**APPROACH**

**Aim 1: Develop a scalable panMAN-based sample placement per read and consensus calling tool (panMAMA)**

Introduction and existing work: Our lab has developed a panMAN-based sample placement tool, panmap, for isolated samples containing reads from a single strain. However, placing mixed samples and calling their consensus requires identifying the haplotypes present and estimating their relative abundance. Deconvoluting strain-level mixed samples is becoming increasingly challenging as reference databases grow exponentially. To address this issue, existing methods have been developed to avoid computationally expensive full-read alignment, but they have limitations. Pseudo-alignment approaches estimate alignment scores, which offers short-term solutions but still requires individual index files and alignments for each reference, and thus scales linearly with the number of references. On the other hand, "alignment-update" approaches, which have been developed for well-characterized species like SARS-CoV-2, fully align reads to the ancestral reference and update alignment scores for other strains based on known Single Nucleotide Polymorphisms (SNPs). While this achieves sub-linear scalability, it requires a well-established ancestral reference and only utilizes SNP information. To overcome these limitations, I propose panMAMA, a tool that achieves sub-linear scalability while utilizing all types of genomic variations. This novel approach will enable more efficient and comprehensive analysis of rapidly growing reference databases and mixed strain samples.

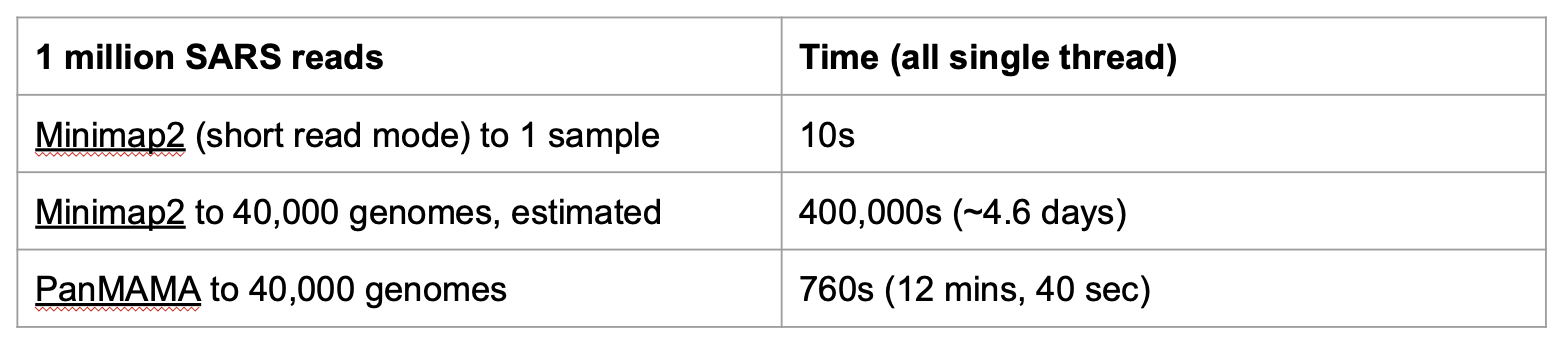
Proposed approach:

**1.1 Fast, scalable haplotype abundance estimation in a mixed metagenomic sample.**

PanMAMA essentially combines the merits of pseudo-alignment and “alignment-update” described in the previous section, by utilizing a k-min-mer based pseudo-chaining algorithm and a seed-annotated tree index previously developed in the lab as part of *panmap*.

