970).

50. Kececioglu, J. D. & Myers, E. W. Combinatorial algorithms for DNA sequence assembly. *Algorithmica* **13**, 7 (1995).

51. Myers, E. W. The fragment assembly string graph. *Bioinformatics* vol. 21 ii79–ii85 Preprint at https://doi.org/10.1093/bioinformatics/bti1114 (2005).

52. Trowsdale, J. & Knight, J. C. Major Histocompatibility Complex Genomics and Human Disease. *Annual Review of Genomics and Human Genetics* vol. 14 301–323 Preprint at https://doi.org/10.1146/annurev-genom-091212-153455 (2013).

53. Vandiedonck, C. & Knight, J. C. The human Major Histocompatibility Complex as a paradigm in genomics research. *Briefings in Functional Genomics and Proteomics* vol. 8 379–394 Preprint at https://doi.org/10.1093/bfgp/elp010 (2009).
54. Horton, R. *et al.* Variation analysis and gene annotation of eight MHC haplotypes: the MHC Haplotype Project. *Immunogenetics* **60**, 1–18 (2008).
55. Gardner, J. C., Michaelides, M. & Hardcastle, A. J. Cone opsins, colour blindness and cone dystrophy: Genotype-phenotype correlations. *S. Afr. Med. J.* **106**, S75–8 (2016).
56. Moro, E. *et al.* Male Infertility Caused by a de Novo Partial Deletion of the DAZ Cluster on the Y Chromosome1. *The Journal of Clinical Endocrinology & Metabolism* vol. 85 4069–4073 Preprint at https://doi.org/10.1210/jcem.85.11.6929 (2000).
57. lh3/dipcall. *GitHub* https://github.com/lh3/dipcall.
58. Gardner, J. C. *et al.* X-linked cone dystrophy caused by mutation of the red and green cone opsins. *Am. J. Hum. Genet.* **87**, 26–39 (2010).
59. Skiena, S. S. *The Algorithm Design Manual*. (Springer London).
60. Zook, J. M. & Salit, M. Advancing Benchmarks for Genome Sequencing. *Cell Syst* **1**, 176–177 (2015).
61. Wagner, J. *et al.* Benchmarking challenging small variants with linked and long reads. Preprint at https://doi.org/10.1101/2020.07.24.212712.
62. Ji, Y., Gong, J., Sedlazeck, F. J. & Fan, S. Characterizing the genetic polymorphisms in 370 challenging medically relevant genes using long-read sequencing data from 41 human individuals among 19 global populations. *bioRxiv* 2022.08.03.502734 (2022) doi:10.1101/2022.08.03.502734.
63. Ebert, P. *et al.* Haplotype-resolved diverse human genomes and integrated analysis of structural variation. *Science* **372**, (2021).
64. Burgess, S. *et al.* Association of LPA Variants With Risk of Coronary Disease and the Implications for Lipoprotein(a)-Lowering Therapies: A Mendelian Randomization Analysis. *JAMA Cardiol* **3**, 619–627 (2018).
65. Sanders, S. J. *et al.* De novo mutations revealed by whole-exome sequencing are strongly associated

with autism. *Nature* **485**, 237–241 (2012).

66. O'Roak, B. J. *et al.* Sporadic autism exomes reveal a highly interconnected protein network of de novo mutations. *Nature* **485**, 246–250 (2012).

67. Spielman, D. A. Spectral Graph Theory and its Applications. *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)* Preprint at https://doi.org/10.1109/focs.2007.56 (2007).

Figure 1 (a) Overall architecture and design scope of the PGR-TK library (b) Each sequence in the database is scanned, and the location of the minimizers are recorded to construct the SHIMMER database and minimizer anchored pangenome graph. (c) Each vertex in the MAP-graph represents a collection of sequence fragments sharing the two ending SHIMMERs in the database. (d) MAP-graph is constructed by merging all paths from all sequences into a graph.

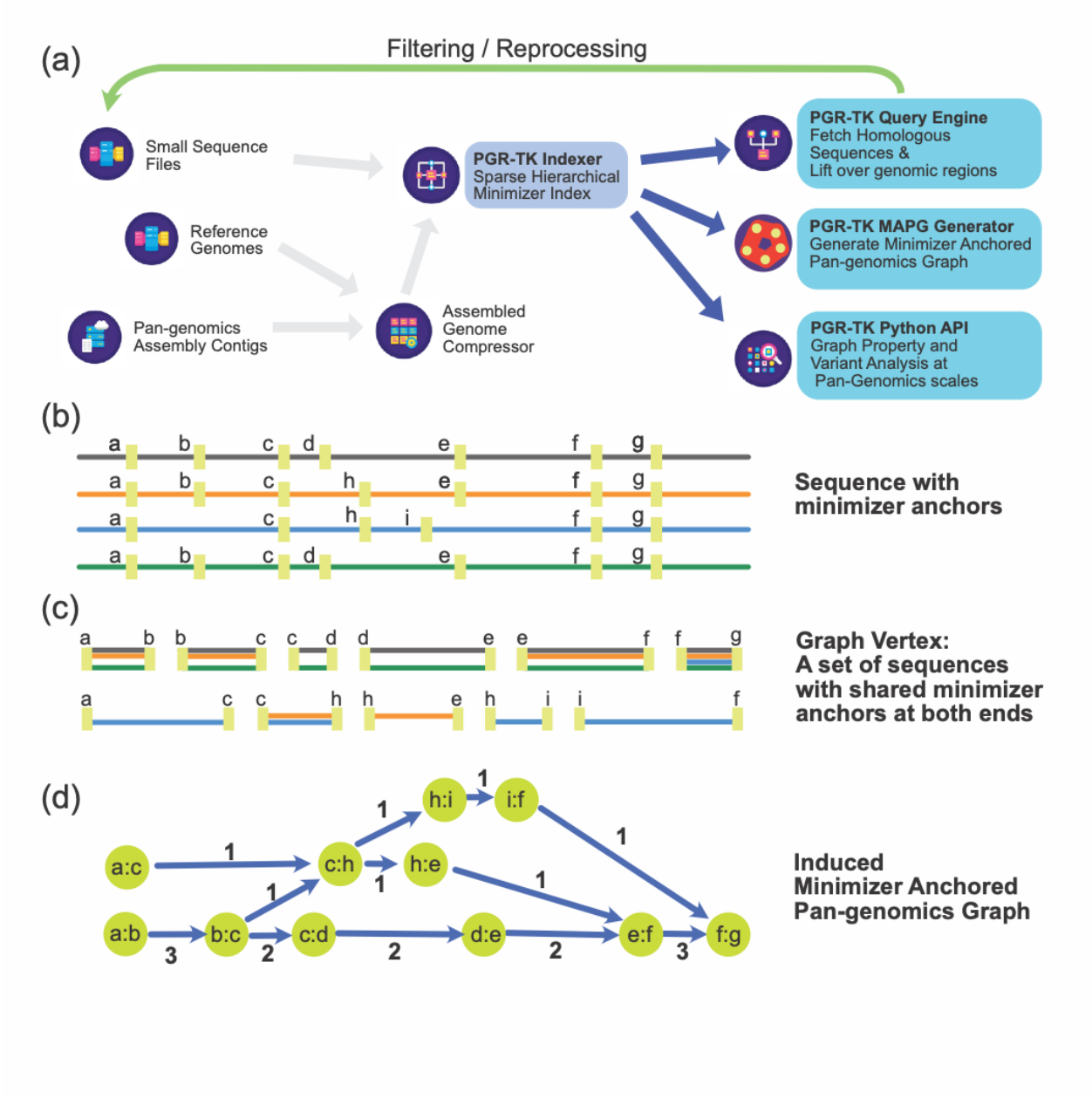


Figure 2. AMY1A MAP-graph in two different scales. (a) The left panel shows a sparse MAP-graph representation of the AMY region with (w, k, r, min_span) = (48, 56, 12, 12). 503 vertices and 699 edges represent the 200 kb~550 kb AMY region. The graph vertices are colored by the principal bundles that correspond to the principal bundle decomposition of selected genomes on the right panel. (gray vertices: the vertices that are not in the principal bundles) (b) The left panel shows a denser MAP-graph with r=4. The graph has 3471 vertices and 2684 edges, about twice as much as the MAP-graph in (a). The principal bundle decomposition reveals a more detailed repeat structure than in (a).

Figure 3: (a) The MAP-Graph generated by PGR-TK (b) The principal bundle decomposition and annotated HLA Class II genes in each of the haplotype sequences. The auxiliary tracks below each sequence on the left panel show the locations of the genes. The colors of the auxiliary tracks match the gene list of genes identified for each haplotype on the right. (c) PCA plots of the MHC Class II sequences. Each panel highlights the different gene haplotype combinations. The vertical color bars indicate the matched haplotype groups in (b) and (c). The circled symbols indicate the haplotypes belong to the corresponding group. The dotted lines represent the connection between the two haplotypes of individuals included in the analysis set who possess both haplotypes.

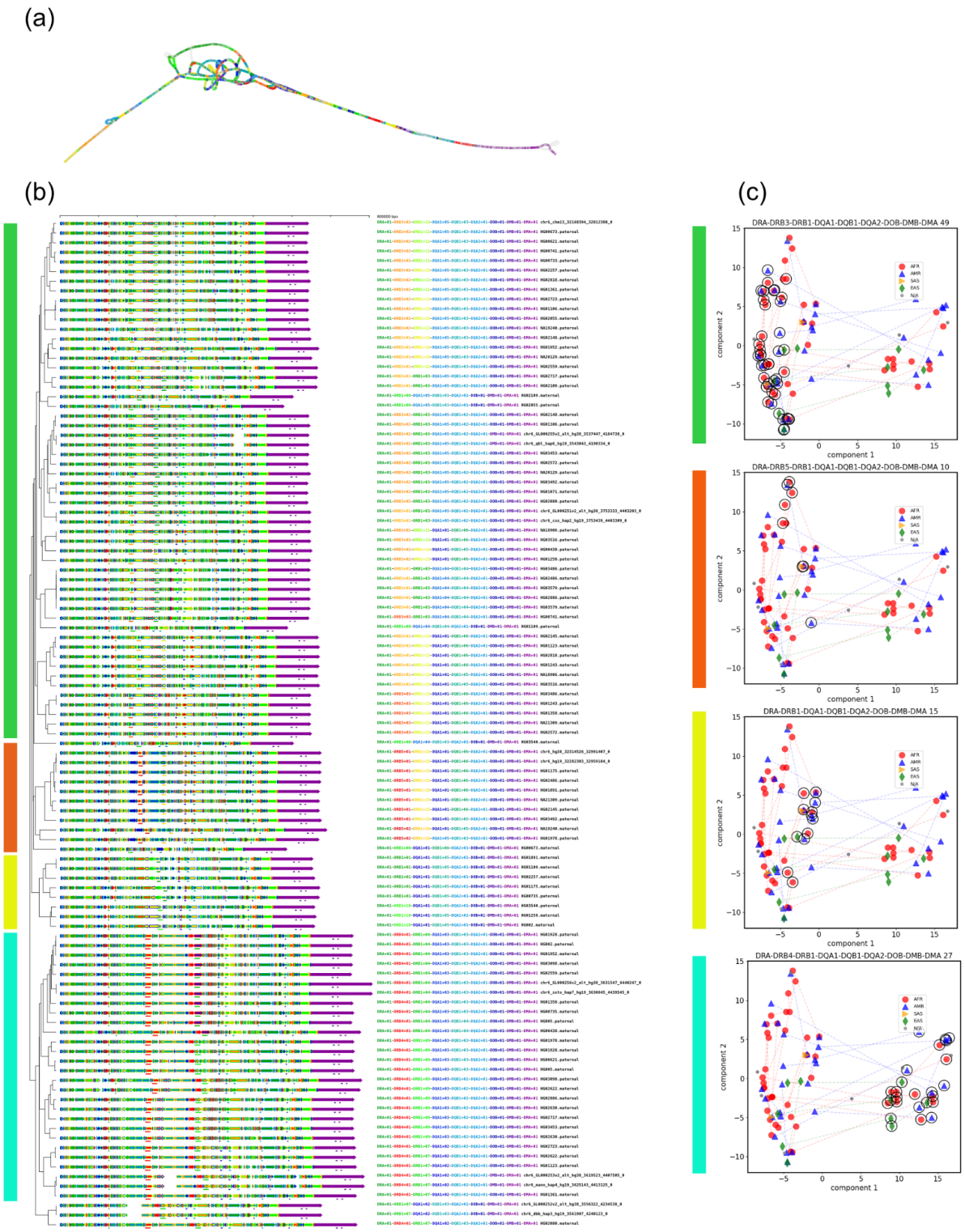
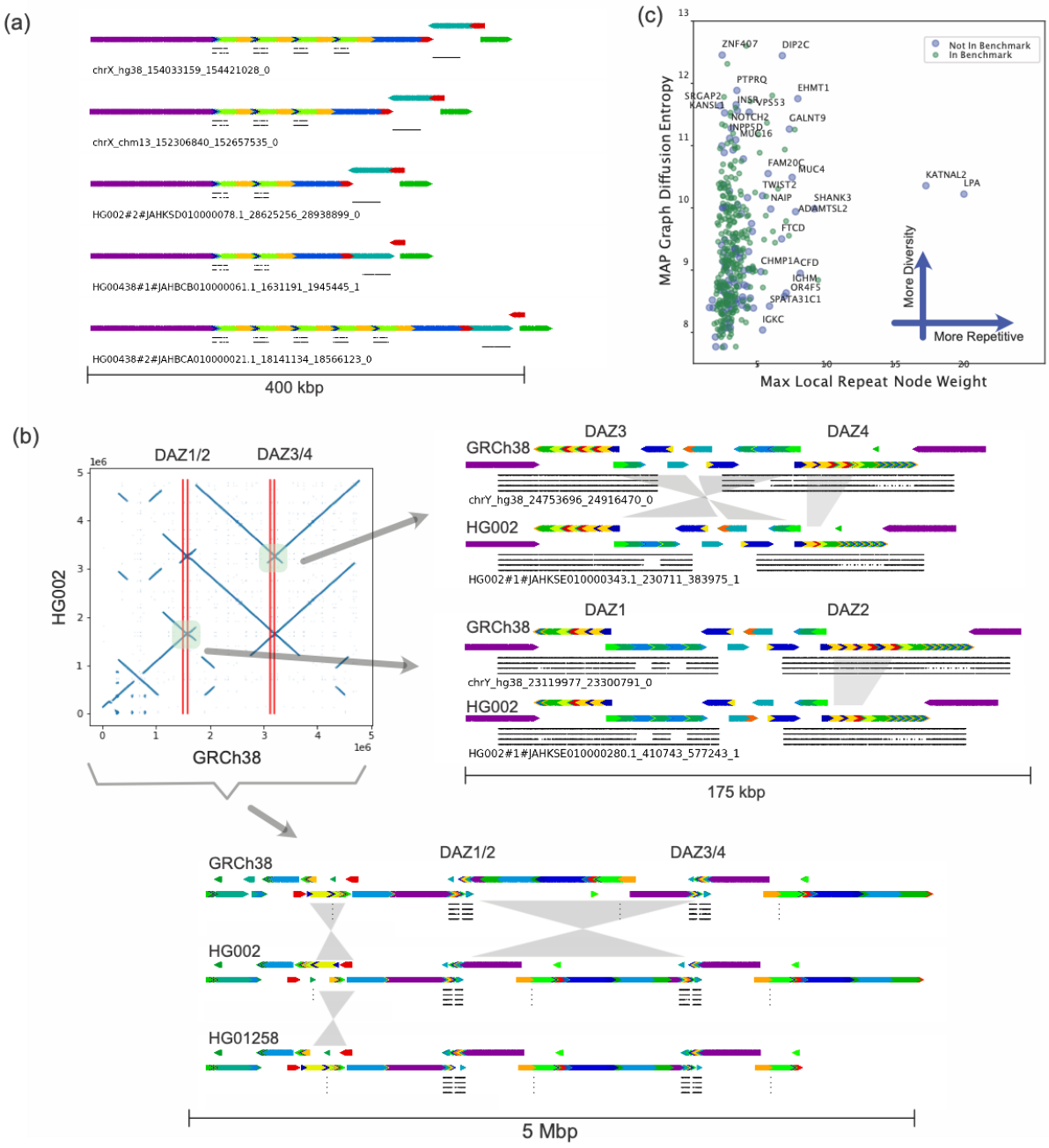

(a)
(b)
(c)
DRA-DRB3-DRB1-DQA1-DQB1-DQA2-DOB-DMB-DMA 49
DRA-DRB5-DRB1-DQA1-DQB1-DQA2-DOB-DMB-DMA 10
DRA-DRB1-DQA1-DQB1-DQA2-DOB-DMB-DMA 15
DRA-DRB4-DRB1-DQA1-DQB1-DQA2-DOB-DMB-DMA 27
component 1
component 2
AFR
AMR
SAS
EAS
NA

Figure4:

(a) MAP-graph principal bundle decomposition shows the repeat number changes of the OPN1LW, OPN1MW1/2/3 to FLNA loci. Auxiliary tracks: top OPN1LW, middle OPN1MW1/2/3, bottom FLNA.

