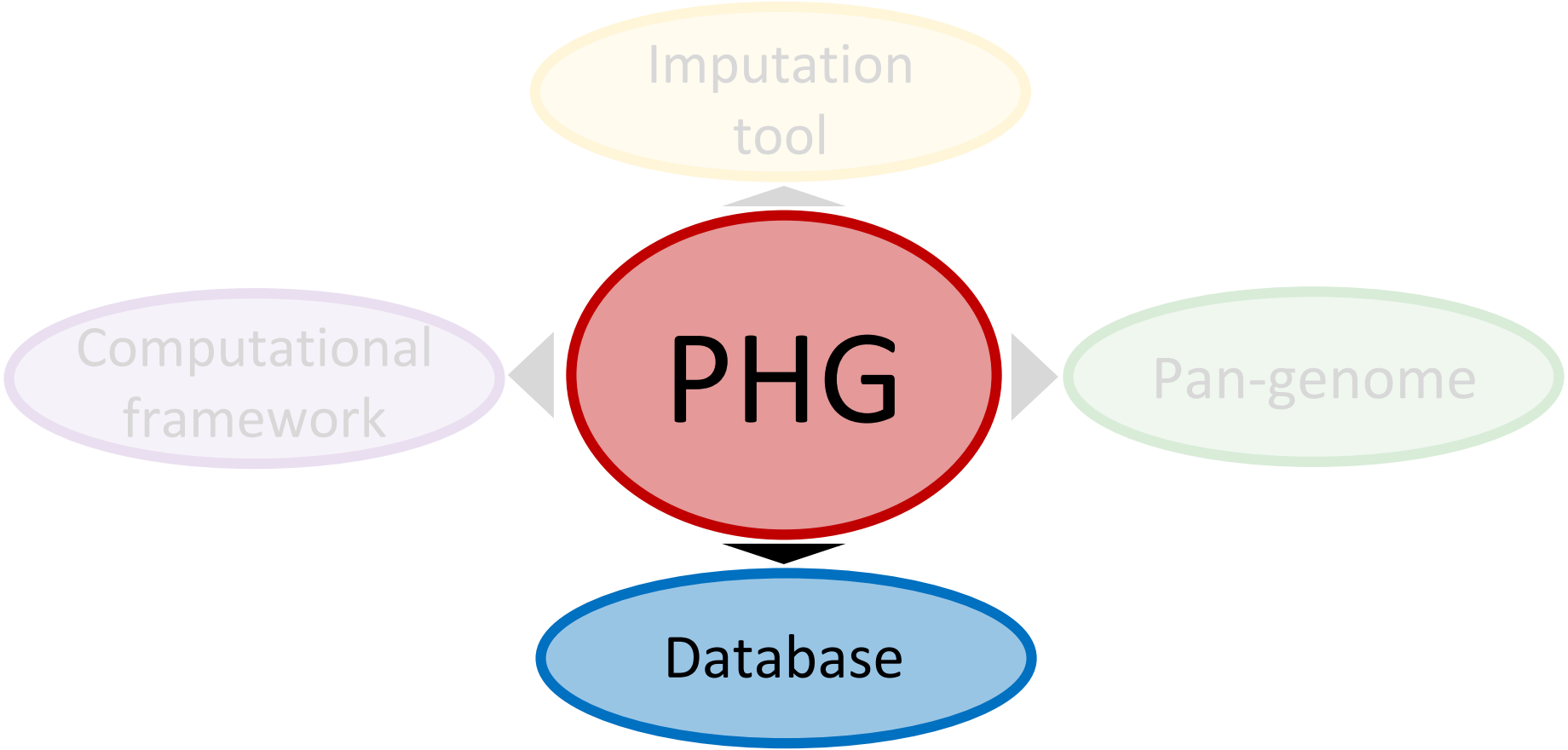
Populating the PHG Database

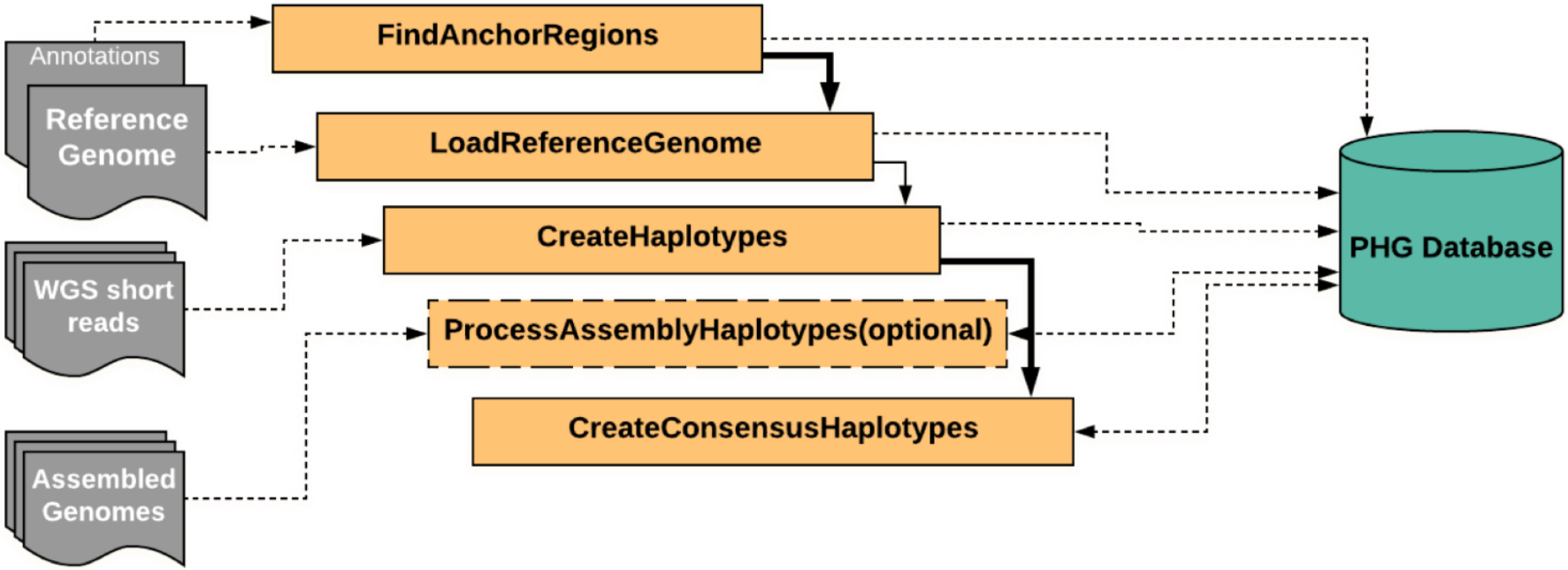
Lynn Johnson
Buckler Lab
June, 2019


Imputation tool
Computational framework
PHG
Pan-genome
Database

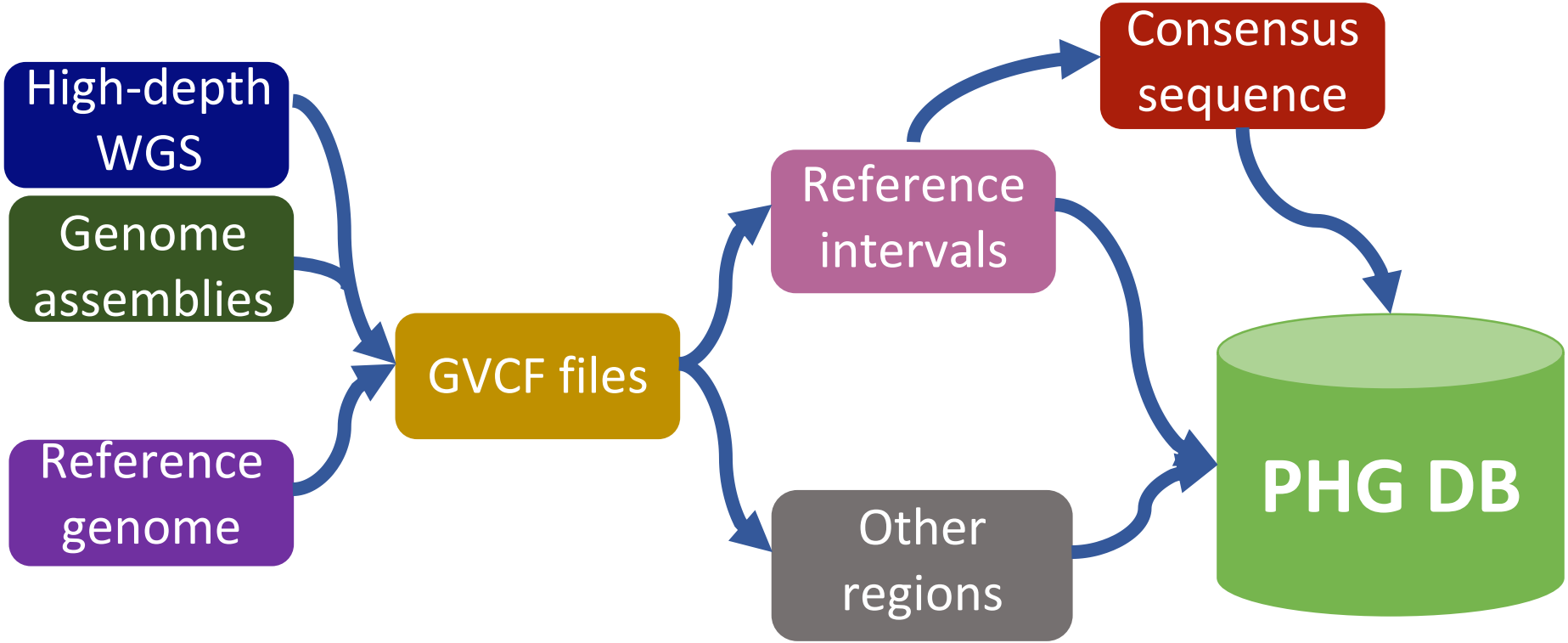
PHG DB Overview

- DBs are crop specific, i.e. each DB has data for a single species
  - Each species has their own reference
  - Each species has their own reference-specific anchors
- Pipelines for populating the database:
  - Data from Reference genomes, assemblies, GATK raw haplotypes, consensus analysis
- Pipelines for using the database for imputation:
  - Path and haplotype count data stored for inferred genotypes