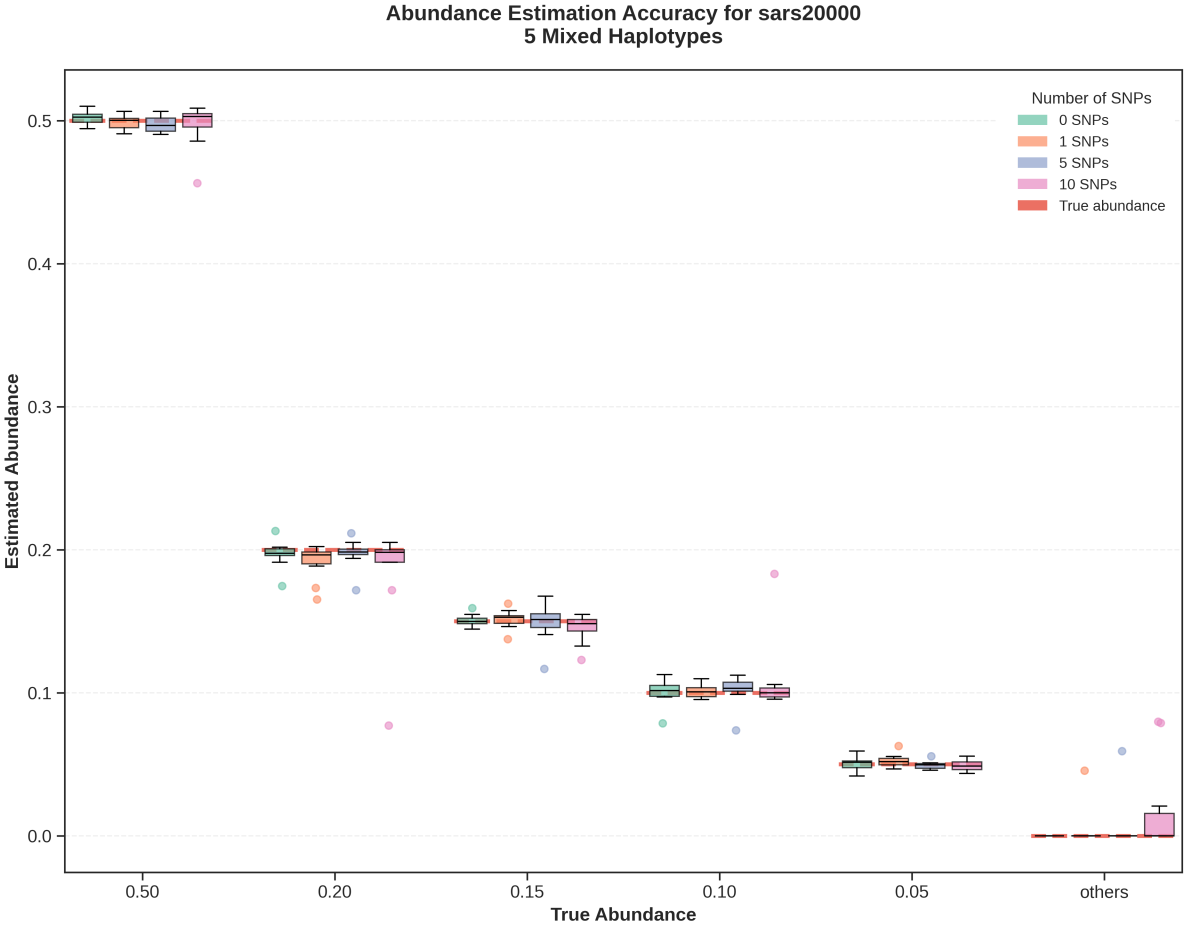
**k-min-mer based pseudo-chaining algorithm.** A k-min-mer can be understood as l-k-mers, or l consecutive k-mers concatenated and used as a single seed. In practice, submers, such as minimizers or syncmers, are used instead of k-mers. K-min-mer based pseudo-chaining generates chains in linear time between a query and a reference by identifying "mini-chains" of exact, consecutive, and non-duplicate matches between the query and reference k-min-mer sets, starting from the 1st k-min-mer of the query. A mini-chain terminates when the current query k-min-mer doesn't exist on the reference or the matching k-min-mer on the reference is not unique, and a new match starts on the next k-min-mer on the query. The mini-chains are then joined to a single chain by identifying the mini-chain with the highest number of matches and all other mini-chains collinear to the highest-scoring mini-chain. Originally designed for long-read chaining, we show that this method also works for short reads and is a good estimator of alignment scores.

**Apply seed-annotated tree index to k-min-mers.** The set of syncmers, a subset of k-mers selected using a window-independent scheme, is pre-computed for each node on a panMAN and stored in a seed-annotated tree index. The index annotates the edges with positions on a parent node haplotype where seeds are added, substituted, or deleted, to give rise to the seed set on its child node. The k-min-mer set for the root node is first computed, and the syncmers and k-min-mers for all descendent nodes are updated using the seed-annotated tree index.