(b) The upper left image displays a dot-plot comparing the HG002 assembly to GRCh38 over a 5Mb region containing the DAZ1/2/3/4 loci, highlighting an inversion between DAZ1/2 and DAZ3/4. The image on the right provides a detailed view of the rearrangements at the gene scale level, with four tracks indicating the local matches to DAZ1/2/3/4 from top to bottom. Comparison to GRCh38's DAZ2 reveals that the HG002 assembly is missing a segment (~10kb) of the darker green. The intergenic region between DAZ3 and DAZ4 also displays a rearrangement that can be described as an incomplete inversion or separate insertions and deletions. The bottom image shows a rearrangement at the whole locus, including all DAZ1/2/3/4 over a 5Mb region. The principal bundle decomposition reveals the different inverted structure of the HG002 T2T assembly and the HG1258 assembly compared to GRCh38.

(c) MAP-graph diffusion entropy vs. repetitiveness survey for the 385 GIAB challenge CMRGs.


(a)
chrX_hg38_154033159_154421028_0
chrX_chm13_152306840_152657535_0
HG002#2#JAHKSD010000078.1_28625256_28938899_0
HG00438#1#JAHBCB010000061.1_1631191_1945445_1
HG00438#2#JAHBCA010000021.1_18141134_18566123_0
400 kbp
(c)
Not In Benchmark
In Benchmark
ZNF407
DIP2C
PTPRQ
EHMT1
SRGAP2
KANSL1
INSR
VPS53
NOTCH2
GALNT9
INPPSD
MUC16
FAM20C
MUC4
TWIST2
NAIP
SHANK3
ADAMTSL2
FTCD
CHMP1A
CFD
IGHM
OR4F5
SPATA31C1
IGKC
KATNAL2
LPA
More Diversity
More Repetitive
MAP Graph Diffusion Entropy
Max Local Repeat Node Weight
(b)
DAZ1/2
DAZ3/4
HG002
GRCh38
1e6
DAZ3
DAZ4
GRCh38
chrY_hg38_24753696_24916470_0
HG002
HG002#1#JAHKSE010000343.1_230711_383975_1
DAZ1
DAZ2
GRCh38
chrY_hg38_23119977_23300791_0
HG002
HG002#1#JAHKSE010000280.1_410743_577243_1
175 kbp
GRCh38
DAZ1/2
DAZ3/4
HG002
HG01258
5 Mbp

Supplementary Material

Building Index

For a large sequence set, e.g. 47 whole genome HPRC assemblies, PGR-TK utilizes the AGC format[46] to store the sequence efficiently. A command line tool "pgr-mdb" is developed with the PGR-TK package to create the index file on top of the AGC file. For example, for a pre-build HPRC year one assembly AGC file (1.33Gb), we create a file (`/data/pgr-tk-HGRP-y1-evaluation-set-v0_input`) include a file system path to the AGC file, `/data/gr-tk-HGRP-y1-evaluation-set-v0.agc`, and call `pgr-mdb` to create the index files with a pre-specified prefix (`/data/pgr-tk-HGRP-y1-evaluation-set-v0`):

```
echo /data/pgr-tk-HGRP-y1-evaluation-set-v0.agc
 > /data/pgr-tk-HGRP-y1-evaluation-set-v0_input


/code/pgr-mdb /data/pgr-tk-HGRP-y1-evaluation-set-v0_input \
 /data/pgr-tk-HGRP-y1-evaluation-set-v0
```

Two files will be generated in this example:

```
/data/pgr-tk-HGRP-y1-evaluation-set-v0.mdb # 15 Gb for (w, k, r, min_span) = (80,56,4,64)
/data/pgr-tk-HGRP-y1-evaluation-set-v0.midx # 3.1 Mb
```

The index and sequence data can be loaded into a python workspace by

```
import pgrtk
sdb = pgrtk.SeqIndexDB()
sdb.load_from_agc_index("pgr-tk-HGRP-y1-evaluation-set-v0")
```

As the indexes are loaded into memory, we suggest using a computing instance which has a random access memory larger than about 4x of the index file to avoid swapping thrashing.

For smaller sequence files, the sequence database object (e.g. the "sdb" in the example above) created by `pgrtk.SeqIndexDB()` can create and load sequences using `load_from_fastx()` method. See the library documentation at https://genedx.github.io/pgr-tk/ for more detailed descriptions of all python objects, methods, and functions in the PGR-TK package.

Query Sequence in the PGR-TK Sequence Database

Command Line example:

A command line tool named `pgr-query` is provided to query a PGR-TK database with a set of sequences, each of which represents a region of interest. It is recommended to select regions

larger than 20kb, as they contain enough SHIMMER anchors. If smaller regions are of interest, padding with flanking sequences can improve the results.

The follow command shows an example querying the database:

```
pgr-query /data/pgr-tk-HGRP-y1-evaluation-set-v0 ROI_seq.fa pg_seqs --merge-range-tol 100000
```

In this example, the `ROI_seq.fa` file contains sequences from the regions of interest. The pgr-query tool generates a set of fasta files with the prefix "pg_seqs" for each query sequence in the "`ROI_seq.fa`" file. Additionally, a "`pg_seqs.hit`" file is produced, which contains information about the alignment range between the query and the results, as well as the number of anchors identified between each pair of query results. This information can be used to filter out unwanted alignments.

Python API example:

For finding homologous sequences in a PGR-TK database, we need to start with a query sequence. We can fetch a sequence in the database giving a known "`data source`", "`contig name`" tuple and the beginning and ending coordinates. As the SHIMMERs are sparsely distributed in a sequence, the query sequence should be long enough to cover enough minimizer anchors. The python statement shows a typical code fragment to generate query results of a region of interest:

```
ref_file_name, roi_chr, roi_b, roi_e = 'hg19_tagged.fa', "chr6_hg19", 32130918, 32959917
padding = 10000
#get a segment of a reference
roi_seq = ref_db.get_sub_seq(ref_file_name, roi_chr, roi_b-padding, roi_e+padding)
# using the roi_seq to find hits in "sdb"
aln_range = pgrtk.query_sdb(sdb, roi_seq, merge_range_tol=200000)
```

The output `aln_range` from the `query_sdb()` call contains data of the hits in the PGR-TK database. Internally, the query_sdb() method performs:

(1) create SHIMMER pairs of the query sequence
(2) use the SHIMMER pairs and the hashmap index to find all hits in the database
(3) perform sparse dynamic programming to find sparse alignments between the query sequence and all hits in the database
(4) merge the alignment segments if any of them are within the `merge_range_tol` parameter.

The parameter `merge_range_tol` is introduced to avoid alignment fragmentation when the query sequence contains a region of high polymorphism but we still want to fetch those diverse sequences for constructing the pan-genomics graph.

Typically, a user needs to process the data in `aln_range` for different analysis. Our example Jupyter Notebooks provides various examples for processing the output to generate dot-plot or MAP-graphs, etc.

Build Minimizer Anchor Pangenome Graph and Principal Bundle Decomposition

Command Line example:

Given a set of sequence in a fasta file, e.g., the query results from `pgr-query` command, we can build the pangenome graph and the principal bundle decomposition (outputted as a bed file) by

```
pgr-pbundle-decomp -w 48 -k 56 -r 8 \
                   --min-span 12 --bundle-length-cutoff 100 \
                   --bundle-merge-distance 1000 --min-branch-size 8 \
                   --min-cov 0 --include file_contain_contig_names \
                   pgr-query pgr_out
```

In the pangenome graph construction process, the following options can be used to control the graph construction: `-w`, `-k`, `-r`, and `--min-span`. The --min-cov option sets the minimum coverage requirement for a vertex to be included in the principal bundle graph. The `--min-branch-size` option allows the user to filter out short branches that contain less than a specified number of vertices in the MAP-Graph. The `--bundle-length-cutoff` option allows the user to exclude bundles shorter than the specified length. When two bundles have the same identifier and are within the distance specified by `--bundle-merge-distance`, they will be merged. If the `--include` option is specified, only those contigs specified in the `file_contain_contig_names` file will be analyzed.

The command generates a set of files with the prefix `pgr_out`:

```
pgr_out.bed # the bed file contains the principal bundle decomposition
pgr_out.ctg.summary.tsv # summary for bundle statistics for each contig
pgr_out.mapg.gfa # MAP-graph in GFA format
pgr_out.mapg.idx # index for the sequence in the GFA file
pgr_out.pmapg.gfa #principal bundle graph in the GFA format
```

The pgr_out.bed has the following format. Each line is a bundle contain in a contig:

`contig_name begin_coordinate end_coordinate bundle_speficiation`

where the bundle_speficiation is six fields delimited by "`:`". The fields are bundle identifier, bundle vertex count, orientation, begin vertex number , end vertex number, "R" or "U" for repetitive or unique in the contig. Note such a region in a contig may not only project to the full bundle. In such cases, end_vertex-bgn_vertex is less than bundle_vertex_count.

The `*.mapg.idx` file contain the information of each vertex in the gfa files. It contains three kinds of tagged lines start with "K", "C", and "F". The "K" line specifies the parameters used to generate the graph. The "C" lines specify the contigs contained in the graph. The "F" lines specify the fragment of sequences contained in each vertex. Here are the fields for echo of them:

K line: w, k, r, min_span
S line: contig_identifer, contg_name, contig_source, contig_length
F line: frag_unique_identifier, frag_numeric_identifer, contig_identifer, start_coordinate, end_coordinate, orientation

Python API:

To build the MAP-Graph and principal bundle decomposition within a python program, one can use the instance methods `generate_mapg_gfa()` and `get_principal_bundle_decomposition()` of an `pgrtk.SeqIndexDB()` object. Please read the documentation at https://genedx.github.io/pgr-tk/ for the API details

Clustering Principal Bundle Decomposition Structure

The follow command perform clustering of the principal bundles store in a bed file:

pgr-pbundle-bed2dist `pgr_out.bed pgr_out.`

It will generate these output files:

```
pgr_out.dist # all pairwise distance
pgr_out.nwk  # a hierarchical tree in Newick format
pgr_out.ddg  # a file contain the dendrogram for the pgr-pbundle-bed2svg to draw the
dendrogram panel
```

Generate Principal Bundle Decomposition Plot in Scalable Vector Graph Format

The following is an example to generate a SVG file pgr_out.svg from pgr_out.bed that layout 500,000 bp with annotation specified in a file called pgr_annotation with a dendrogram panel on the left from the pgr_out.ddg dendrogram data

```
/code/pgr-pbundle-bed2svg pgr_out.bed pgr_out \
              --track-range  500000 --track-tick-interval 100000 \
              --track-panel-width 1200 --stroke-width 0.5 \
              --annotations pgr_annotation  \
              --ddg-file pgr_out.ddg \
              --highlight-repeats 3
```

Performance Evaluation

Sequence Query

While PGR-TK is not designed for creating sequence alignments, the query sequence to database query provided function to identify homologuos sequences in the database to the query sequences. We compare the computing resource for such utility in PGR-TK to minimap2, currently the state of are for fast sequence alignment. For a set of ten selected regions for querying 11 haplotype pangenome references, PGR-TK can index all genome in parallel and provide comparable query time.

Supplementary Table 1

	Tool

Computation Resources	pgr-tk	minimap2
index time (elapsed time)	6 min 43 sec (agc=2:59 + index =2:44)	13 min 07 sec
query time	8.45 sec (including fetching the sequences)	17.93 sec
sequnece storage	807 Mb	9.0 Gb
index storage	2.0 Gb	75.9 Gb

Our testing pangenome dataset contain the assemblies from the HGRP samples:

```
HG00438.maternal, HG00438.paternal, HG00621.maternal, HG00621.paternal,
HG00673.maternal, HG00673.paternal, HG00735.maternal, HG00735.paternal,
HG00741.maternal, HG00741.paternal
```

minimap2:
source: https://github.com/lh3/minimap2
revision: `01b98e8e52a8acfed5a9d57853f028267eaf045f`

commands:

Minimap2 index:

```
\time -v ./minimap2/minimap2 HG00438.maternal.f1_assembly_v2_genbank.fa.gz -t 16 -d
HG00438.maternal.f1_assembly_v2_genbank.fa.gz.idx &>> minimap_timing1.log
\time -v ./minimap2/minimap2 HG00438.paternal.f1_assembly_v2_genbank.fa.gz -t 16 -d
HG00438.paternal.f1_assembly_v2_genbank.fa.gz.idx &>> minimap_timing1.log
\time -v ./minimap2/minimap2 HG00621.maternal.f1_assembly_v2_genbank.fa.gz -t 16 -d
HG00621.maternal.f1_assembly_v2_genbank.fa.gz.idx &>> minimap_timing1.log
\time -v ./minimap2/minimap2 HG00621.paternal.f1_assembly_v2_genbank.fa.gz -t 16 -d
HG00621.paternal.f1_assembly_v2_genbank.fa.gz.idx &>> minimap_timing1.log
\time -v ./minimap2/minimap2 HG00673.maternal.f1_assembly_v2_genbank.fa.gz -t 16 -d
HG00673.maternal.f1_assembly_v2_genbank.fa.gz.idx &>> minimap_timing1.log
\time -v ./minimap2/minimap2 HG00673.paternal.f1_assembly_v2_genbank.fa.gz -t 16 -d
HG00673.paternal.f1_assembly_v2_genbank.fa.gz.idx &>> minimap_timing1.log
\time -v ./minimap2/minimap2 HG00735.maternal.f1_assembly_v2_genbank.fa.gz -t 16 -d
HG00735.maternal.f1_assembly_v2_genbank.fa.gz.idx &>> minimap_timing1.log
\time -v ./minimap2/minimap2 HG00735.paternal.f1_assembly_v2_genbank.fa.gz -t 16 -d
HG00735.paternal.f1_assembly_v2_genbank.fa.gz.idx &>> minimap_timing1.log
\time -v ./minimap2/minimap2 HG00741.maternal.f1_assembly_v2_genbank.fa.gz -t 16 -d
HG00741.maternal.f1_assembly_v2_genbank.fa.gz.idx &>> minimap_timing1.log
\time -v ./minimap2/minimap2 HG00741.paternal.f1_assembly_v2_genbank.fa.gz -t 16 -d
HG00741.paternal.f1_assembly_v2_genbank.fa.gz.idx &>> minimap_timing1.log
\time -v ./minimap2/minimap2 chm13.draft_v1.1.fasta.gz -t 16 -d chm13.draft_v1.1.fasta.gz.idx
&>> minimap_timing1.log
```