Building the PHG database



Building the PHG database



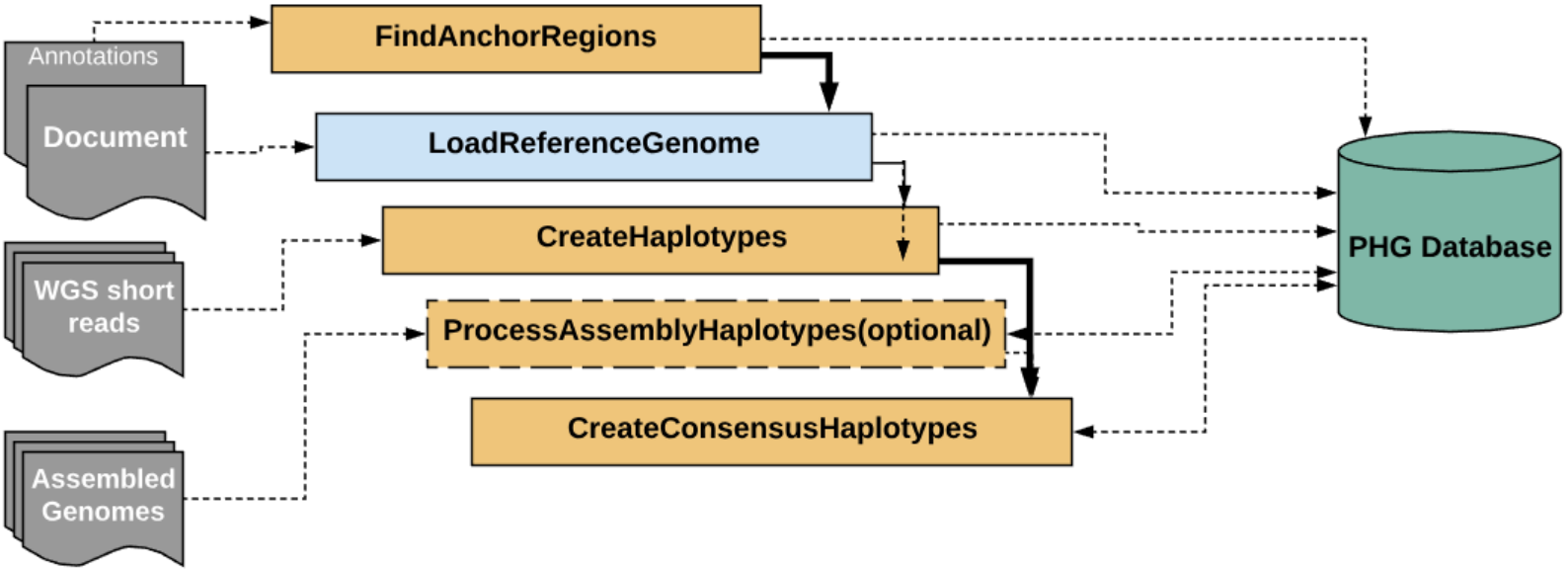
Reference Ranges

- Intervals are defined by a bed file input
- There can be no overlapping intervals
- Once reference ranges are given, they cannot be changed
- ***But*** users can specify different sets of reference ranges to be used versus ignored
  - If a range is found to be difficult or inconsistent, don't use it.
  - If a range is close to a causal locus include it.
  - Used ranges should be conserved and easy to align to. => They are often genic.
- A specific set of reference ranges are defined by a ***Method***.

Example.bed

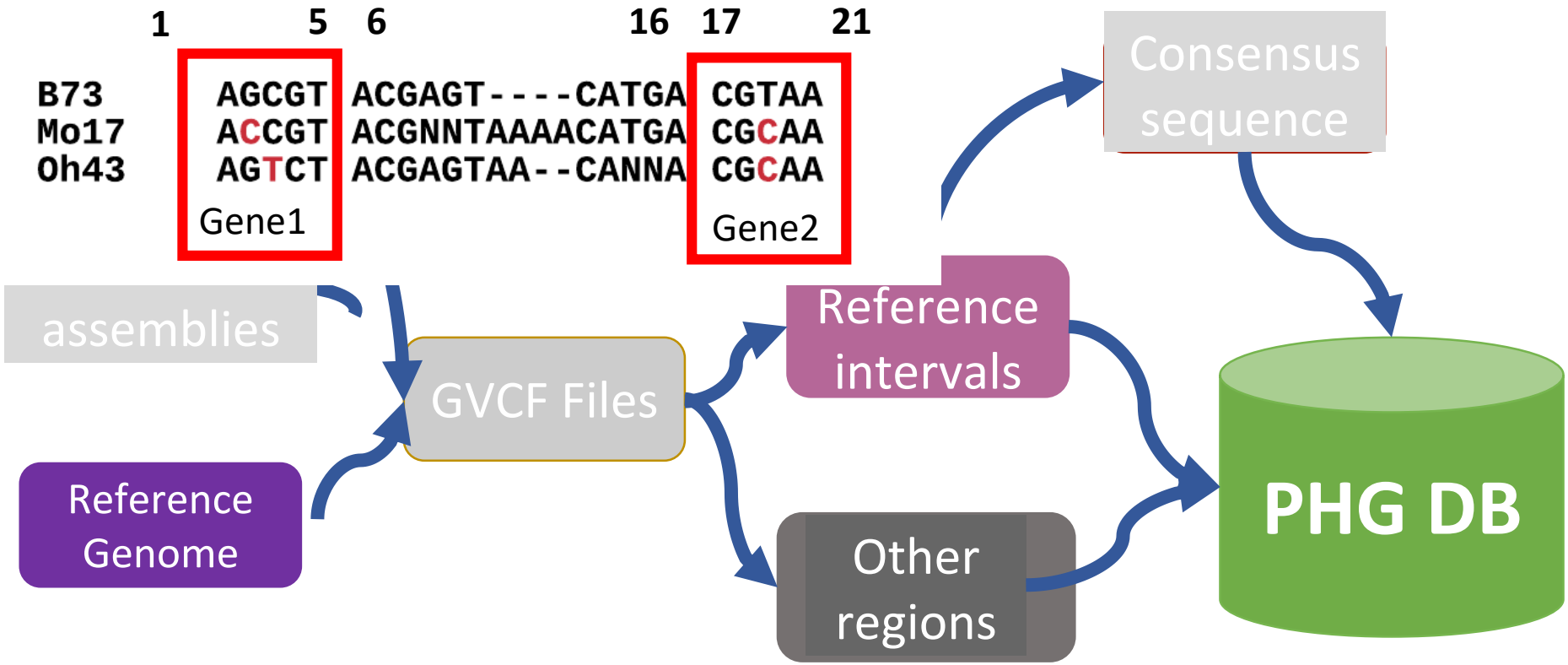
Chr	Start	Stop
5A1	19129	24258
5A1	228425	231745
5A1	308819	309219
5A1	334508	339187
5A1	363835	364358
5A1	640706	643475
5A1	664306	676925
5A1	770286	773382
5A1	857915	858546
5A1	1088848	1089955
5A1	1113620	1114871
5A1	1135096	1138280
5A1	1193534	1203012
5A1	1215288	1227432
5A1	1240949	1241819
5A1	1264806	1276851

Loading the Reference Genome



Loading the reference ranges and loading the reference genome haplotypes occurs together

Loading the reference ranges



Reference genome is the first sequence loaded to the PHG database

- Usually the reference genome is represented as haploid: a single string of bases with a unique base per genomic position
- It is usually not represented as heterozygous genotypes, which would require two (for diploids) bases per genomic position
- The P***H***G requires haplotypes, which you can get from fully inbred individuals or from phased diploids

Phased vs. un-phased diploids

This is a Practical ***Haplotype*** Graph

Haplotypes

Phased genotypes:

Different

Unphased genotypes:

Same...