**Update pseudo-chaining scores using seed-annotated tree index.** K-min-mers are hashed using the same hashing scheme for both references and query reads. A modified nt-hash algorithm is implemented to hash both syncmers and k-min-mers in linear time. During hashing, a hash-to-read index is constructed, where each hashed k-min-mer points to a list of reads and indices on the read k-min-mer set that store the respective hash. Initially, all reads are pseudo-chained to the root node of the tree. At each descendant node during a tree traversal, the reference k-min-mer set is updated and changes are recorded. Based on the k-min-mer changes, the hash-to-read index quickly identifies the reads and exact indices on their k-min-mer sets affected by the update. Mini-chains can then be locally split or merged, and chaining scores dynamically updated for the current node. The chaining scores are recorded for each node.

**Figure 1.** *Comparison of runtime to score 1 million short reads between Minimap2 and panMAMA. Minimap2's runtime for mapping against 40,000 genomes is extrapolated from its runtime for mapping against 1 genome.*

**Estimate abundance using an Expectation Maximization algorithm.**

After a pre-order traversal of the tree, a matrix of pseudo-chaining scores for all reads against all nodes is obtained. The score matrix is used to estimate a probability matrix representing the probability of each read being generated from each haplotype. This probability matrix and a uniformly initialized relative abundance vector are then fed into an Expectation Maximization (EM) algorithm. The final abundance vector undergoes a specified number of rounds of filtering, where nodes with abundance lower than a specified value are removed and a new uniformly distributed abundance vector is created for the remaining nodes, followed by a separate EM run. As EM scales poorly with the number of nodes, a pre-EM filter round is performed to select probable nodes. Overlap-coefficients are computed for the reads and each node using their k-min-mer sets, and nodes with 100% overlap and top N (user-specified) nodes are selected to proceed to the EM step. Additionally, a turboEM is implemented internally to accelerate convergence.

**Figure 1.** *Abundance estimation accuracy for mixed samples of 5 SARS haplotypes with abundance 0.5, 0.2, 0.15, 0.1, 0.05. Red dotted lines show the true abundance of haplotypes, and whisker plots show the distribution of the estimated abundance across 10 replicates. Replicates are simulated at 1000x depth.*

**1.2 Hybrid heuristic and maximum likelihood read assignment and consensus calling.**

For a read i, we can extract its pseudo-chaining scores against each reference in the final list of haplotypes estimated to be present, denoted Si. Denote sij as the pseudo-chaining score of read i against haplotype j. For each haplotype j, if sij >= max(Si) - t, where t is a tolerance score (default to 5), we assign read i to haplotype j. We then map the assigned reads to each of their respective haplotype's sequence using Minimap2. Denote reads assigned to haplotype j as Rj, we split Rj into groups by other haplotypes to which they are also assigned (e.g. reads assigned to haplotype A only are grouped together, reads assigned to haplotypes A and B are grouped together, reads assigned to haplotypes A and C are grouped together, etc.) Read groups that are assigned to only one haplotype are referred to as "allele-determining group." Positions on a haplotype with allele-determining group are assigned alleles based on the majority allele supported by the allele-determining group. For positions on a haplotype without "allele-determining group," allele assignment depends on whether a cryptic mutation is observed. If there is no cryptic mutation at a position, under the assumption there is no back mutation, we assign the reference allele to each haplotype. If there is cryptic mutation at a position, using the estimated abundance and assigned alleles for other haplotypes, a joint likelihood is calculated from the likelihoods of each read group, and the allele with the maximum likelihood is assigned.

**1.3 Evaluate abundance estimation and variant calling accuracy.**

The performance of panMAMA will be evaluated using simulated datasets. Simulated mixed samples will be simulated by randomly selecting multiple nodes on a panMAN and generating mixed samples of simulated reads. Simulated samples will vary by the number of mixed haplotypes, the true proportion of each haplotype, the number of mutations from the true reference, read length, and depth. I will specifically evaluate how well panMAMA estimates the presence of true haplotypes and their relative abundance. For samples simulated with mutations, I will also evaluate how close the consensus sequences are to the real simulated variant haplotypes.

Preliminary results: I have developed panMAMA and am finalizing the optimization in its runtime and accuracy. PanMAMA provides significant speedup compared to full-read alignment (Fig.1) while also achieving high accuracy in estimating relative abundances in mixed samples (Fig. 2).