Minimap2 query:

```
cat << EOF | \time -v parallel -j 16 2> minimap_timing.log 1> minimap.hits
./minimap2/minimap2 -x asm5 HG00438.maternal.f1_assembly_v2_genbank.fa.gz.idx ROI_seq.fa
./minimap2/minimap2 -x asm5 HG00438.paternal.f1_assembly_v2_genbank.fa.gz.idx ROI_seq.fa
./minimap2/minimap2 -x asm5 HG00621.maternal.f1_assembly_v2_genbank.fa.gz.idx ROI_seq.fa
./minimap2/minimap2 -x asm5 HG00621.paternal.f1_assembly_v2_genbank.fa.gz.idx ROI_seq.fa
./minimap2/minimap2 -x asm5 HG00673.maternal.f1_assembly_v2_genbank.fa.gz.idx ROI_seq.fa
./minimap2/minimap2 -x asm5 HG00673.paternal.f1_assembly_v2_genbank.fa.gz.idx ROI_seq.fa
./minimap2/minimap2 -x asm5 HG00735.maternal.f1_assembly_v2_genbank.fa.gz.idx ROI_seq.fa
```

```
./minimap2/minimap2 -x asm5 HG00735.paternal.f1_assembly_v2_genbank.fa.gz.idx ROI_seq.fa
./minimap2/minimap2 -x asm5 HG00741.maternal.f1_assembly_v2_genbank.fa.gz.idx ROI_seq.fa
./minimap2/minimap2 -x asm5 HG00741.paternal.f1_assembly_v2_genbank.fa.gz.idx ROI_seq.fa
./minimap2/minimap2 -x asm5 chm13.draft_v1.1.fasta.gz.idx ROI_seq.fa
EOF
```

prg-tk:
revision: 75fa20b41592941c9e6eef3f914d97788ee06b86
commands:

```
ls *.fa.gz > agc_inputs
\time -v ~/benchmark/pgr-tk/agc/agc create chm13.draft_v1.1.fasta.gz -i agc_inputs >
test.agc 2>> timing.log
echo test.agc > pgr_input
\time -v pgr-mdb pgr_input test 2>> timing.log
\time -v pgr-query test ROI_seq.fa pgr-query-out 2> pgr-query-out.log
```

Here is the list of the testing query sequences:

Regions of interest for testing querying					
Name	Reference	Chromosom	begin	end	Strand
MHC-C2	GRCh38	chr6	32313513	32992088	
RCCE	GRCh38	chr6	31976719	32117146	0
AMY	GRRh38	chr1	103542345	103798299	0
LPA	GRRh38	chr6	160529904	160666180	0
IGH	GRRh38	chr14	106205008	106874830	0
HLA-CB	GRRh38	chr6	31143427	31484914	0
ABO	GRRh38	chr9	133163441	133361030	0
TSPY1	GRRh38	chrY	9294496	9591276	0
15q15	GRRh38	chr15	43531685	43769928	0
16p21	GRRh38	chr16	28139916	28830868	0

Supplementary Table 2: We compared the query results for two selected regions and found them to be consistent. In these two cases, due to differences in their design, the "pgr-query" command only produced a single aligned region for each reference assembly, rather than multiple supplementary alignments. Similar to minimap2, "pgr-query" provides additional information about the hits, allowing the user to apply filters and define criteria to eliminate false positive alignments caused by repeats in more complex scenarios.

	pgr-query results		minimap2 results		consistent
MHC Class 2	begin	end	begin	end	
HG00438#1#JAHBCB010000040.1	23357242	24010477	23356506	24011428	Yes

HG00438#2#JAHBCA010000042.1	23362233	24141470	23361497	24142421	Yes
HG00621#1#JAHBCD010000020.1	23356282	24115439	23352369	24116390	Yes
HG00621#2#JAHBCC010000005.1	32265868	32906962	32264542	32910924	Yes
HG00673#1#JAHBBZ010000030.1	32179011	32823596	32177436	32824276	Yes
HG00673#2#JAHBBY010000031.1	886239	1474176	884899	1475258	Yes
HG00735#1#JAHBCH010000013.1	32366232	33038084	32364489	33042050	Yes
HG00735#2#JAHBCG010000038.1	3651996	4413560	3650989	4414240	Yes
HG00741#1#JAHALY010000025.1	23365046	24008887	23363908	24010574	Yes
HG00741#2#JAHALX010000077.1	25645945	26293199	25644213	26294150	Yes
chm13 chr6	32168394	32812380	32166819	32813462	Yes
AMY					
HG00438#1#JAHBCB010000015.1	33562835	33889251	33561142	33889429	Yes
			33657228	33801412	
			33761202	33865960	
HG00438#2#JAHBCA010000012.1	51600987	51760853	51599294	51761031	Yes
			51611167	51668018	
			51663126	51695258	
			51695374	51737568	
HG00621#1#JAHBCD010000034.1	2187959	2536123	2187725	2537760	Yes
			2211188	2315970	
			2253384	2410116	
			2469036	2509286	
HG00621#2#JAHBCC010000031.1	16758012	17011992	16756319	17012170	Yes
HG00673#1#JAHBBZ010000075.1	16748192	16908034	16746499	16908212	Yes
			16758372	16815202	
			16810310	16842663	
			16842559	16884745	
HG00673#1#JAHBBZ010000329.1	312	93447	22834	50859	
			22920	48155	
			22930	48307	
			24483	94099	
			8	87032	
HG00673#2#JAHBBY010000109.1	16763372	17111516	16761679	17111694	Yes
HG00735#1#JAHBCH010000004.1	101636258	101890244	101634565	101890422	Yes
HG00735#2#JAHBCG010000068.1	4817143	4977006	4815450	4977184	Yes

			4827323	4884173	
			4879281	4911562	
			4911529	4953720	
HG00741#1#JAHALY010000007.1	18847041	19219687	18846807	19221324	Yes
			18931609	19009465	
			19037277	19125244	
HG00741#2#JAHALX010000013.1	51662338	51916331	51660645	51916509	Yes
			51672517	51729368	
			51756716	51850848	
			51802317	51893042	
chm13:chr1	103392985	103835251	103391292	103835429	
			103680709	103699433	

In **Supplementary Table 3**, we compared the results of the pgr-query and minimap2 for a test set of pangenomic sequences. While pgr-query is designed to fetch homologous sequences from the database using long query sequences, rather than as a general sequence aligner, it is important to demonstrate its performance in fetching sequences accurately. We compared all hits larger than 10 kb from the pgr-query output for a set of 395 query sequences from the CMRG to the minimap2 output and found that the results are highly consistent in most cases. Some discrepancies are due to (1) short hits and (2) low minimap2 mapQV output. The pgr-query output might be more sensitive, such that some repetitive sequences are in the query output without proper filtering.

Supplementary Table 3a, We compare the unfiltered pgq-query output and the filtered output with hits that has more than 5 minimizer anchors found with the minimap2 output unfiltered or filtered by MapQV. Base on the hits from minimap2, pgr-query captures 97% to 99% hits, depending on the filtering criteria.

	Minimap2 (All)	Minimap2 (MapQV > 30)
minimap2 hits	3668	3190
overlapped pgr-query hits (filtered)	3601	3164
80% overlapped percentage	98.17%	99.18%
overlapped pgr-query hits (filtered)	3573	3164
80% overlapped percentage	97.41%	99.18%

Supplementary Table 3b, Base on the hit output from pgr-query, minimap2 capture 85% to 94% hits, depending on filtering certeria.

	unfiltered	filtered
pgr-query hits	3637	3589
overlapped minimap2 hits	3397	3396
80% overlapped percentage	93.40%	94.62%
overlapped minimap2 hits (MapQV>30)	3104	3103
80% overlapped percentage	85.35%	86.46%

Pangenome Graph Construction

The measure resource usage for making index with different parameter

Data: 97 haplotype human genome assembly							
w	k	r	Index file size (Gb)	elapse time (min:sec)	User Space CPU time (s)	System CPU time (s)	Memeory Usage (kbytes)
80	56	12	3	13:11.09	10749	455	35605852
80	56	8	6.1	14:21.31	10865	462	41329332
80	56	6	9.5	15:50.63	10966	472	48986112
80	56	4	15	18:26.03	11171	488	61270196
80	48	4	15	18:10.64	11159	485	59441552
80	32	4	15	17:58.06	11129	484	59576876
80	24	4	15	17:47.83	11139	482	59178764
64	56	4	17	19:13.54	11393	517	65174852
48	56	4	18	20:14.21	11685	543	70309960

command:

```
echo /wd/data/pgr-tk-HGRP-y1-evaluation-set-v0.agc > input

\time -v pgr-mdb -r 4  input  pgr-tk-HGRP-y1-evaluation-set-v0-r4 >& log_r4
\time -v pgr-mdb -r 6  input  pgr-tk-HGRP-y1-evaluation-set-v0-r6 >& log_r6
\time -v pgr-mdb -r 8  input  pgr-tk-HGRP-y1-evaluation-set-v0-r8 >& log_r8
\time -v pgr-mdb -r 12  input  pgr-tk-HGRP-y1-evaluation-set-v0-r12 >& log_r12

\time -v pgr-mdb -k 48  input  pgr-tk-HGRP-y1-evaluation-set-v0-k48 >& log_k48
\time -v pgr-mdb -k 32  input  pgr-tk-HGRP-y1-evaluation-set-v0-k32 >& log_k32
\time -v pgr-mdb -k 24  input  pgr-tk-HGRP-y1-evaluation-set-v0-k24 >& log_k24
```

```
\time -v pgr-mdb -w 64  input  pgr-tk-HGRP-y1-evaluation-set-v0-w64 >& log_w64
\time -v pgr-mdb -w 48  input  pgr-tk-HGRP-y1-evaluation-set-v0-w48 >& log_w48
```

Supplementary Table 4: Comparison of graph build time to seqwish and minigraph (input sequence data HLA Class II sequence from the 97 pangnome references)

Tool	Command Line	User time (seconds)	System time (seconds)	Elapsed (wall clock) time (m:ss)	memory usage (kb)
seqwish					
command	`wfmash HLA-ClassII_seq.fa HLA-ClassII_seq.fa -t 32 -X`	5270.32	4.39	3:09.82	878516
command	`seqwish -s HLA-ClassII_seq.fa -p HLA-ClassII_seq.paf -g HLA-ClassII_seq.gfa`	74.26	4.37	0:29.22	1619660
minigraph					
command	`minigraph -t 32 -cxggs chm13_HLA_C2.fa MHC*.fa > out.gfa`	799.91	41.2	13:19.61	2273760
pgr-tk					
command	`pgr-pbundle-decomp HLA-ClassII_seq.fa HLA-ClassII`	10.63	1.2	0:04.32	466448
command	`pgr-pbundle-decomp -r 3 HLA-ClassII_seq.fa HLA-ClassII_r3`	12.51	1.32	0:05.46	661628
command	`pgr-pbundle-decomp -r 1 HLA-ClassII_seq.fa HLA-ClassII_r1`	19.71	3.94	0:11.39	1294248

Tool	Command Line	number of vertices	number of edges	average vertex size (bp)	(Graph base length) / (total sequence length)
seqwish					
command	`wfmash HLA-ClassII_seq.fa HLA-ClassII_seq.fa -t 32 -X`				

command	`seqwish -s HLA-ClassII_seq.fa -p HLA-ClassII_seq.paf -g HLA-ClassII_seq.gfa`	121061	196640	122.4	0.1785
minigraph					
command	`minigraph -t 32 -cxggs chm13_HLA_C2.fa MHC*.fa > out.gfa`	293	409	3140.6	0.0111
pgr-tk					
command	`pgr-pbundle-decomp HLA-ClassII_seq.fa HLA-ClassII`	18258	29830	310.458	0.0683
command	`pgr-pbundle-decomp -r 3 HLA-ClassII_seq.fa HLA-ClassII_r3`	25274	40969	233.776	0.0712
command	`pgr-pbundle-decomp -r 1 HLA-ClassII_seq.fa HLA-ClassII_r1`	50773	80572	129.932	0.0795

Software versions used

seqwish:
source: https://github.com/ekg/seqwish.gi
revison:`f362f6f5ea89dbb6a0072a8b8ba215e663301d33`

minigraph
source: https://github.com/lh3/minigraph
revision: `3398263be225ba923140a1081b505b71f2cdf8fb`

pgr-pbundle-decomp (part of PGR-TK)
revision: `75fa20b41592941c9e6eef3f914d97788ee06b86`

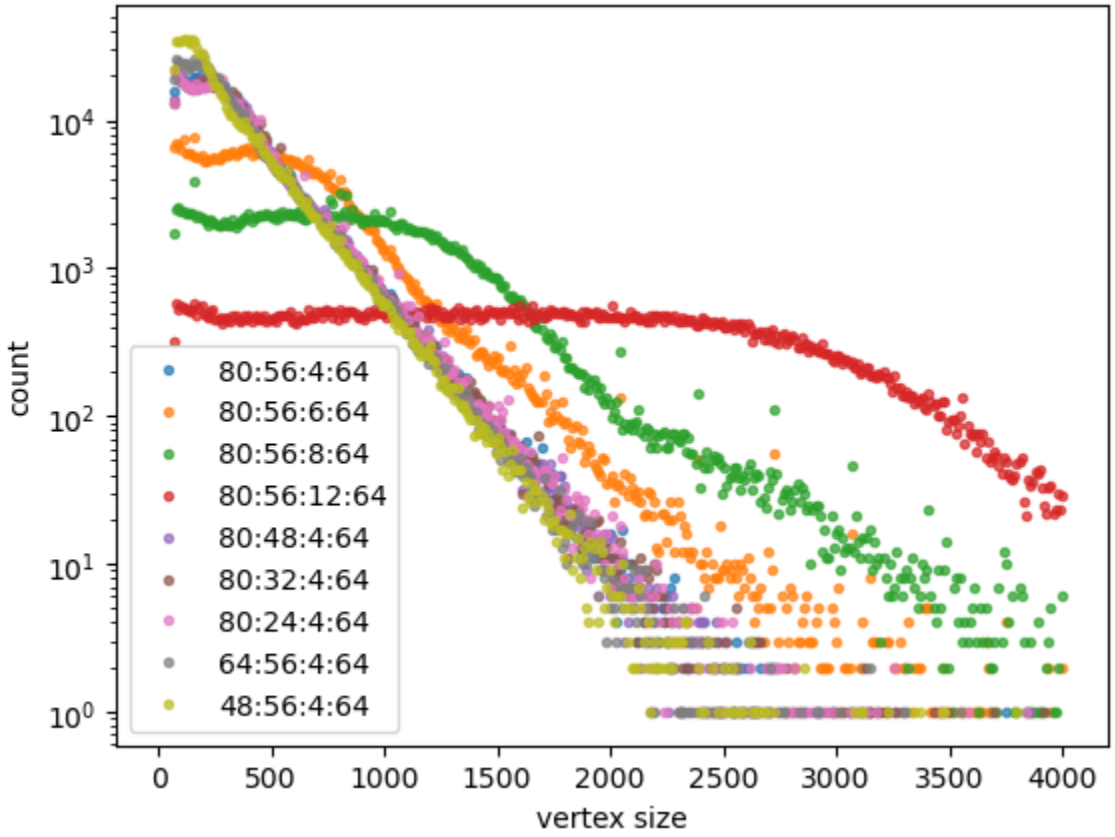
The test sequence file "HLA-ClassII_seq.fa" comprises 147 sequences with an average length of 564,570 base pairs. It is important to note that not all sequences were incorporated in the Minigraph output as certain MHC Class II sequences displayed significant divergence from the CHM13 MHC Class II reference. The ratio of the total number of bases in the Minigraph output to the total number of bases in the input sequence file was observed to be significantly lower compared to the results produced by Seqwish and PGR-TK. The Seqwish graph was denser than the MAP-graph generated by `pgr-pbundle-decomp` and provided more detailed information that could be utilized for the direct identification of base-level differences.

On the other hand, PGR-TK demonstrated a significant advantage in terms of computational efficiency, with a construction time of the pangenome graphs that was 500x faster in terms of user CPU time and 60x faster in terms of wall clock time compared to Seqwish, and 75x faster and 160x faster, respectively, compared to Minigraph using default `pgr-pbundle-decomp` parameters.