Inbreds are automatically phased

Working with inbreds simplifies things a lot here

The PHG software has data structures to deal with outcrossed diploids if needed

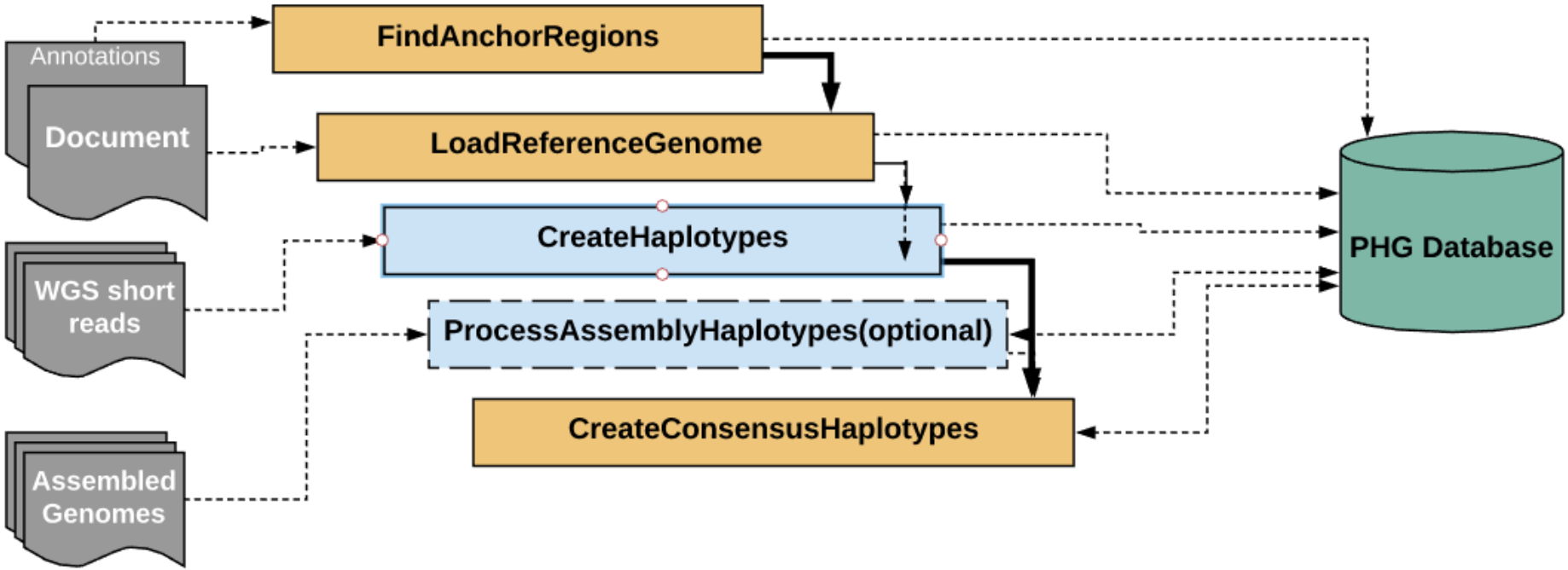
We will not go into that in this workshop

gametes, gamete groups, gamete haplotypes

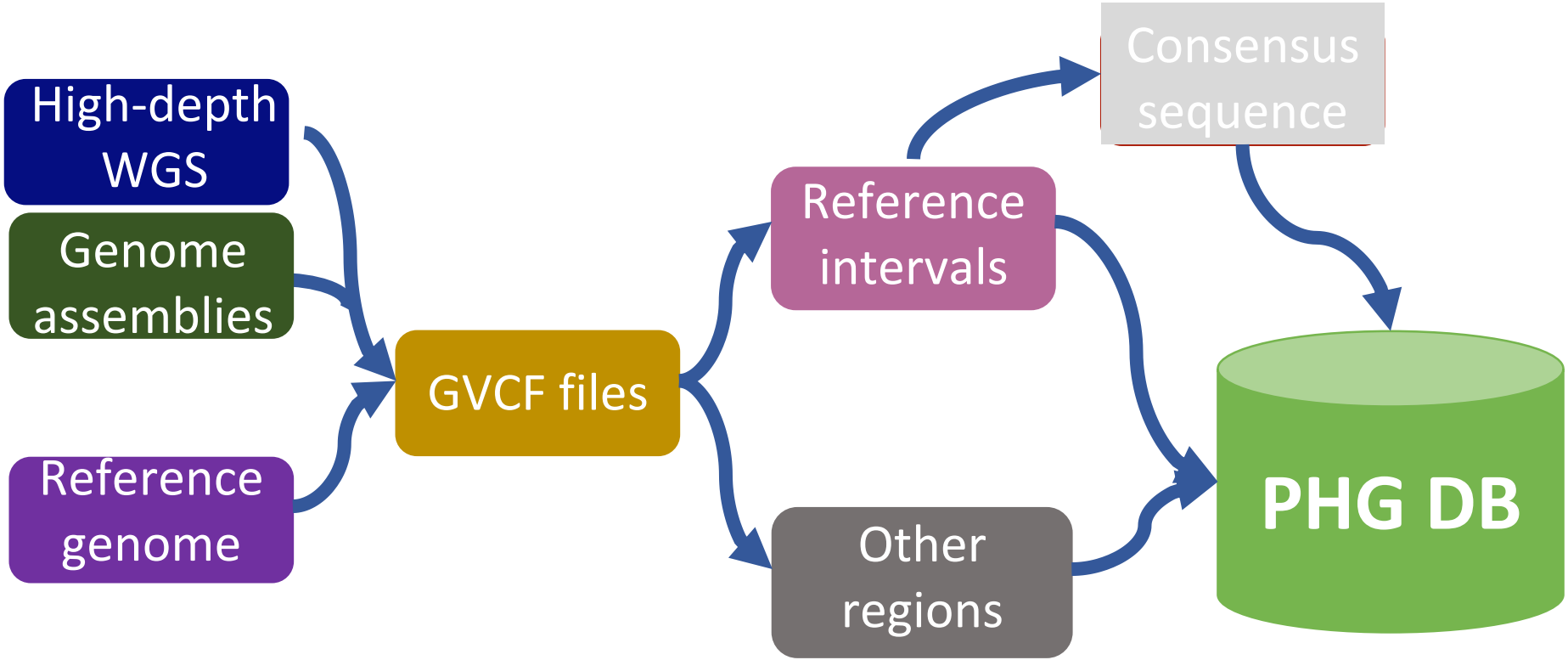
PHG haplotype data may represent a single gamete, or may represent the consensus of several gametes.