Research challenges, limitations, and alternative approaches: PanMAMA has several limitations. As we use k-min-mers to estimate alignment score, highly similar sequences with only a few nucleotide distances can be indistinguishable to panMAMA as their k-min-mer sets might be identical if their nucleotide differences do not make different syncmer sets. As seen in Figure 1., we get some decreased accuracy at estimating the abundance for a few haplotypes of the simulated mixed samples. While this is only observed for a few haplotypes out of all replicate samples and that we still correctly estimate the haplotypes that are present, there is a limit on the the number of mutations that panMAMA can demix haplotypes with relative accuracy. Additionally, consensus calling for highly similar genomes, such as SARS-CoV-2 genomes, can be limited as cryptic alleles can't be assigned to any haplotype with high confidence, particularly for samples with a high number of mixed haplotypes.

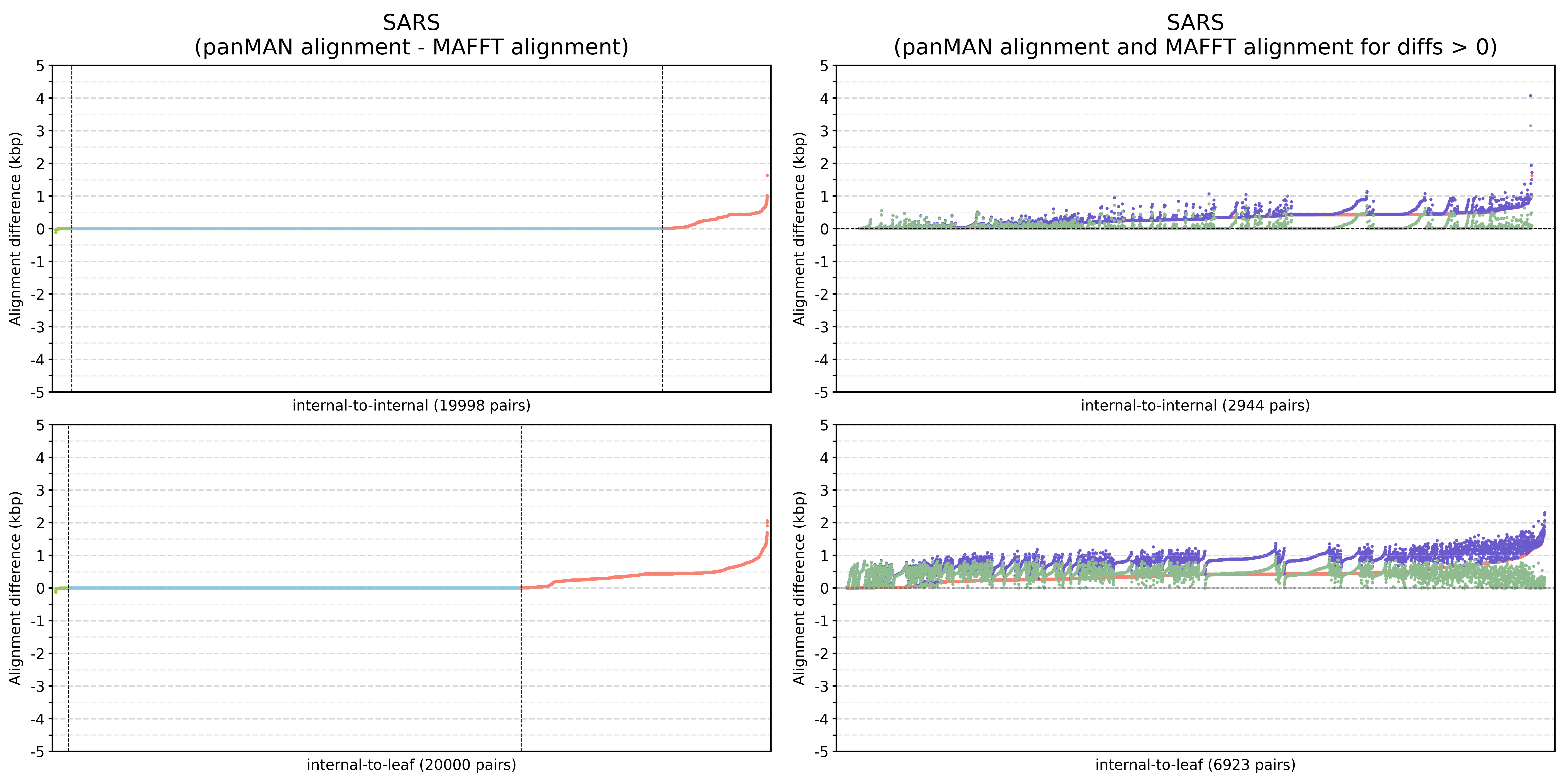
**Aim 2: Develop a tool to impute missing data on a panMAN using a phylogenetic based method and liftover genome annotations (panMANIA)**