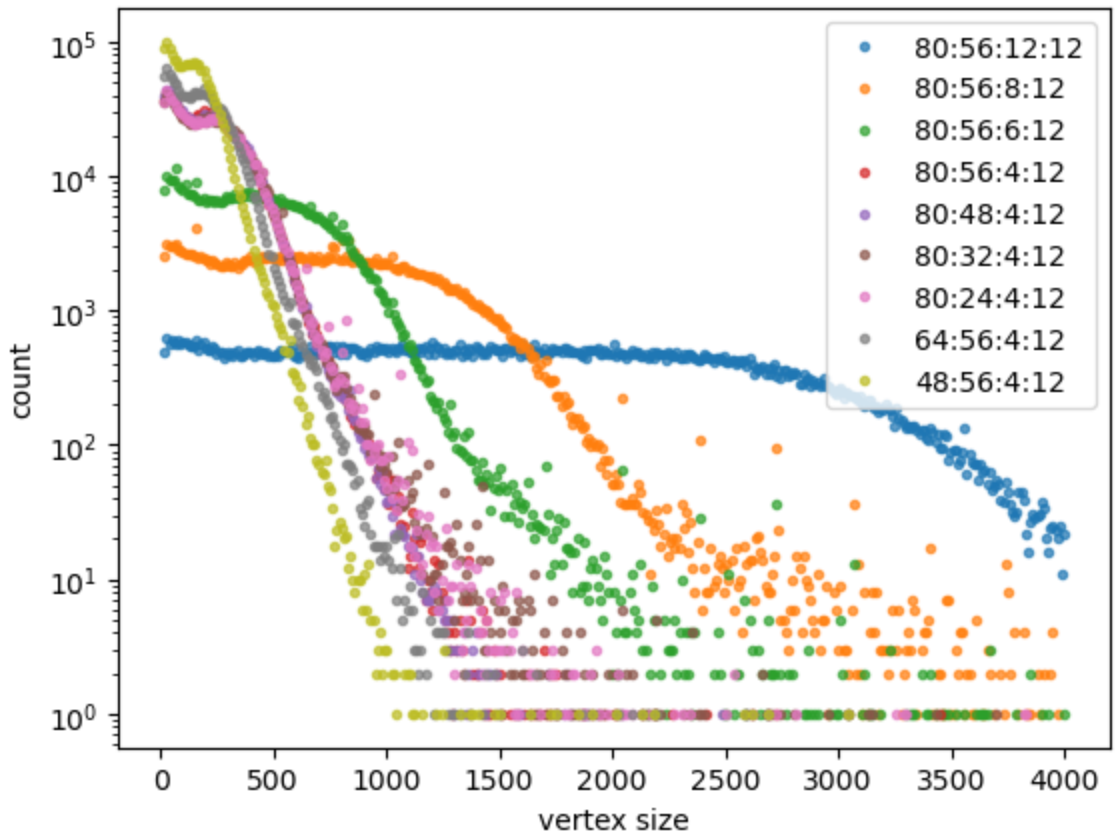
Effect of the Parameter Choice to The MAP Vertex Sizes

Vertex Sizes of the Chromosome 1 of Chm13 with different w and r, (min_span = 64).

Supplementary Figure 1a, The vertex size influences the resolution of the sequence being analyzed. Observing the vertex size distributions using various parameter sets, a flat region is observed followed by an exponential tail. To effectively query the database, it's best to ensure that the query sequence length is a multiple of the average vertex size.



Supplementary Figure 1b, This vertex size distribution plot with the same parameter sets as those in **1a**, but with a small min_span. To reduce excessing minimizer in long simple repeat regions, we remove all pairs of minimizer anchors that are smaller than min_span to reduce unnecessary additional computation resources for processing simple repeat regions.



Supplementary Table 5: the descriptive statistics from the different set of parameter choice.

parameter set (w:k:r:m)	total vertex	media size	mean size	standard deviation	99.9%	99.0%
80:56:12:64	146967	1574	1690.1	1551.1	148956	24605
80:56:8:64	309494	754	802.6	519.4	7821	3388
80:56:6:64	484243	462	512.9	338.5	3551	2202
80:56:4:64	759890	269	326.9	235.0	2283	1600
80:48:4:64	756118	272	328.5	234.9	2248	1583
80:32:4:64	748297	275	331.9	237.4	2404	1620
80:24:4:64	742768	277	334.4	240.8	2349	1623
64:56:4:64	831991	236	298.5	221.4	2146	1515
48:56:4:64	903455	202	274.9	216.7	2124	1483
80:56:12:12	153192	1518	1621.4	1505.4	148956	23945

80:56:8:12	341088	695	728.2	484.5	7655	2841
80:56:6:12	583065	398	426.0	287.9	3003	1588
80:56:4:12	1159085	195	214.3	148.7	1204	822
80:48:4:12	1165335	196	213.1	148.5	1125	809
80:32:4:12	1132410	201	219.3	154.9	1620	930
80:24:4:12	1138924	200	218.1	154.2	1370	895
64:56:4:12	1402189	162	177.1	122.0	1000	688
48:56:4:12	1779213	128	139.6	95.4	777	558

Suggested Parameter Choice For Region of Size Up to 5Mb

Based on our observations in **Supplementary Table 4**, it is clear that the parameter r has the greatest impact on the size of the vertices. To simplify the process, we recommend using the default values of w=48, k=56, and min_span=12 for general cases and adjusting the value of r based on the length of the sequence of interest. This approach can serve as a starting point and can be fine-tuned if specific detailed features are of interest.

Supplementary Table 6

w=48, k=56, min_span=12
r = floor(min(12, max(2, floor(2 * (mean(sequence lengths)/50000)^0.5))))

According to the formula, here is a table for the choice of r of different lengths of the sequences of interest:

sequence length(bp)	r
20,000	2
40,000	2
80,000	4
160,000	5
320,000	8
640,000	11
1,280,000	12
2,560,000	12
5,120,000	12

Generate MAP-Graph and Principal Bundle Decomposition for AMY and MHC region

We use GRCh38 chr6:32,313,513-32,992,088 (for MHC Class II) and GRCh38 chr1:103,54,2345-103,798,299 as the query sequences to find the homologous sequences in the pangenome reference database (pgr-tk-HGRP-y1-evaluation-set-v0):

```
cat << EOF  | tr " " "\t" > regions_interest
MHC-C2 hg38_tagged.fa chr6_hg38 32313513 32992088
AMY hg38_tagged.fa chr1_hg38 103542345 103798299 0
EOF
```

We use the pgr-fetch-seqs command in PGR-TK to get the two references:

```
pgr-fetch-seqs pgr-tk-HGRP-y1-evaluation-set-v0 \
-r regions_interest > ROI_seq.fa
```

Then, we use the pgr-query command to get the sequences in the pangenome reference. We merge the hits that are less than 100kb apart from each other:

```
pgr-query /wd/data/pgr-tk-HGRP-y1-evaluation-set-v0 \
/wd/results/pgr-out/ROI_seq.fa /wd/results/pgr-out/pg_seqs --merge-range-tol 100000
```

After fetching the sequence in the database, we filter out partial aligned contigs. With in the aligned contig, we generate the MAP-graph and the principal bundle decomposition by

```
pgr-pbundle-decomp -w ${w} -k ${k} -r ${r} \
        --min-span ${m} --bundle-length-cutoff 100 --min-branch-size 8 \
        ${fasta_file} /wd/results/pgr-out/${prefix}
```

For MHC class II, we choose w=48, k=56, r=7, m=12, and for AMY1A, we choose w=48, k=56, r=4, m=12 determined by the formula above.

The command pgr-pbundle-decomp generated the MAP-graph as gfa file and the principal bundle decomposition in bed format. For example, the first five bundles of Chm13 of the AMY1A region are represented as

```
chm13_tagged::chr1_chm13_103392985_103835251_0  174     27484   1:203:0:10:202:U
chm13_tagged::chr1_chm13_103392985_103835251_0  27428   27631   16:2:0:0:1:U
chm13_tagged::chr1_chm13_103392985_103835251_0  27575   50412   2:161:0:0:160:R
chm13_tagged::chr1_chm13_103392985_103835251_0  50356   51227   10:6:0:0:5:R
chm13_tagged::chr1_chm13_103392985_103835251_0  51171   55089   8:26:0:0:25:R
```

PGR-TK provides a command line tool for quick all pair-wise sparse alignment and compute distances between all pairwise sequences. With the distance we can perform hierarchical clustering to group bundles with similar structures for analysis or visualization. For example, the following command computes the distance based on the principal bundle decomposition

```
pgr-pbundle-bed2dist ${bed_file} ${prefix}
```

It generates three files:

```
${prefix}.dist # this file contains the distances between sequences
```

```
${prefix}.nwk  # the clustering tree in Newick format

${prefix}.ddg  # the file contains the dendrogram information for plotting a clustering
               # tree alone with the principal bundle decomposition with the command
               # pgr-pbundle-bed2svg
```

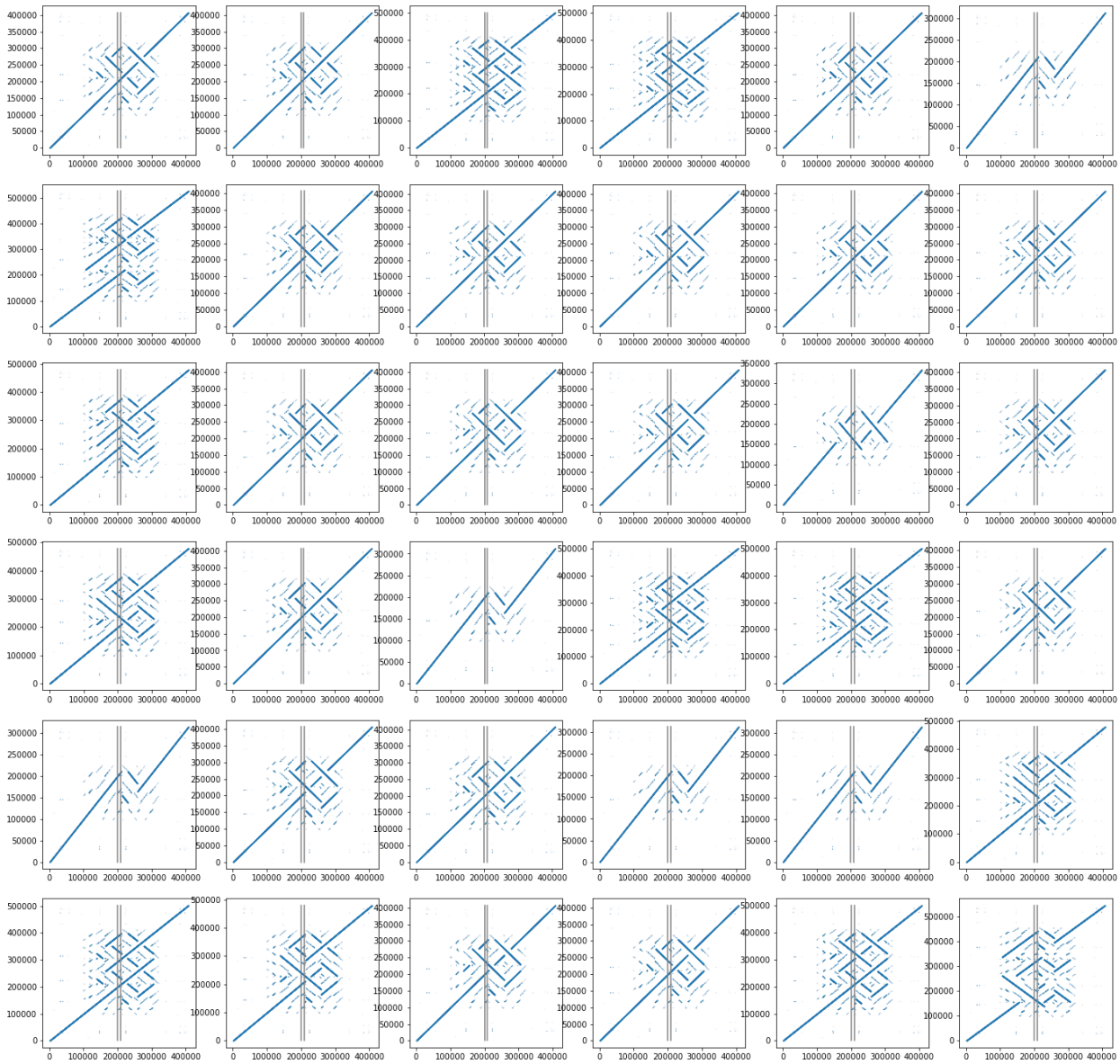
We can generate the principal decomposition plot with the command `pgr-pbundle-bed2svg`. For example, with the follow, we can generate a principal decomposition plot (`${prefix}.svg`) with the clustering dendrogram with annotation specified by a file ${prefix}.ord:

```
pgr-pbundle-bed2svg ${bed_file} ${prefix} \
              --track-range 250000 --track-tick-interval 10000
              --track-panel-width 1200 --stroke-width 1.2 \
              --annotations ${prefix}.ord \
              --ddg-file ${prefix}.ddg"
```

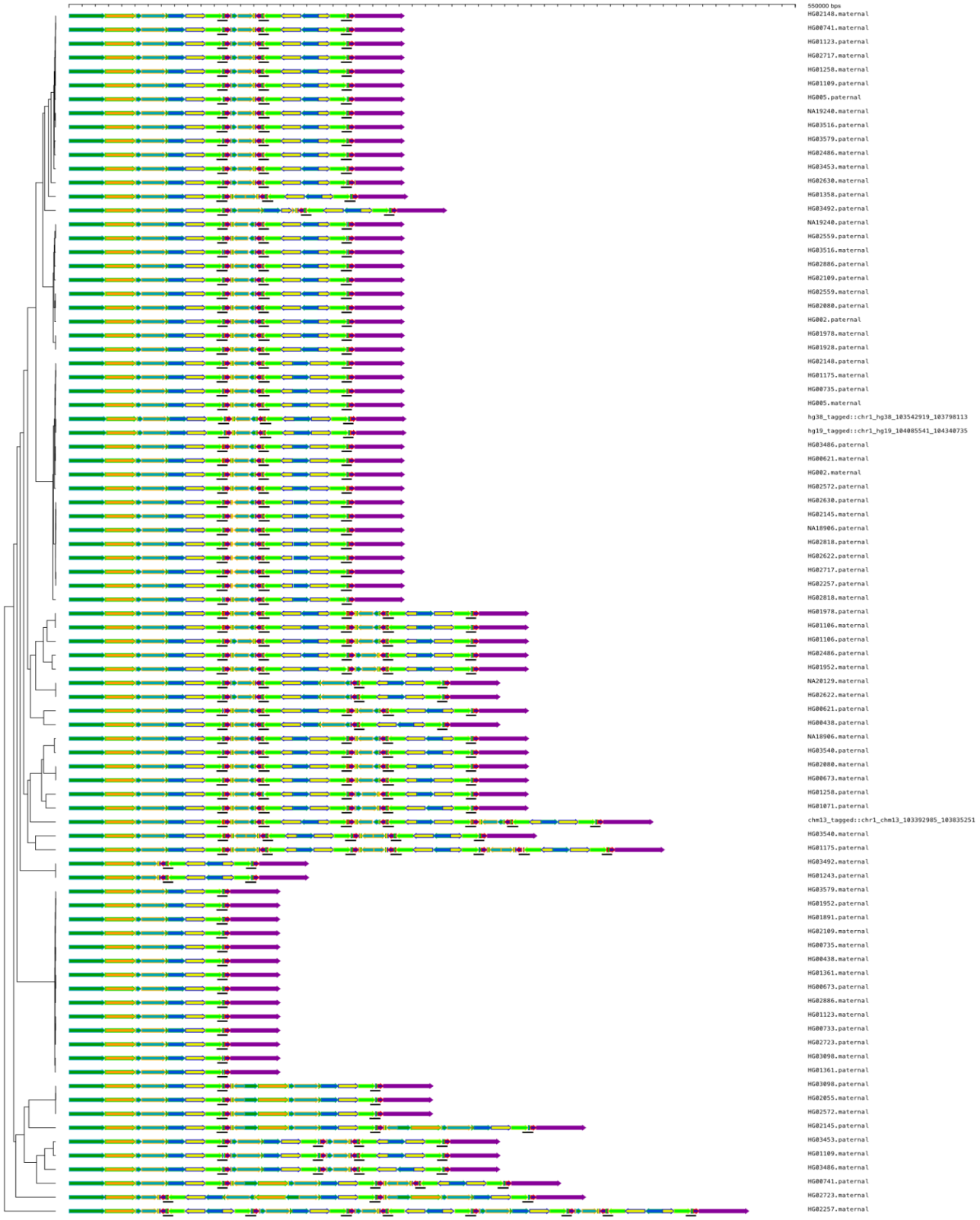
Please see more concrete examples in the git repo: https://github.com/GeneDx/pgr-tk

Supplementary Figure 2a

AMY1A repeat dot plots and principal bundle decomposition plots



Supplementary Figure 2b: The principal bundle plots of the AMY1A repeat regions. The black short bars indicate the regions homologous to AMY1A sequence.