- Reference, assembly and GATK raw haplotypes have data derived from a single gamete.
- Consensus haplotypes are derived from multiple gametes
- A db table keeps a mapping from each individual gamete to the groups in which it is represented.

Loading Raw Haplotypes



Building the PHG database



Sources for Raw Haplotypes

- Assemblies
  - Sequence aligned at a chromosome level
- fastq files of WGS sequence
- bam files from WGS sequence aligned to Reference
- gvcf files of WGS sequence
- Data from all 4 types may be input

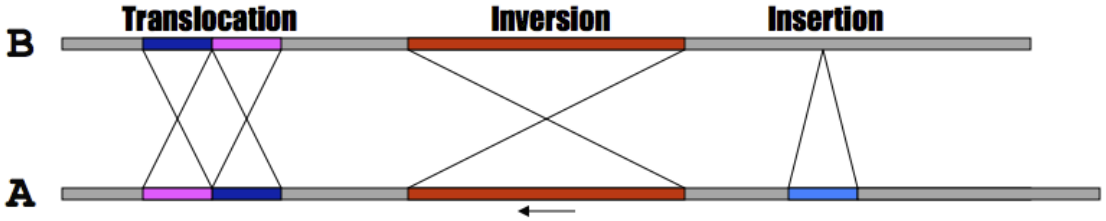
How do I include my Assembly Genome?

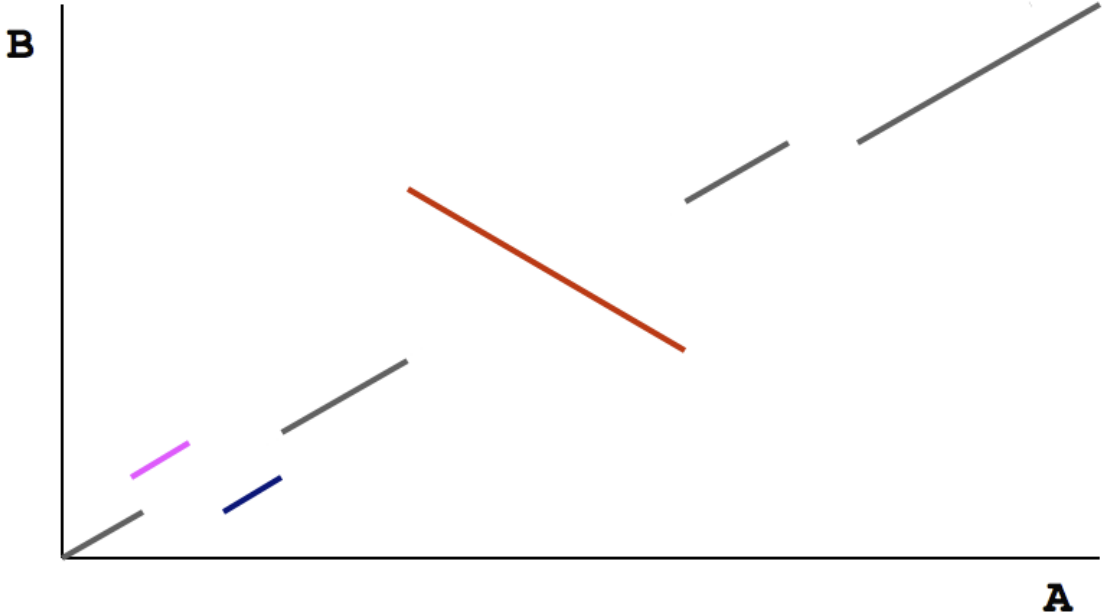
- Assemblies provide provide valuable information on intra-genic variation
- Sequencing or assembly errors may exist with the chosen reference genome for a species.
- The inclusion of additional assembled lines for a given species increases the accuracy of identifying SNPs and regions of interest.
- Improve annotation of the genome

This will become the dominant pipeline to load the PHG

Finding the reference intervals on the assembly is the key task

- Assemblies are frequently smaller contigs and scaffolds
  - (PHG requires chromosome level alignment)
- Alignment is necessary to break the assembly into reference intervals
- Alignment identifies the variants (including insertions and deletions)
- Challenges: translocations, inversions, insertions