Introduction and existing work: Direct bioinformatic analysis on compressed dataset is essential in modern genomics, as uncompressing data and processing uncompressed data are inefficient and limiting. In addition to compressing genomic data, panMANs also provides a platform for direct phylogenetic or population genetic analyses by virtue of the evolutionary information they encode. However, as panMAN is still at its infancy, there exists internal alignment and missing information issues that can limit its potential as a bioinformatic analysis platform.

Proposed approach:

**2.1 Optimize panMAN’s internal alignment.**

Sequences on a panMAN are internally aligned by blocks of sequence segments. Blocks can be switched on or off between nodes to indicate insertions or deletions, respectively; this is called block mutations. We have observed that nodes with identical or nearly identical block sequences have on-state for different duplicated blocks to maximize compressibility (e.g. one block can be preferred over another when it only needs 1 nucleotide mutation while the other block needs 2 subsequent nucleotide mutations). This has created large segments of mis-aligned regions (Fig 3.) as blocks are aligned sequentially. It is as expected that MAFFT alignments are better than the panMAN alignments, but differences in the magnitudes of several hundreds or thousands indicate unoptimized block states. I will first develop a tool to optimize the block states on a panMAN to maximize internal alignment qualities.



**Figure 3.** *Figures on the left column show the difference between panMAN alignment distances and MAFFT alignment distances between every parent-child pair (panMAN difference - MAFFT difference) on a SARS panMAN containing 20,000 samples. Figures on the right column show the panMAN alignment distances and MAFFT alignment distances for panMAN-MAFFT differences greater than 0.*

**Identify potential duplicate blocks.** As each block can have different sequences at different node due to nucleotide mutations, I will collect a representative sequences for each block by picking the sequence of a block at a node at the lowest depth on the tree when the block is first turned on and has zero gap. If there are more than one node that satisfies this criteria at the same depth, the node that's reached first via a pre-order traversal is arbitrarily chosen. This sequence will be a block's representative sequence. I will then cluster these representative sequences using CD-HIT, and each cluster contains potential duplicate blocks.

**Optimize block states to maximize internal alignment qualities.** I will initialize a duplicated neighbor index that keeps track of the number of blocks turned on between neighboring duplicate blocks. At each node during a pre-order traversal, I will update this index using the block mutation on the edge, and pairs of neighboring duplicate blocks, in which one block is turned on only in the parent node and the other block is turned only in the child node, that have no blocks turned on between the block pair on one node (could be parent or child) are flagged as potential site for optimization. If the block pairs' sequences on their respective node exceed a specific similarity threshold, I will then swap the states between the block pairs and update necessary nucleotide mutations.

**2.2 Impute missing genetic information using a Markov chain and maximum likelihood.**

Following PanMAN's internal alignment optimization is the imputation of missing genetic information. The phylogenetic information on a panMAN provides a platform for phylogenetic imputation of ambiguous bases.

**Calculate tree-specific mutation spectrum and build a generalized time reversible model.** The mutation information encoded on a panMAN makes it trivial to compute a tree-specific mutation spectrum, which can be calculated by a single pre-order traversal and counting the mutations and base frequencies at each node. I will then capitalize on this to build a tree-specific generalized time reversible (GTR) model that will be used for imputation.

**Impute missing data from partial likelihoods and Markov chains.** I will first calculate the partial likelihoods at ambiguous positions in internal nodes using Felsenstein's tree-pruning algorithm. Starting from the leaf nodes to the root, I will iteratively calculate the partial likelihoods at each level of the tree in a post-order traversal. As when a sample contains an ambiguous base, its parent or chains of parent's could be encoded with an ambiguous base at the same position. I can extend this chain of ambiguous bases until I reach a node that has a canonical base at the position. Then I will start Markov chain from this node and calculate the most probable path on which the base will mutate. If a chain were to split, in cases where multiple closely related samples all have missing information at the same position, the Markov chain stops, and from which new chains will start.