550000 bps
HG02148.maternal
HG00741.maternal
HG01123.paternal
HG02717.maternal
HG01258.maternal
HG01109.paternal
HG005.paternal
NA19240.maternal
HG03516.paternal
HG03579.paternal
HG02486.maternal
HG03453.maternal
HG02630.maternal
HG01358.paternal
HG03492.paternal
NA19240.paternal
HG02559.paternal
HG03516.maternal
HG02886.paternal
HG02109.paternal
HG02559.maternal
HG02080.paternal
HG002.paternal
HG01978.maternal
HG01928.paternal
HG02148.paternal
HG01175.maternal
HG00735.paternal
HG005.maternal
hg38_tagged::chr1_hg38_103542919_103798113
hg19_tagged::chr1_hg19_104085541_104340735
HG03486.paternal
HG00621.maternal
HG002.maternal
HG02572.paternal
HG02630.paternal
HG02145.maternal
NA18906.paternal
HG02818.paternal
HG02622.paternal
HG02717.paternal
HG02257.paternal
HG02818.maternal
HG01978.paternal
HG01106.maternal
HG01106.paternal
HG02486.paternal
HG01952.maternal
NA20129.maternal
HG02622.maternal
HG00621.paternal
HG00438.paternal
NA18906.maternal
HG03540.paternal
HG02080.maternal
HG00673.maternal
HG01258.paternal
HG01071.paternal
chm13_tagged::chr1_chm13_103392985_103835251
HG03540.maternal
HG01175.paternal
HG03492.maternal
HG01243.paternal
HG03579.maternal
HG01952.paternal
HG01891.paternal
HG02109.maternal
HG00735.maternal
HG00438.maternal
HG01361.maternal
HG00673.paternal
HG02886.maternal
HG01123.maternal
HG00733.paternal
HG02723.paternal
HG03098.maternal
HG01361.paternal
HG03098.paternal
HG02055.maternal
HG02572.maternal
HG02145.paternal
HG03453.paternal
HG01109.maternal
HG03486.maternal
HG00741.paternal
HG02723.paternal
HG02257.maternal

Comparing Principal Bundle Decomposition with Different Set of Parameters

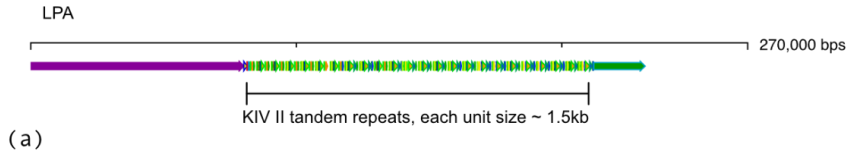
We provide two illustrations of principal bundle decompositions, each with varying scales by changing the parameter sets. The first illustration, shown in **Supplementary Figures 3a** and **3b**, pertains to a 130 kb region of interest containing the LPA KIV-II repeats. The second illustration, also shown in **Supplementary Figures 3c** and **3d**, is of a 2.85 Mbp region located on chromosome 7 from positions 72752602 to 75600937 on GRCh38. This region is known to contain a microdeletion caused by nested repeats, which results in Williams-Beuren syndrome.

In **Supplementary Figure 3a**, there are 231 bundle segments spanning the 103,538 bp CHM13 LPA sequence, while only 93 bundle segments are present in **Supplementary Figure 3b**. The sparser decomposition with r=12 in **Supplementary Figure 3b** for this region may not provide sufficient detail for analyzing repeat elements in the sequences

In contrast, for the large 2.85 Mbp region, the choice r=12 provides a better representation of the overall structure (**Supplementary Figure 3c**), as it contains only 251 bundle segments out of the 2,916,749 bp chromosome 13 sequence, compared to the r=6 choice (**Supplementary Figure 3d**), which has 685 bundle segments. The higher number of bundle segments in the r=6 choice results in over-fragmentation of the sequences, making it more difficult to identify interesting repeats.

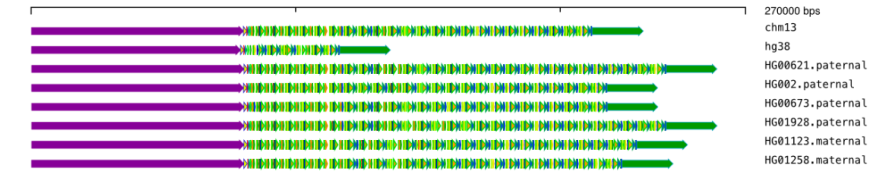
The `pgr-pbundle-decomp` command-line tool generates a summary of all contigs, providing valuable information for analysis by reporting the total number and average lengths of repetitive and non-repetitive fragments. This information can help the user make informed decisions if the default parameters suggested in **Supplementary Table 6** do not capture the desired features when comparing pangenome sequences.

Supplementary Figure 3



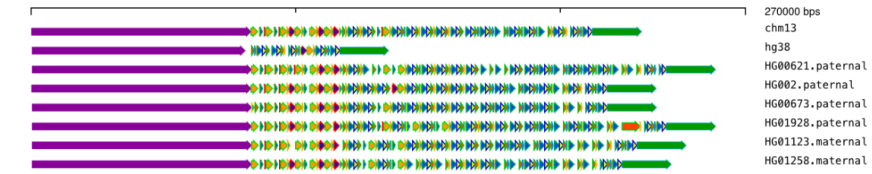
(a)

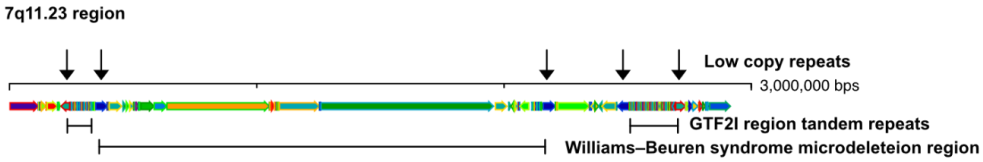
```
pgr-pbundle-decomp -w 48 -k 56 -r 4 --min-span 12 --bundle-length-cutoff 100
     --min-branch-size 8 pg_seqs.fa  LPA_pg_seqs. --min-cov 0
```



(b)

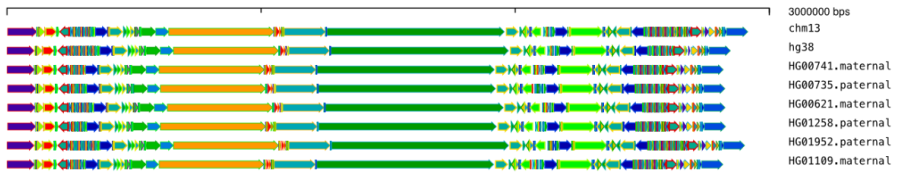
```
pgr-pbundle-decomp -w 48 -k 56 -r 12 --min-span 12 --bundle-length-cutoff 100
     --min-branch-size 8 pg_seqs.fa  LPA_pg_seqs. --min-cov 0
```





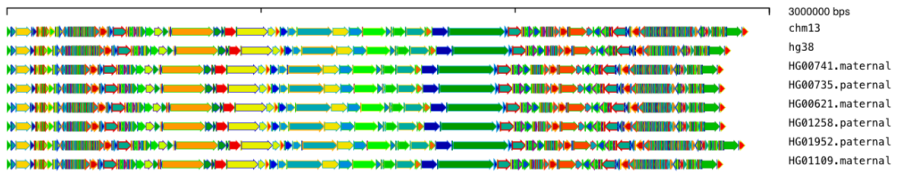
(c)

```
pgr-pbundle-decomp -w 48 -k 56 -r 12 --min-span 12 --bundle-length-cutoff 100
     --min-branch-size 8 pg_seqs.fa  WBS_pg_seqs. --min-cov 0
```



(d)

```
pgr-pbundle-decomp -w 48 -k 56 -r 6 --min-span 12 --bundle-length-cutoff 100
     --min-branch-size 8 pg_seqs.fa  WBS_pg_seqs. --min-cov 0
```



The Genome in the Bottle Consortium provides variant call benchmarks on seven benchmark genomes[35,60,61]. These benchmarks were initially formed by integrating multiple short-read technologies, but the latest version integrated linked-read and long-read technologies to form benchmarks in regions difficult to map with short reads. However, a set of 395 challenging but medically relevant genes were identified as substantially (>10%) excluded from the mapping-based benchmark due to long repeats, large structural variations, segmental duplications, and/or high polymorphism between the benchmark genome HG002 and the reference genome hg19 or GRCh38. A long-read genome assembly approach provided a reliable benchmark call set of 273 out of the 395 genes [35]. The remaining 122 were still excluded mainly because they were not accurately assembled or no benchmarking tools exist to compare different representations of complex variants in the genes. Overall 395 genes have recently shown to include high levels of polymorphism across different ethnicities making it highly challenging to represent their variations[62].

The current GIAB variant benchmarks focus on a small set of seven well-characterized genomes with extensive short-, linked-and long-read data to ensure robust benchmarks and more tractable method development and benchmark evaluations. Meanwhile, limited representation of genomes in a population may miss significant structural variants, additional copies of genes, and context for important variants for diseases not observed in a smaller dataset. Given that increasingly accurate long-read and assembly level data are being produced at pangenome scales now, we are surveying how we can utilize such resources to benchmark variant accuracy at a broader population level for the challenging clinical and medically important genes. Such pangenome analysis will help to generate guidelines for future practice.

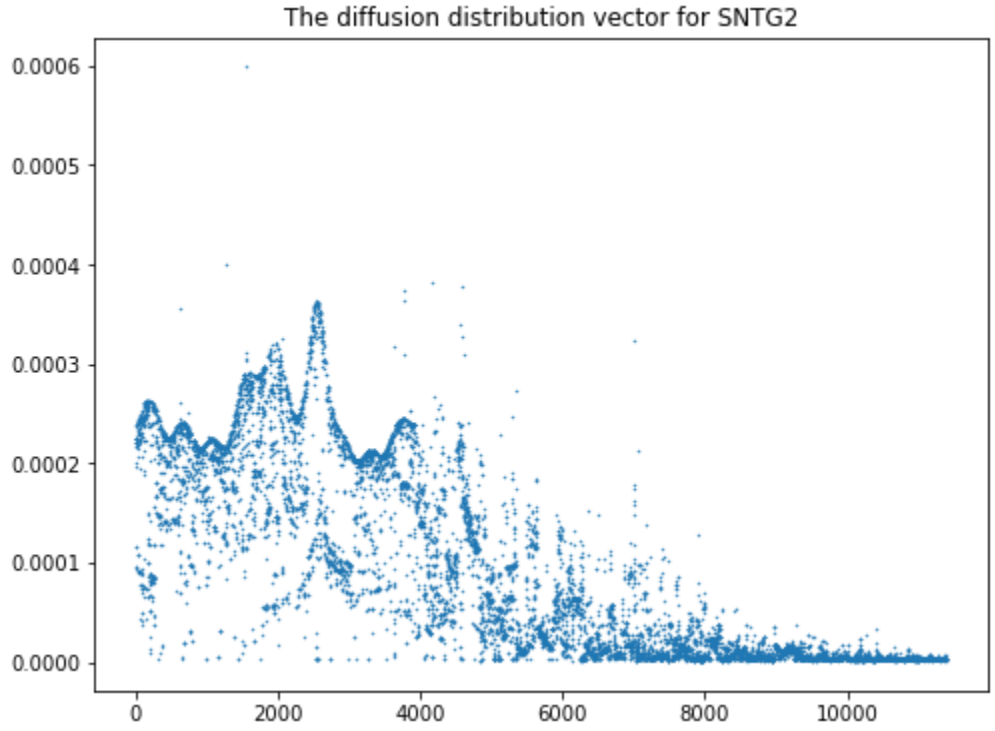
With PGR-TK, we extract all sequences from the HPRC year one release (94 haplotype assemblies), and CHM13 v1.1, GRCh38 and hg19 of all 385 CMRG. We generate a MAP-graph of each gene and output in GFA format. For each graph, we derived two metrics to estimate (1) the degree of polymorphism among the pangenomes, and (2) the repeat content taking account of the variations of the pangenomes.

To estimate the degree of polymorphism, we consider a diffusion process in the graph (see below) and derive an entropy-like quantity from a normalized equilibrium distribution.. The higher entropy values indicate more complicated graphs likely from the polymorphism from different genomes. The diffusion weight on each vertex is also associated with multiplicity and repetitiveness of the corresponding segments in the pangenomes. We pick the average of the top 32 diffusion weights from the MAP-graph of each gene as a simple metric to measure the most challenging repeat content within a gene.

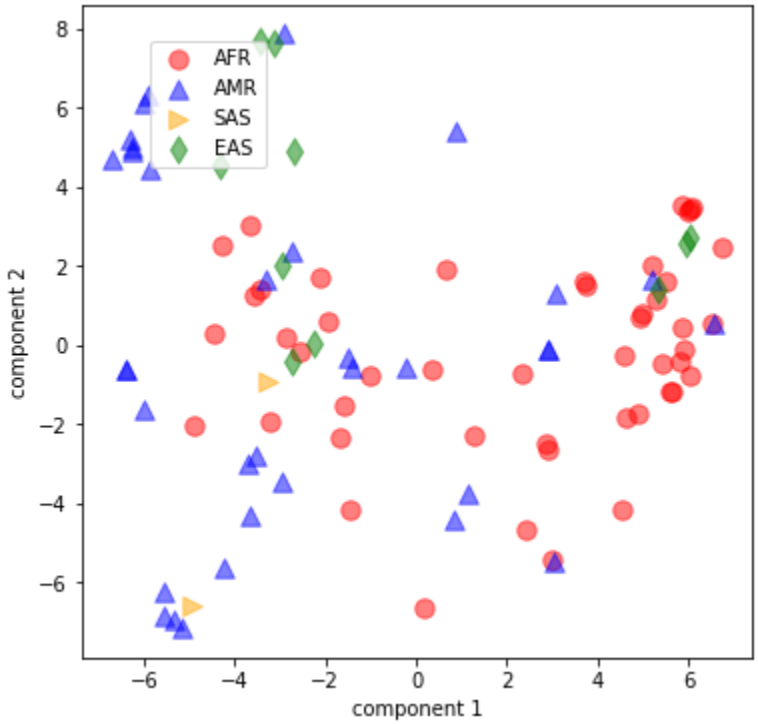
To gain insight about the challenge for calling variants of the CMRG set at a pangenome scale, we plot the diffusion entropy versus the maximum local repeat weights for each gene (**Figure 4c**). As there are no obvious correlations, these two quantities provide independent measurements of two aspects of the MAP-graph structures of these genes. We find, unsurprisingly, high repetitive genes are harder to create a reliable variant benchmark call set. Many highly repetitive genes are excluded from the current CMRG benchmark set. We do not observe that higher entropy is correlated with the reduction of the gene in the benchmark set. We find the high entropy genes also span larger regions in the genome. While the entropy can indicate the complexity of variations in the population, we observe different clustering

structures of the top entropy genes. (Please see the comparison of the MAP-graph PCA plots of SNTG2 and KMT2C in **Supplementary Figure 4**.)