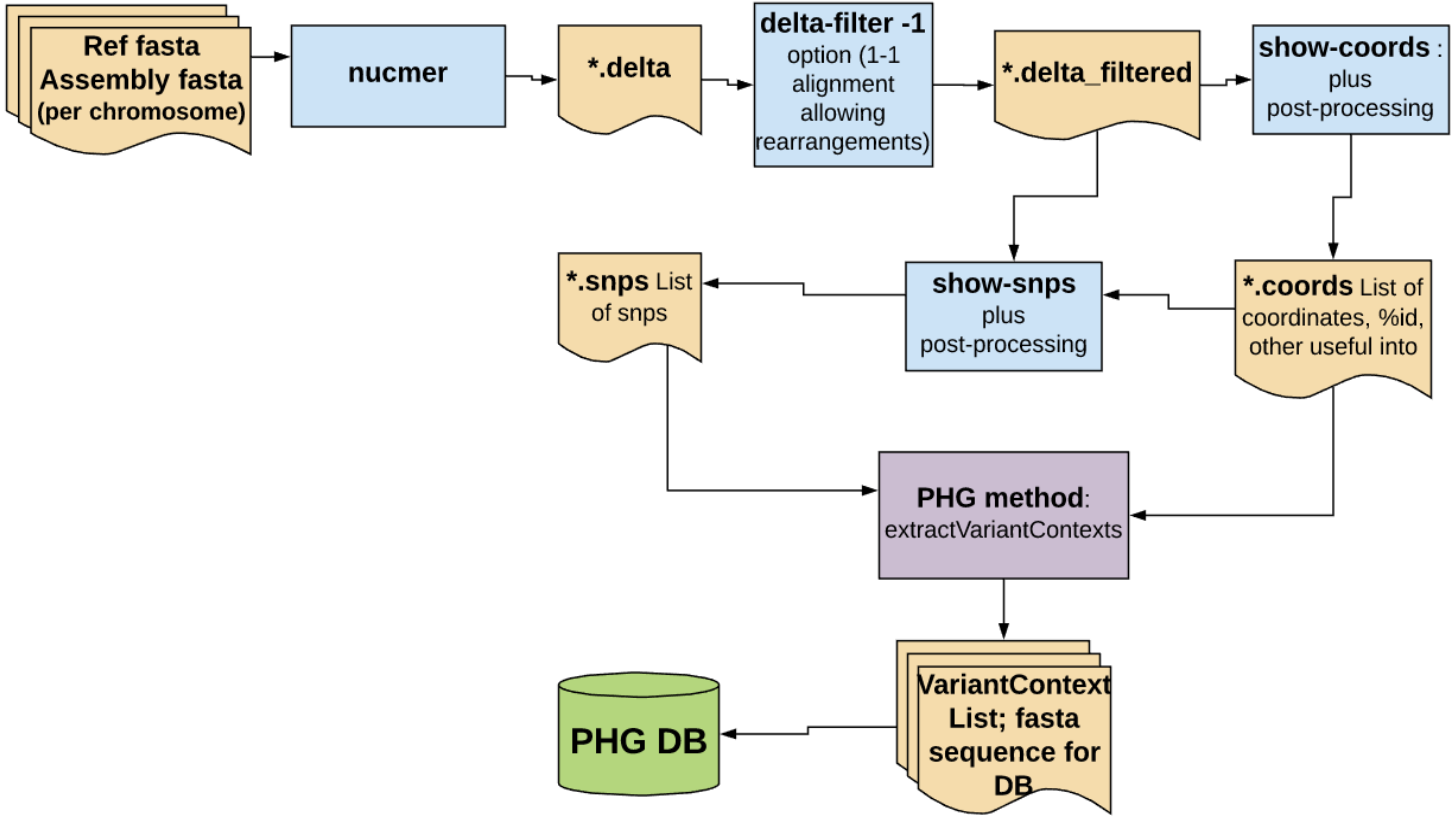




Example alignment showing translocation, inversion and insertion.

Slide from Michael Schatz: http://schatzlab.cshl.edu/teaching/2011/2011.Lecture4.Alignment%20and%20Assembly.pdf
More example alignment type slides: http://mummer.sourceforge.net/manual/AlignmentTypes.pdf

Identifying Assembly Haplotypes: Mummer4



All processes in blue are mummer4 commands

Raw haplotypes from WGS reads

Fastq :

- Use if assemblies or gvcfs are not available

BAMs:

- Saves bwa alignments

GVCFs:

- Saves steps, so saves time
- Use if available and you are comfortable with the alignment method and parameters

How to go from WGS to Haplotype

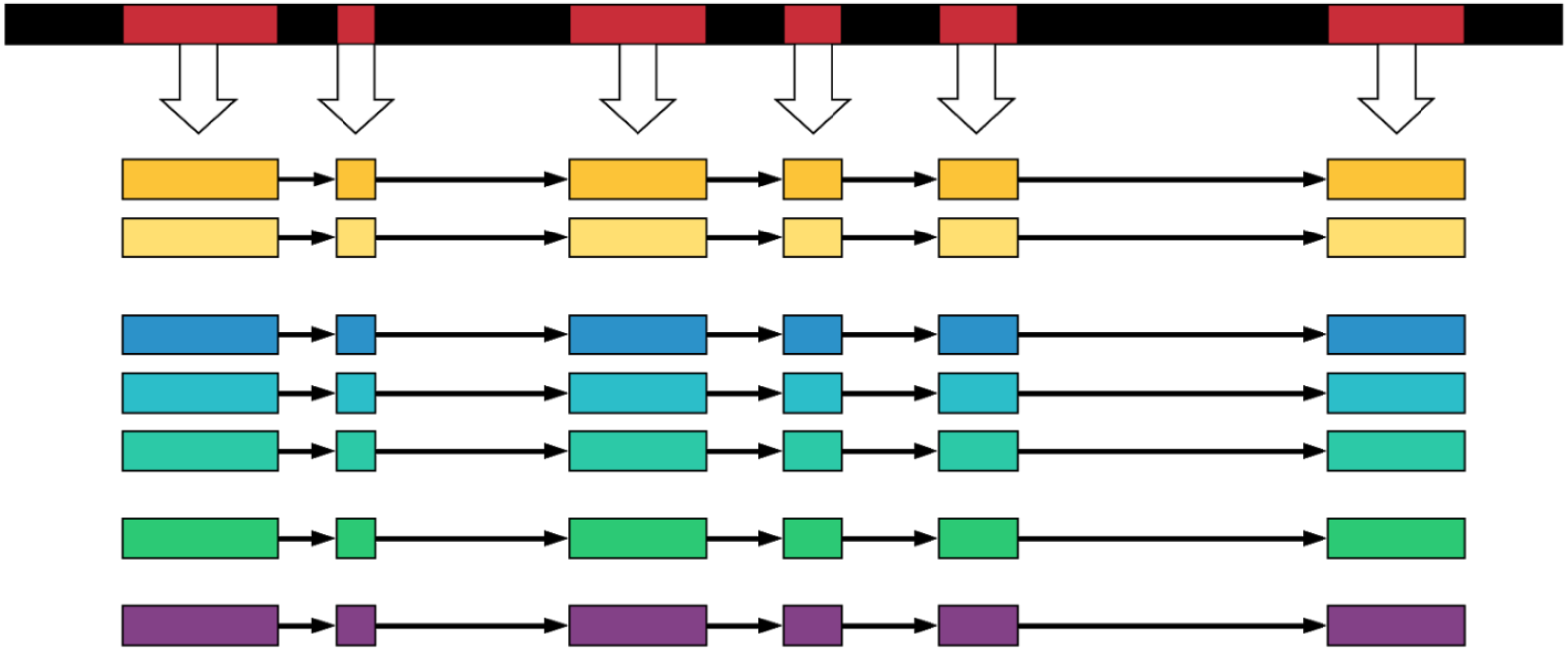
- Align to Reference
- Filter BAM by MapQ
- Run GATK HaplotypeCaller on all bams for a taxon
- Filter GVCFs
- Extract out haplotypes from GVCFs and upload to DB