**2.3 Evaluate internal alignment optimization and missing data imputation accuracy**

Results of internal alignment optimization will be evaluated the same way as shown in Fig 3., where I compare how different the panMAN and MAFFT alignments are. For bigger trees, I will random select a subset of node pairs and perform the same comparison. The accuracy of this imputation will be assessed by comparing the imputed SARS-CoV-2 panMAT to existing UShER-MAT and smaller phylogenies produced with maximum likelihood by IQ-tree-2. I will also simulated missing data from canonical bases and compare our imputation to the truths.

Preliminary results: A member of lab has developed an imputation method using fitch algorithm and tested it on several smaller trees. The results are promising and suggest greater improvements when maximum likelihood method is applied.

Research challenges, limitations, and alternative approaches: I think the most challenging part of Aim 2 will be identifying potential duplicate blocks, for which my approach assumes that sequences on a block are not heavily mutated and that block sequences remain relatively similar to each other throughout the tree. If this assumption is not true, pre-clustering the blocks becomes pointless. An alternatively approach, which will be less efficient, could be, instead of pre-clustering the blocks and making a duplicated neighbor index, iterating through on blocks and see the an on block with identical length on the other node but is off on the current node can be reached without encountering an on block.

**Aim 3: Develop an encoding of Pangenome Mutation-Annotated Ancestral Recombination Graph (panMAARG)**

Introduction and existing work: Since panMAN is a newly developed tool, it currently only supports haploid viral and microbial genomes. The next goal is to extend panMAN format to encode eukaryotic genomes, and this faces several challenges, one of which is the handling of higher recombination rate. The current panMAN encodes a recombination event as two nodes descending to a single node representing the ancestral recombined haplotype, from which an individual panMAT is created to represent all descendants of the recombination event. This approach scales poorly for genomes with high recombination rates, as it requires encoding as many panMATs in a panMAN as the number of recombination events.

Proposed approach:

To overcome this limitation, I propose borrowing ideas from the tree-sequence format of Ancestral Recombination Graphs (ARGs). By encoding each non-recombining segment as a single panMAT, entire chromosomes can be represented as highly compressible sequences of trees. This approach solves the scalability problem and offers a lossless ARG format. I will collaborate with our partners to develop the encoding and construction of a Pangenome Mutation-Annotated Ancestral Recombination Graph.

**1.1 Encode a layer of tree-sequence format over the panMAN format.**

One major difference between panMAARG and existing ARG format is that panMAARG will be lossless, encoding not only SNPs but also large INDELs and other complex mutations. The first step will be dividing up blocks into two separate types: core blocks and accessory blocks. Core blocks will contain sequences shared by all samples, similar to what are encoded in existing ARGs, while accessory blocks encode population or individual specific large INDELs and complex mutations. This will involve breaking up and creating new blocks from existing blocks on a traditional panMAN. ARGs will be constructed using the core blocks as the "backbone" and then be "decorated" with accessory blocks. Each non-recombining region of the genomes will thus contain its own core blocks and accessory blocks to constitute a single tree on a tree-sequence. This allows use to encode a layer of tree-sequences on top of the panMAN format without sacrificing much compressibility. In addition to encoding the changes in sequential tree topologies, there will be more blocks encoded but the aligned sequence will have the same length as a traditional panMAN but split up by recombination junctions. Since core blocks are shared by all samples, theoretically only accessory blocks will have block mutations, mitigating the increased number of block mutations due to the creation of more blocks. Block mutations and nucleotide mutations can also be intuitively indexed to indicate on which trees they will be executed.

**1.2 Add additional input to ts-infer to maximize information used for ARG inference.**

Existing ARG methods use only SNP information to infer samples' genealogical relationships and history, while INDEL or complex mutations can provide additional information that can be inferred from SNPs only. One method to accomplish is to encode INDEL similar to a multi-nucleotide polymorphism, where gaps can be represented as a '5th' base and INDELs could be treated as substitutions to or from the '5th' base.

Research challenges, limitations, and alternative approaches: I expect to face the most challenges in Aim 3. I will first focus on formalizing and optimizing my methods in more detail with test codes and test results at each step in the development to ensure feasibility. I can evaluate my ideas by comparing how much compressibility is achieved using tree-sequence encoding compared to the existing network encoding for recombination events. The concept of encoding panMAN in a tree-sequence might sound easy and intuition but I expect to face major hurdles in implementation, and changes to my original approach are expected. One worry I have about encoding panMAARG is how much compressibility will be sacrificed to encode lossless information compared to existing ARG tools.