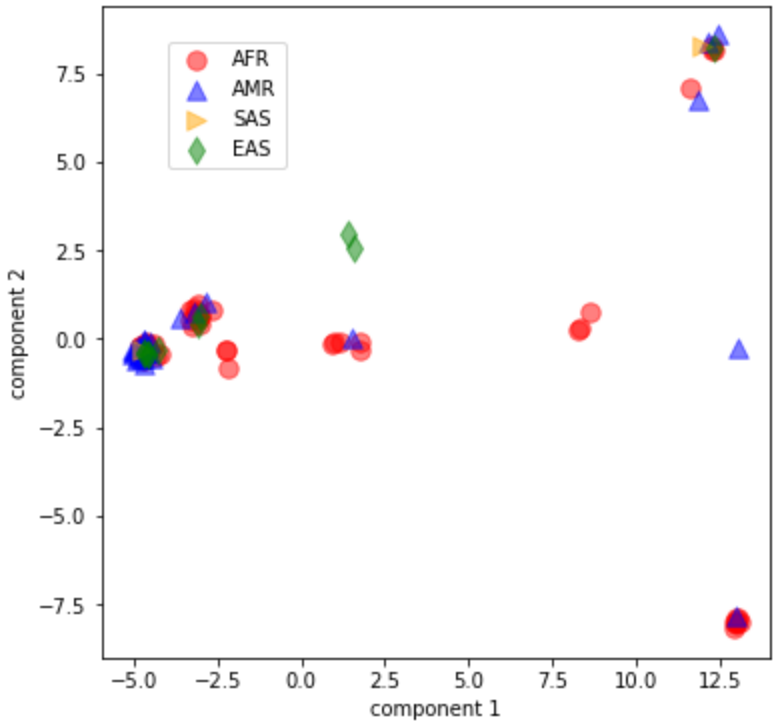
Supplementary Figure 4a



Supplementary Figure 4b: PCA plot for SNTG2 (Highest Entropy in the CMRG gene set)



Supplementary Figure 4c: PCA plot for KMT2C (Highest Entropy in the CMRG gene set)

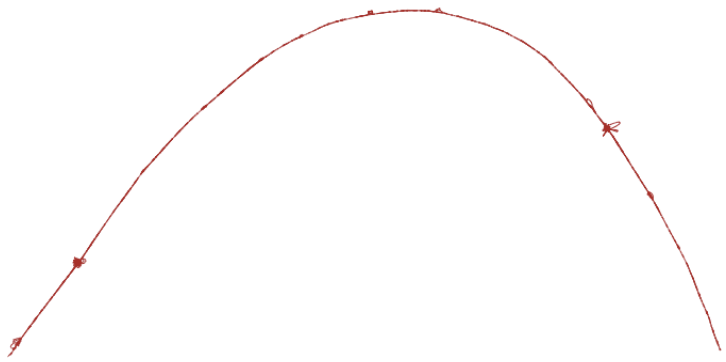

AFR
AMR
SAS
EAS
component 2
component 1
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
12.5

We highlight several genes with high entropy or high repetitiveness. The MAP-graphs and the IGV view of the pangenome assemblies of a selected set genes (LMF1, ANKRD11, SRGAP2, KMT2C, LPA, MUC4, MUC3A, KATNAL2, FLG) aligned to GRCh38 are shown in **Supplementary Figure 5.** In the IGV view of LMF1 (**Supplementary Figure 5a**), a number of variation hotspots are visible and may correspond to localized structural variants. This serves as a simple example to examine the concordance of structural variation in the population and the principal bundle decomposition. We use PAV[63] to call structural variants for comparison. In **Supplementary Figure 5b**, the structural variant calls resulting from PAV are provided as an auxiliary track (black) below the principal bundle decomposition tracks for a selected set of HPRC year one genomes, where both haplotypes are resolved in the region. It is evident that the SV calls correspond to the regions where principal bundles have complex structure and are distinct from the reference genome GRCh38 (the top track).

As shown in LPA (associated with Coronary Disease[64], **Supplementary Figure 7)**, KATNAL2 (loss of function variant discovered in autistic proband[65,66]) also has long tandem repeat variations in the HPRC pangenome cohort. We find the number of the 5.8kbp repeats inside KATNAL2 ranges from 3 to 25 (**Supplementary Figure 8**). Applying MAP-graph decomposition on genes such as KATNAL2 with a big pangenome reference panel will provide additional insights to the natural of the variability of the repetitiveness and its impacts to the underlying biology like other more well studied gene, e.g. LPA, in the coming years.

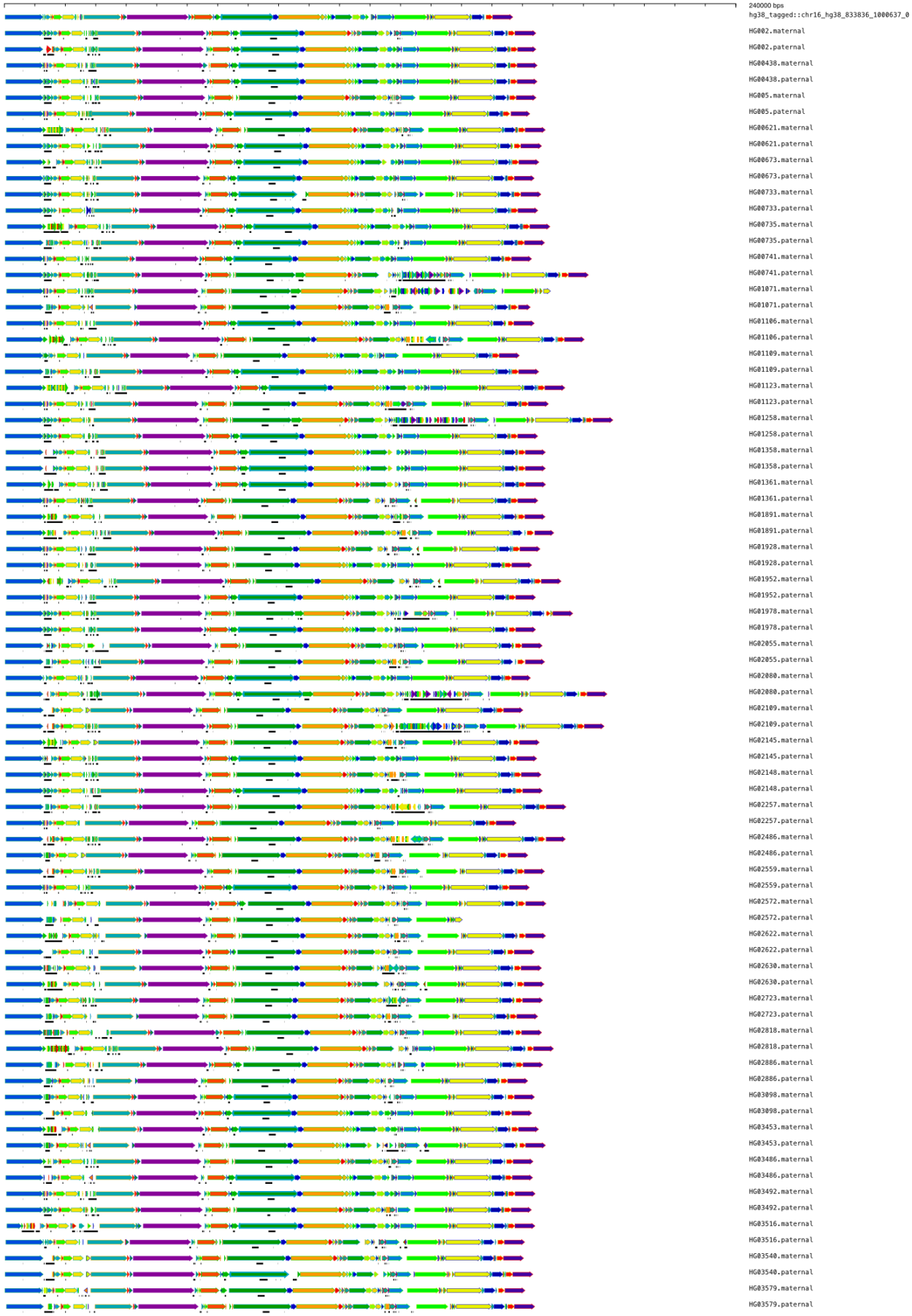
Supplementary Figure 5

GIAB CMRG cases:
(a) LMF1

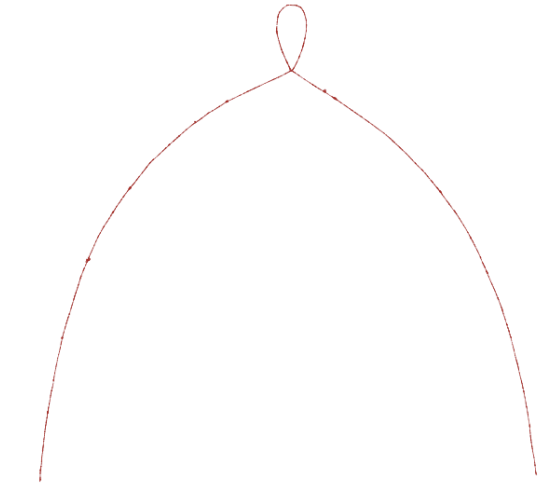


(b) Comparing the PAV structural variant calls indicated by the black auxliary tracks to the principal bundle decomposition illustrates how the SV calls correspond to the changes in the principal bundles between each individual genome and the reference used for the SV call

(GRCh38, the top track).

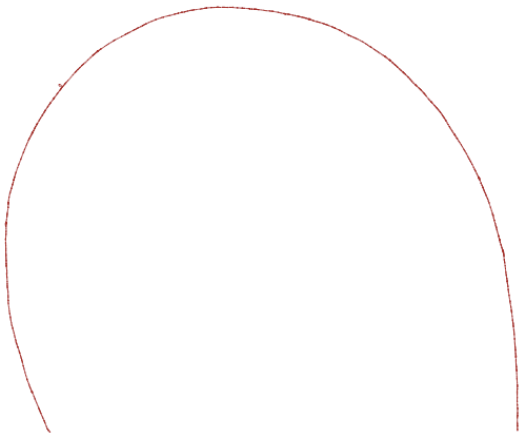


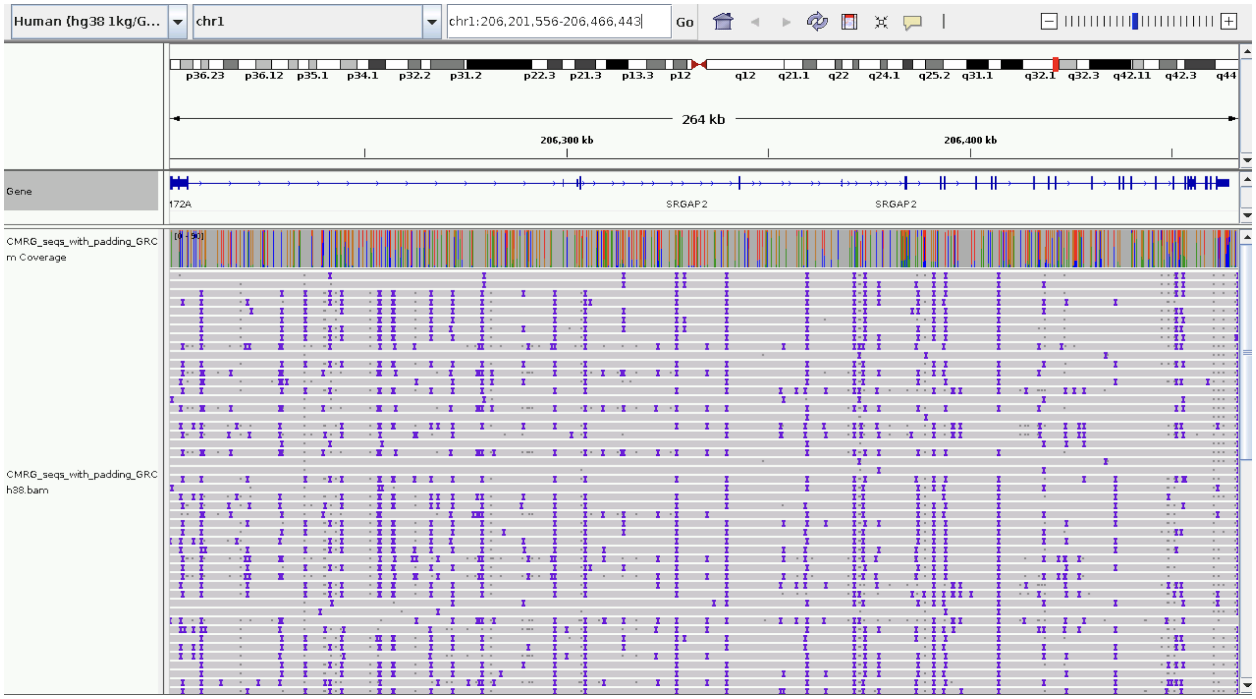
(b) ANKRD11



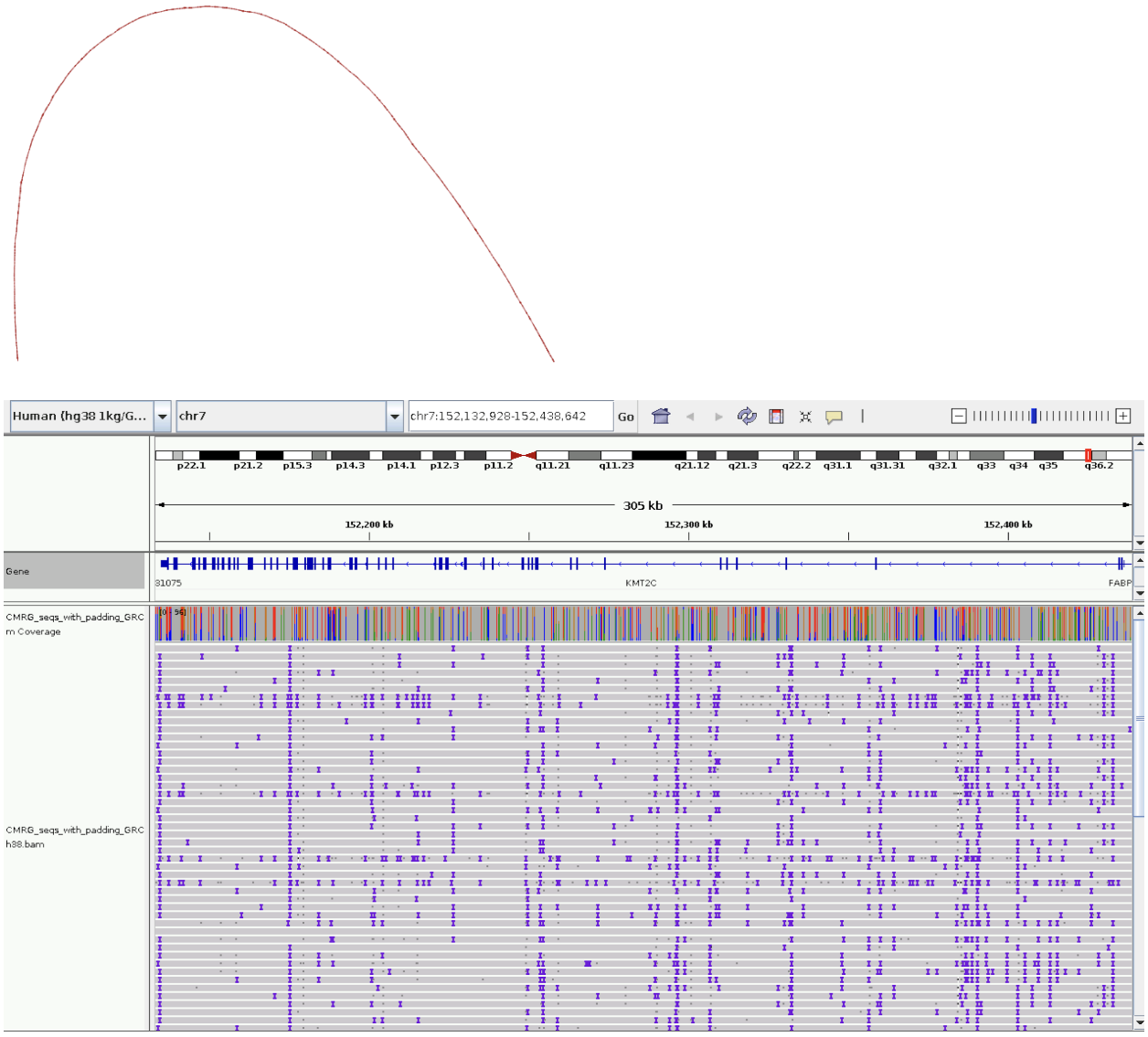

Human (hg38 1kg/G...
chr16
chr16:89,265,625-89,492,556
Go
p13.3
p13.2
p13.12
p12.3
p12.2
p12.1
p11.2
p11.1
q11.2
q12.1
q12.2
q21
q22.1
q22.3
q23.2
q24.1
226 kb
89,280 kb
89,300 kb
89,320 kb
89,340 kb
89,360 kb
89,380 kb
89,400 kb
89,420 kb
89,440 kb
89,460 kb
89,480 kb
Gene
ANKRD11
ANKRD11
ANKRD11
NR_110982.1
CMRG_seqs_with_padding_GRCm Coverage
CMRG_seqs_with_padding_GRCh38.bam