Storing Raw haplotypes to DB

- Haplotype sequences are created for each reference range interval and stored to the haplotypes table

- Gamete group for a raw sequence has only 1 member.

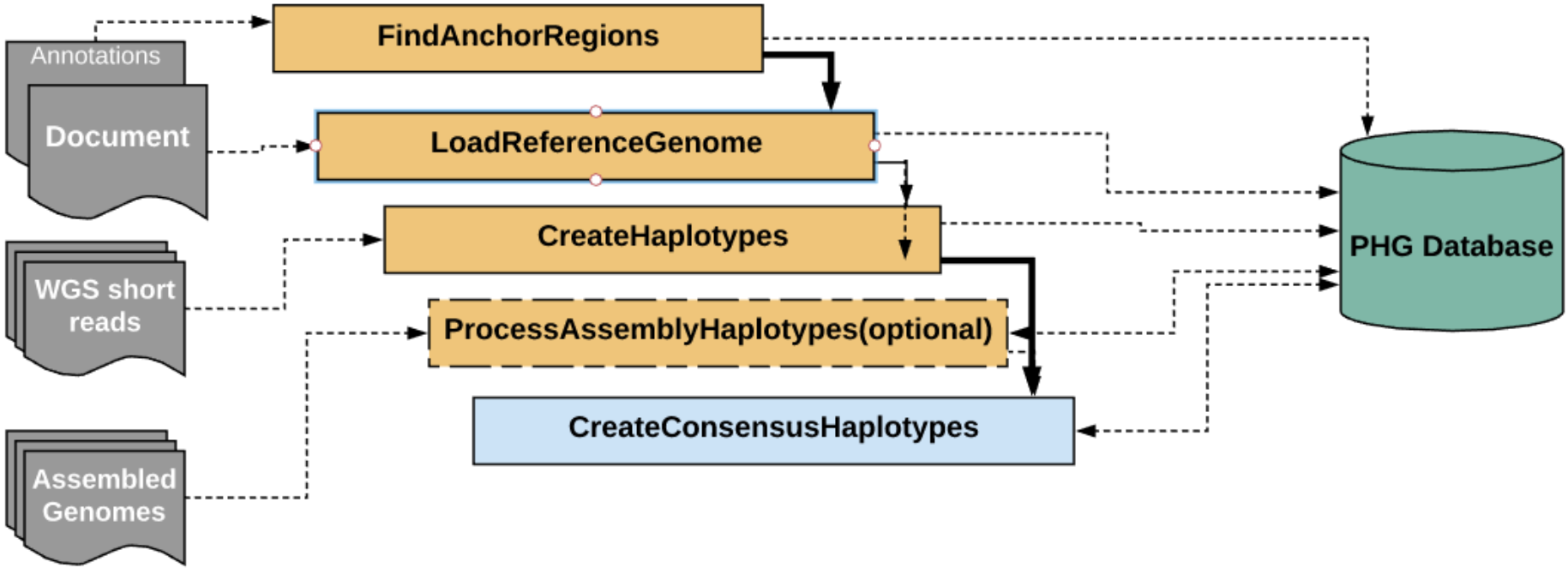
Raw Haplotypes



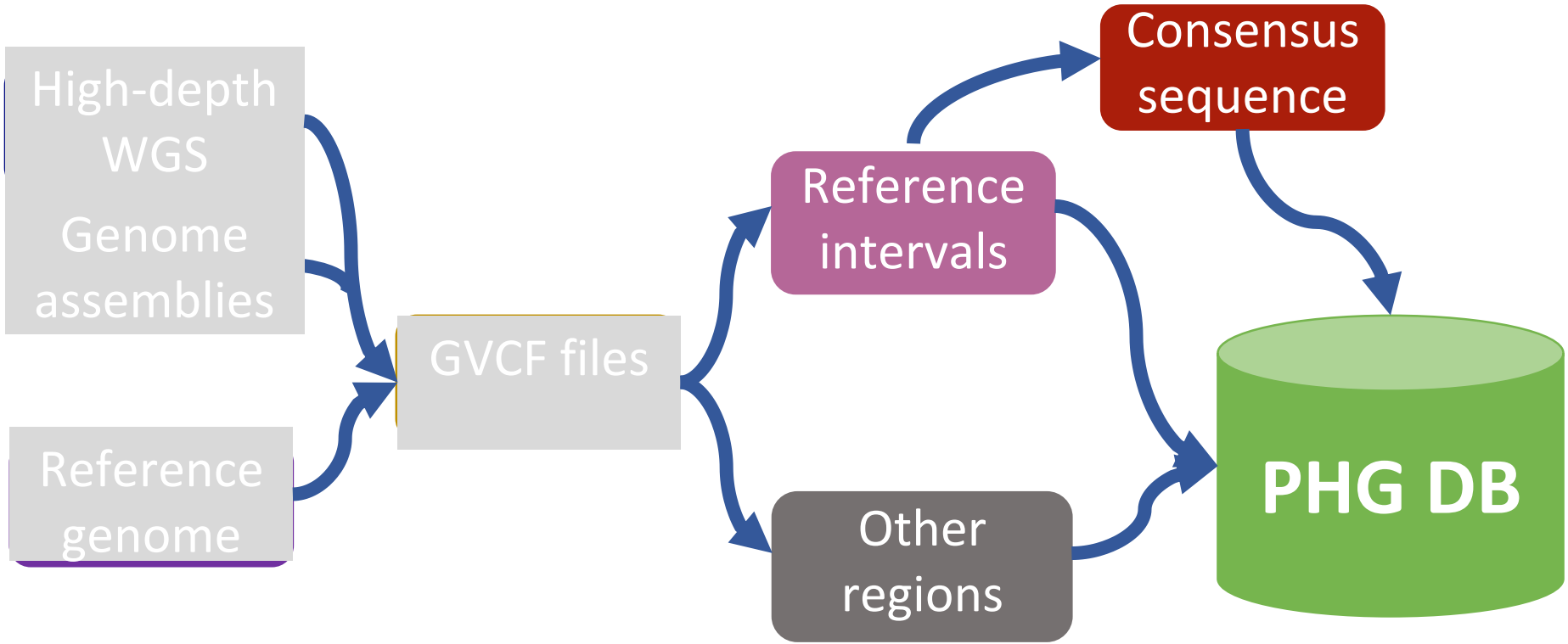
Haplotypes table

- Holds sequence for all haplotypes: ref, assembly, GATK raw haplotypes, consensus haplotypes
- **A method id** identifies type of haplotype data: ref, assembly, GATK raw haplotypes, consensus
- **A gamete group id** identifies the taxa associated with the haplotype