(c) SRGAP2

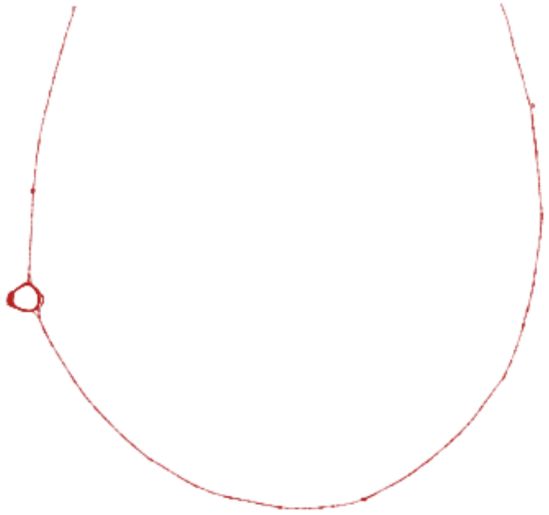



Human (hg38 1kg/G...
chr1
chr1:206,201,556-206,466,443
Go
p36.23 p36.12 p35.1 p34.1 p32.2 p31.2 p22.3 p21.3 p13.3 p12 q12 q21.1 q22 q24.1 q25.2 q31.1 q32.1 q32.3 q42.11 q42.3 q44
264 kb
206,300 kb
206,400 kb
Gene
172A
SRGAP2
SRGAP2
CMRG_seqs_with_padding_GRCm Coverage
[0 - 50]
CMRG_seqs_with_padding_GRCh38.bam

(d) KMT2C

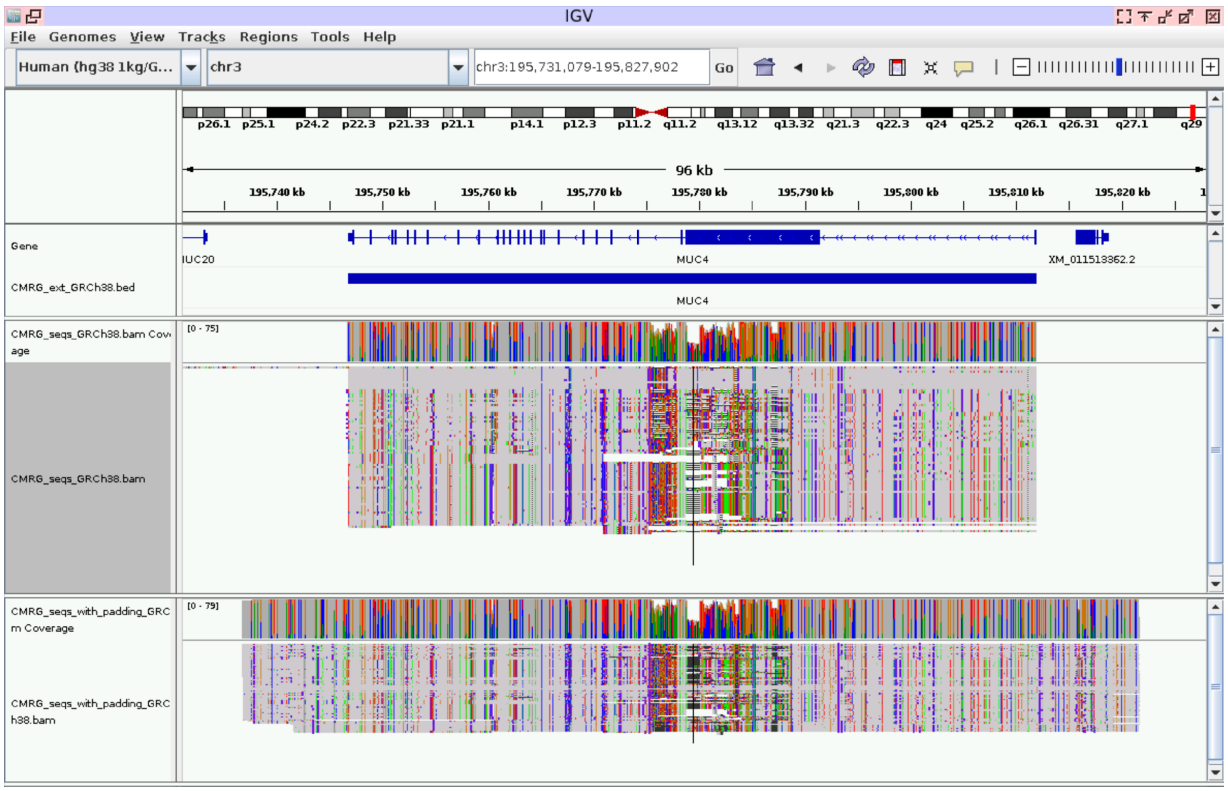

Human (hg38 1kg/G...
chr7
chr7:152,132,928-152,438,642
Go
p22.1 p21.2 p15.3 p14.3 p14.1 p12.3 p11.2 q11.21 q11.23 q21.12 q21.3 q22.2 q31.1 q31.31 q32.1 q33 q34 q35 q36.2
305 kb
152,200 kb
152,300 kb
152,400 kb
Gene
31075
KMT2C
FABP
CMRG_seqs_with_padding_GRCm Coverage
0 - 50
CMRG_seqs_with_padding_GRCh38.bam

(e) LPA

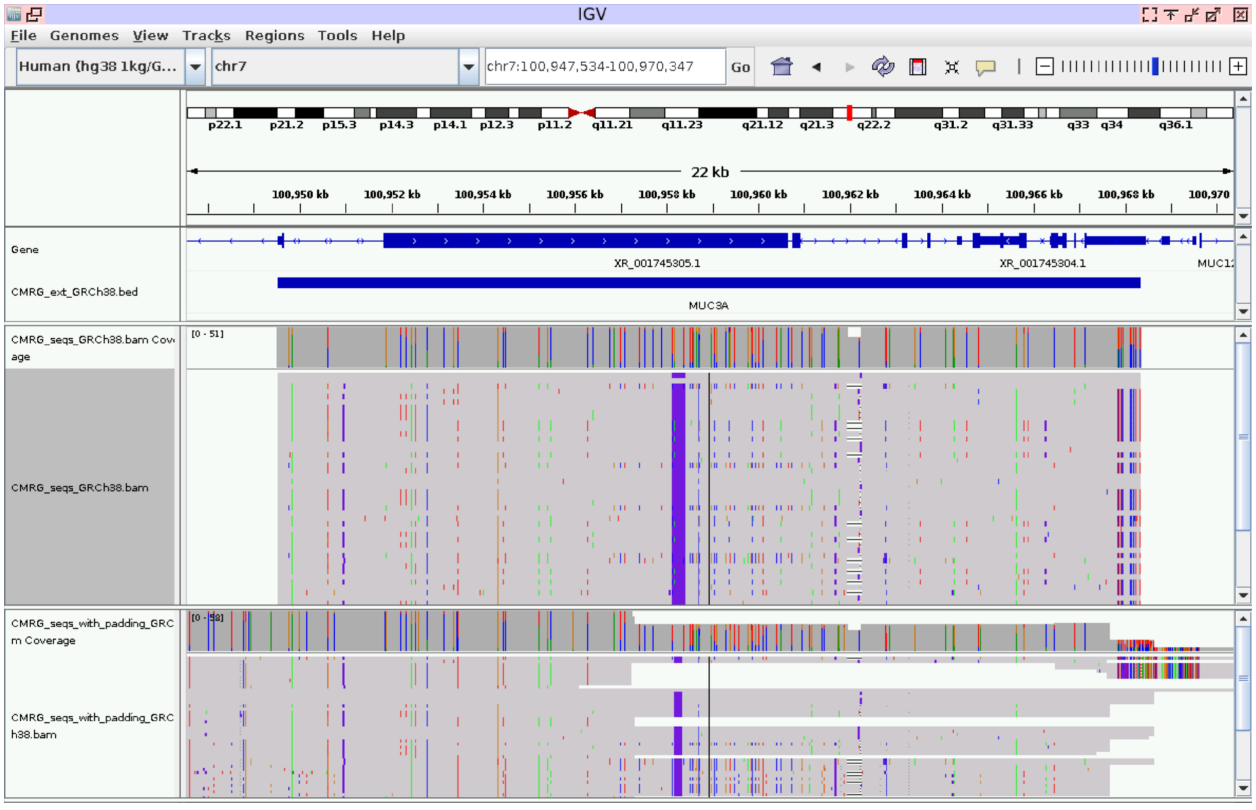



IGV
File Genomes View Tracks Regions Tools Help
Human (hg38 1kg/G...
chr6
chr6:160,529,482-160,666,275
Go
p25.1 p23 p22.2 p21.31 p21.1 p12.2 q11.1 q13 q14.1 q15 q16.2 q21 q22.1 q22.32 q23.3 q24.3 q25.3 q27
136 kb
160,540 kb 160,560 kb 160,580 kb 160,600 kb 160,620 kb 160,640 kb 160,660 kb
Gene
LPA
CMRG_ext_GRCh38.bed
LPA
CMRG_seqs_GRCh38.bam Coverage
[0 - 351]
CMRG_seqs_GRCh38.bam
CMRG_seqs_with_padding_GRCm Coverage
[0 - 358]
CMRG_seqs_with_padding_GRCh38.bam

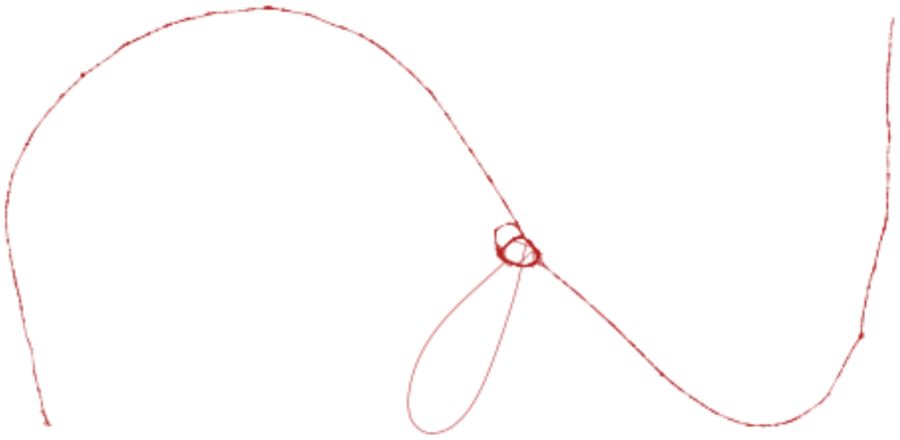
(f) MUC4

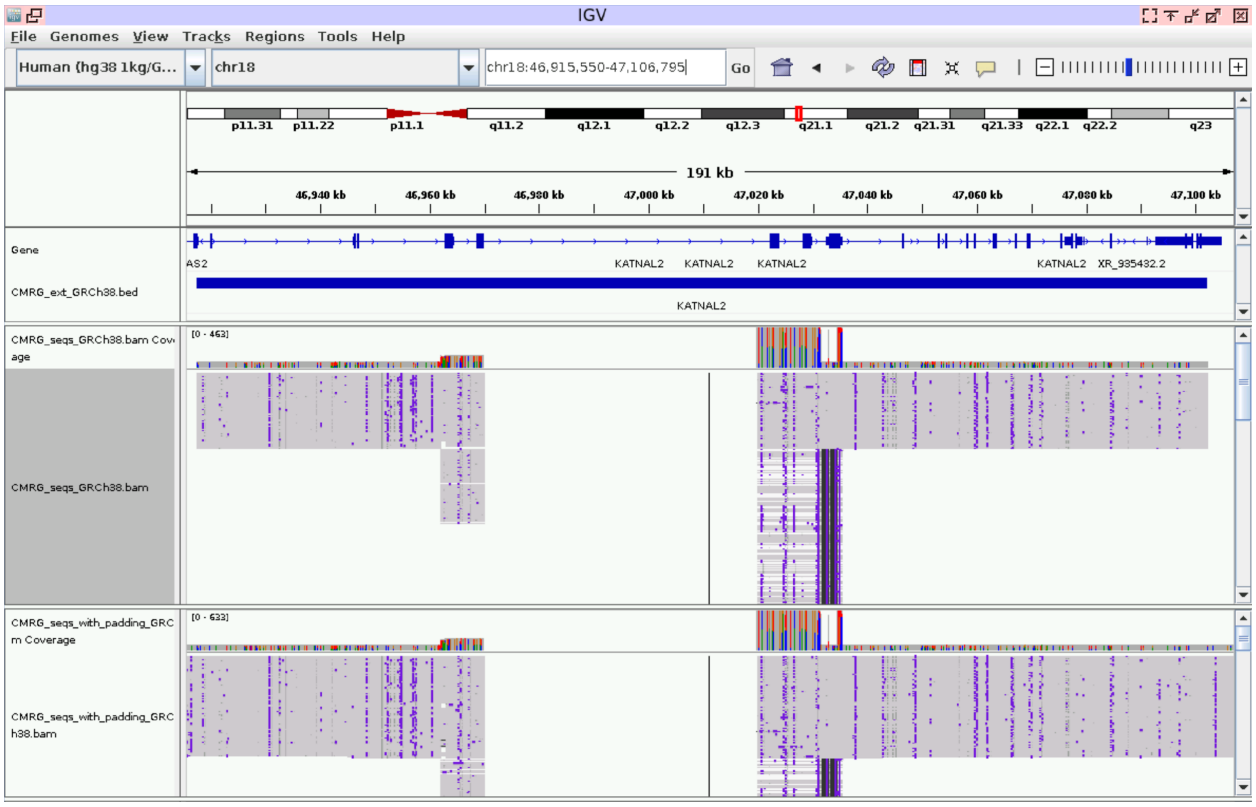

IGV
File Genomes View Tracks Regions Tools Help
Human (hg38 1kg/G...
chr3
chr3:195,731,079-195,827,902
Go
p26.1 p25.1 p24.2 p22.3 p21.33 p21.1 p14.1 p12.3 p11.2 q11.2 q13.12 q13.32 q21.3 q22.3 q24 q25.2 q26.1 q26.31 q27.1 q29
96 kb
195,740 kb 195,750 kb 195,760 kb 195,770 kb 195,780 kb 195,790 kb 195,800 kb 195,810 kb 195,820 kb
Gene
IUC20
MUC4
XM_011513362.2
CMRG_ext_GRCh38.bed
MUC4
CMRG_seqs_GRCh38.bam Coverage
[0 - 75]
CMRG_seqs_GRCh38.bam
CMRG_seqs_with_padding_GRCh Coverage
[0 - 79]
CMRG_seqs_with_padding_GRCh38.bam

(g) MUC3A


IGV
File Genomes View Tracks Regions Tools Help
Human (hg38 1kg/G...
chr7
chr7:100,947,534-100,970,347
Go
p22.1 p21.2 p15.3 p14.3 p14.1 p12.3 p11.2 q11.21 q11.23 q21.12 q21.3 q22.2 q31.2 q31.33 q33 q34 q36.1
22 kb
100,950 kb 100,952 kb 100,954 kb 100,956 kb 100,958 kb 100,960 kb 100,962 kb 100,964 kb 100,966 kb 100,968 kb 100,970
Gene
XR_001745805.1
XR_001745804.1
CMRG_ext_GRCh38.bed
MUC3A
CMRG_seqs_GRCh38.bam Coverage
[0 - 51]
CMRG_seqs_GRCh38.bam
CMRG_seqs_with_padding_GRCh Coverage
CMRG_seqs_with_padding_GRCh38.bam

(h) KATNAL2

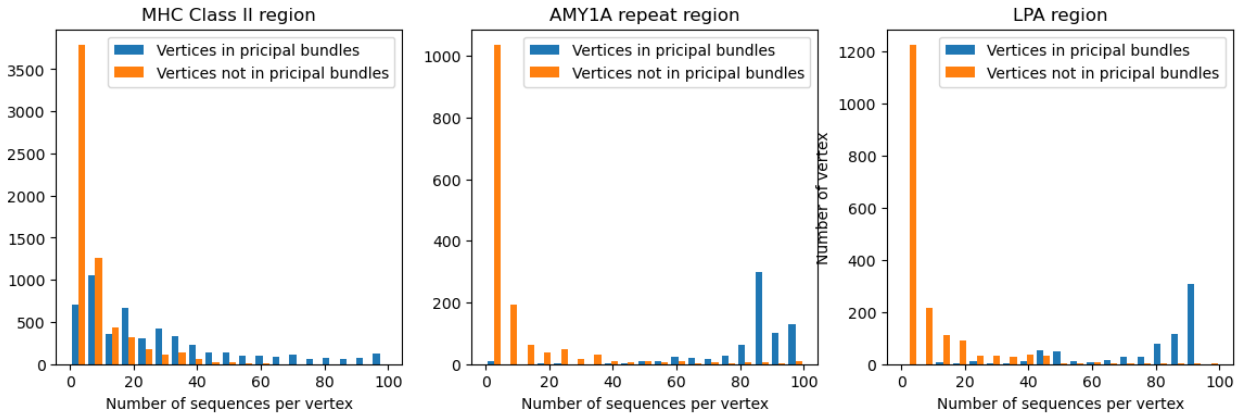



IGV
File Genomes View Tracks Regions Tools Help
Human (hg38 1kg/G...
chr18
chr18:46,915,550-47,106,795
Go
p11.31 p11.22 p11.1 q11.2 q12.1 q12.2 q12.3 q21.1 q21.2 q21.31 q21.33 q22.1 q22.2 q23
191 kb
46,940 kb 46,960 kb 46,980 kb 47,000 kb 47,020 kb 47,040 kb 47,060 kb 47,080 kb 47,100 kb
Gene
AS2 KATNAL2 KATNAL2 KATNAL2 KATNAL2 XR_935432.2
CMRG_ext_GRCh38.bed
KATNAL2
CMRG_seqs_GRCh38.bam Coverage
[0 - 462]
CMRG_seqs_GRCh38.bam
CMRG_seqs_with_padding_GRCm Coverage
[0 - 633]
CMRG_seqs_with_padding_GRCh38.bam

(i) FLG

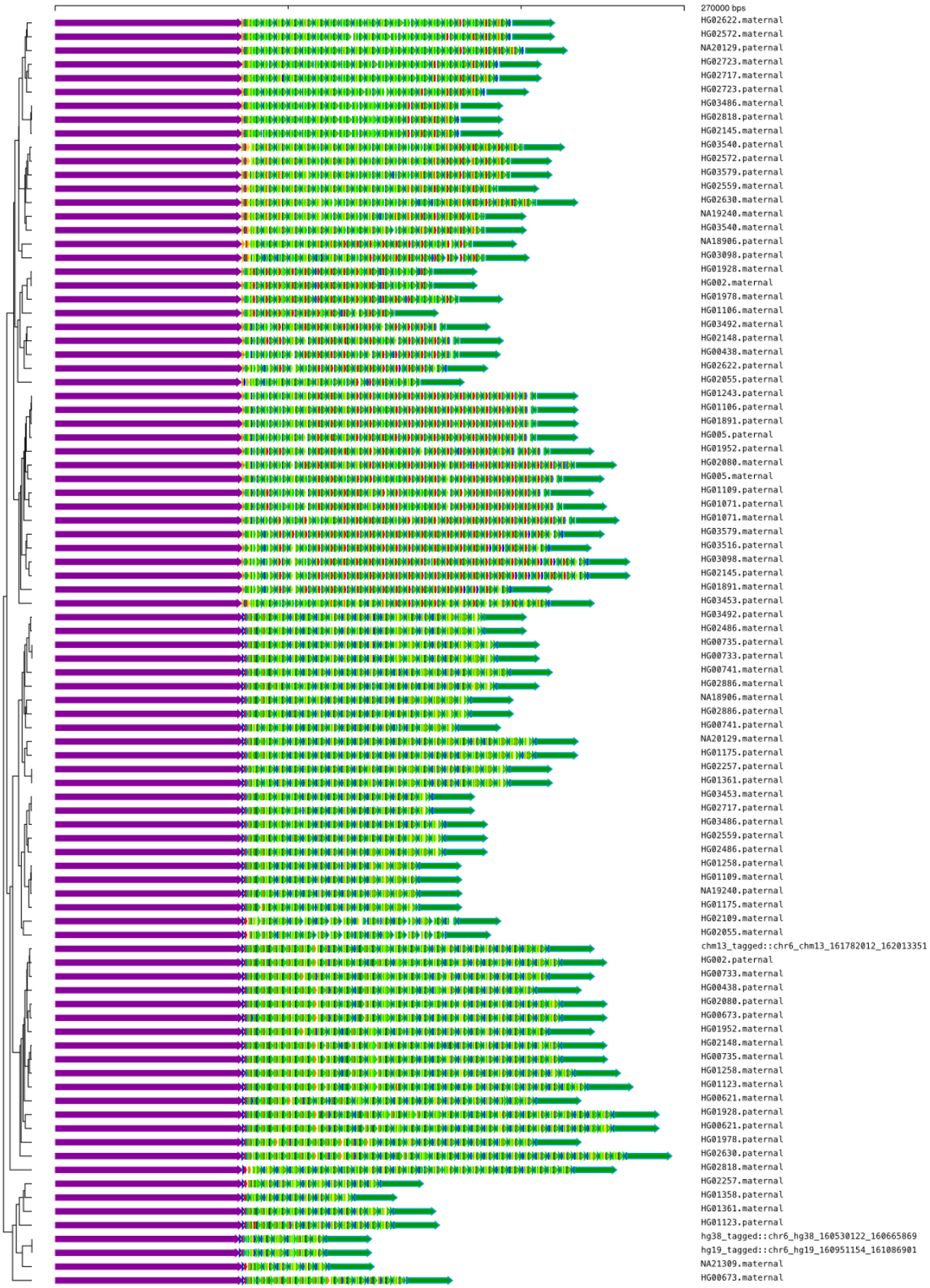

IGV
File Genomes View Tracks Regions Tools Help
Human (hg38 1kg/G...
chr1
chr1:152,300,165-152,327,239
Go
p36.23 p36.12 p34.3 p33 p32.1 p31.1 p22.2 p21.2 p13.2 q11 q12 q21.1 q22 q24.1 q25.2 q31.2 q32.2 q41 q42.2 q44
27 kb
152,310 kb
152,320 kb
Gene
CMRG_ext_GRCh38.bed
FLG
FLG-AS1
FLG
CMRG_seqs_GRCh38.bam Coverage
[0 - 97]
CMRG_seqs_GRCh38.bam
CMRG_seqs_with_padding_GRCh Coverage
[0 - 152]
CMRG_seqs_with_padding_GRCh38.bam

Supplementary Figure 6: The distribution of the vertex weight on the principal bundle vertices and non-principal bundle vertices for the three cases MHC class II, AMY1A and LPA regions.



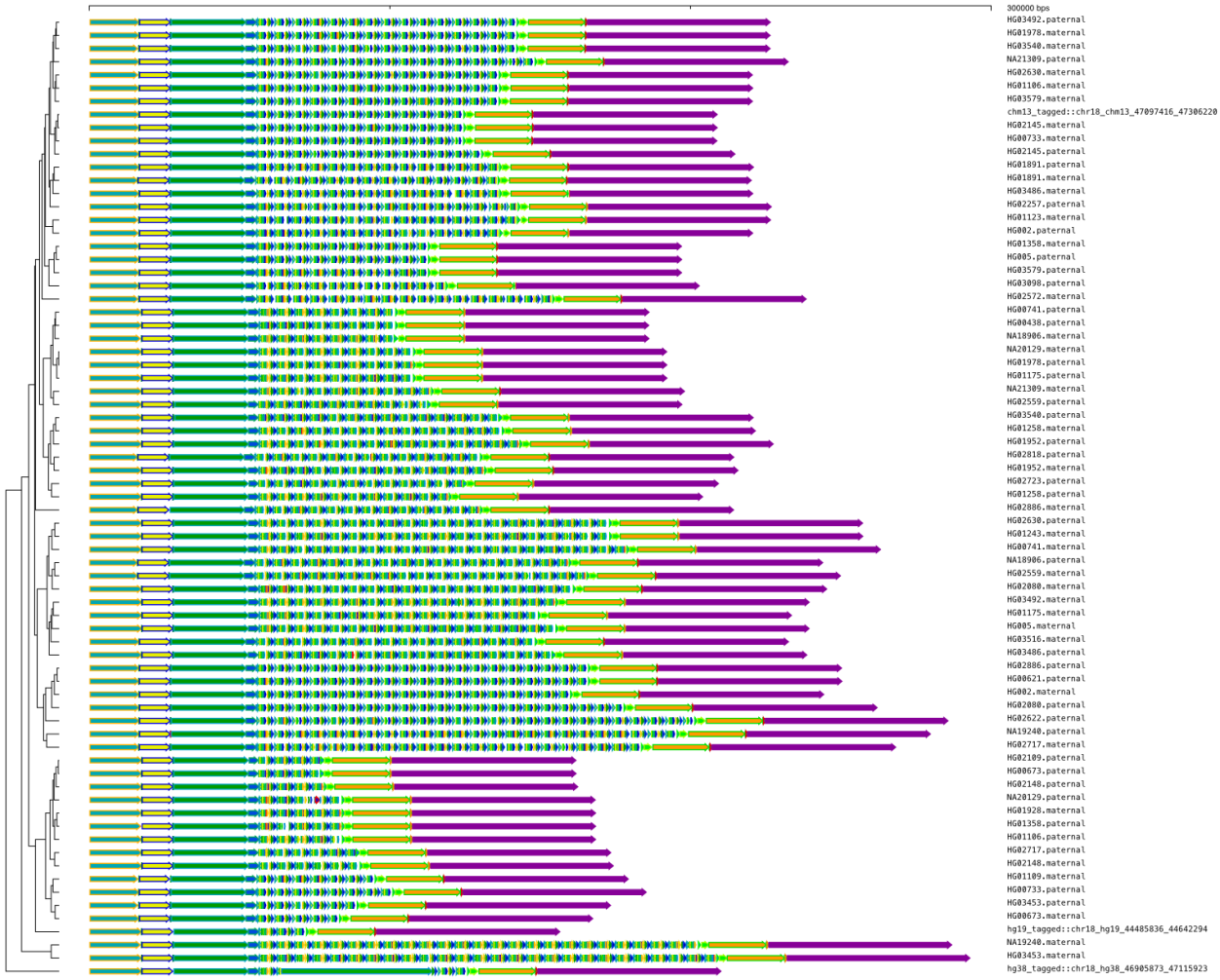
Supplementary Figure 7

LPA, KIV-II repeats principal bundle decomposition plot



Supplementary Figure 8

Principal bundle plot for KATNAL2: GRCh38 chr18:46905550-47116795 showing different numbers of the repeat.



Compute Graph Diffusion Entropy and Max Repetitive Weight

It would be desirable to derive quantitative measurements so we can characterize a set of large numbers of MAP-graphs fast. One thing we are interested in quantifying is "how complex a graph is". The intuition is that if a region of the genome is more polymorphic in the population, the graph will have more alternative paths, or bubbles. We like to generate a quantity as a proxy for that. For this, we borrow the idea from network science study and spectral graph theory to consider a diffusion/random walk process on a graph[67]. For a graph, we consider a set of random walkers starting at each vertex. The random walkers can drift on the graph though the edge-connection. We can consider the distribution of the random walkers in the final equilibrium state. If a graph is relatively simple, then the final distribution will be uniform (subject to minor boundary condition corrections.) On the other hand, if the weights of the vertices or topology of the graph are more complex, we would expect the final distribution of the walkers would be less uniform and reflect the complicated nature of the graph.

The final distribution of the such diffusion process can be obtained by simple matrix multiplication iteration from the adjacent matrix of a MAP-graph. Given an adjacency matrix **A**,

where the matrix element $\mathbf{A}_{ij}$ = number of sequence supports edge from v_i to v_j. The final distribution **P** can be written as

$\mathbf{P} = (1/N) \lim_{n->\inf} \mathbf{M}^n \mathbf{1}$, where $\mathbf{M} = \mathbf{A}\mathbf{D}^{-1}$, D is the degree matrix defined as $\mathbf{D}_{ii}$ = degree of vertex v_i and $\mathbf{D}_{ij}$ =0 if $i \neq j$, N is the total number of vertices, and **1** is a column vector in which every element is one. (When we compute **P**, we only repeat the number of multiplications N times to approximate the final distribution.)

P is a normalized column vector $[p_0, p_1,...,p_{n-1}]^T$ such that $\sum_{i=0..n-1} p_i = 1$. (See **Supplementary figure 4a** for an example.) The diffusion entropy used in this work is defined as $S = - \sum_{i=0..n-1} p_i \log_2(p_i)$.

To find the highly repetitive elements inside a region of interest, we look into largest elements in the unnormalized vector N**P** as a proxy of average number of repeats considering the graph structure. We pick the top 32 elements in N**P** and use the average of those as a proxy number to estimate the repetitiveness of potential repeat units inside a region of interest.

Data files:

The HPRC year one release sequence and pre-built index:
https://giab-data.s3.amazonaws.com/PGR-TK-Files/pgr-tk-HGRP-y1-evaluation-set-v0.tar

Scripts and source data URLs for constructing the HPRC AGC file:
https://github.com/GeneDX/pgr-tk-notebooks/tree/main/pgr-tk-sequence-source

All GFA files, fetched sequences from HPRC year one release of the 385 CMRG:
https://giab-data.s3.amazonaws.com/PGR-TK-Files/CMRG_output_dir_v0.3.3.tar

Example Notebooks using PGR-TK including code making most of (the source of) the plots in this manuscript: https://github.com/GeneDX/pgr-tk-notebooks

Information about a docker image with pre-built PGR-TK library and Jupyter Lab Server and Usage: https://github.com/GeneDX/pgr-tk/blob/main/pgr-tk-workstation/Readme.md

Code:

source: https://github.com/GeneDX/pgr-tk

- API document: https://genedx.github.io/pgr-tk/

pre-built binaries: https://github.com/GeneDX/pgr-tk/releases/tag/v0.4.0