- **A reference range id** identifies the reference range to which this haplotype is mapped.

Loading Consensus Haplotypes



Building the PHG database



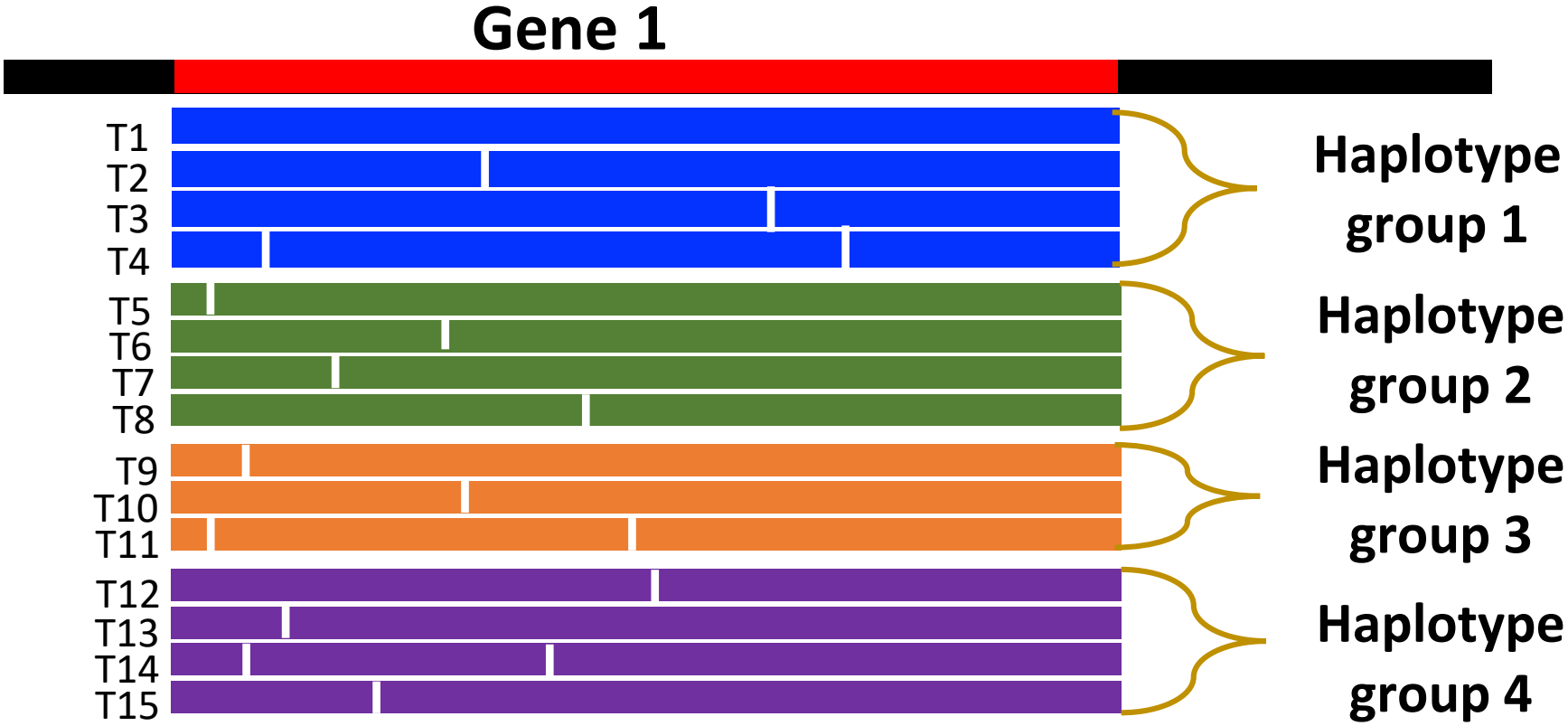
Consensus Haplotypes

- The consensus haplotypes are aggregated haplotypes of similar taxa at each reference range
- After this is done, there are haplotypes with a consensus method and these haplotypes are associated with multiple (not just one) taxa
- Gamete group id identifies taxa included in each consensus

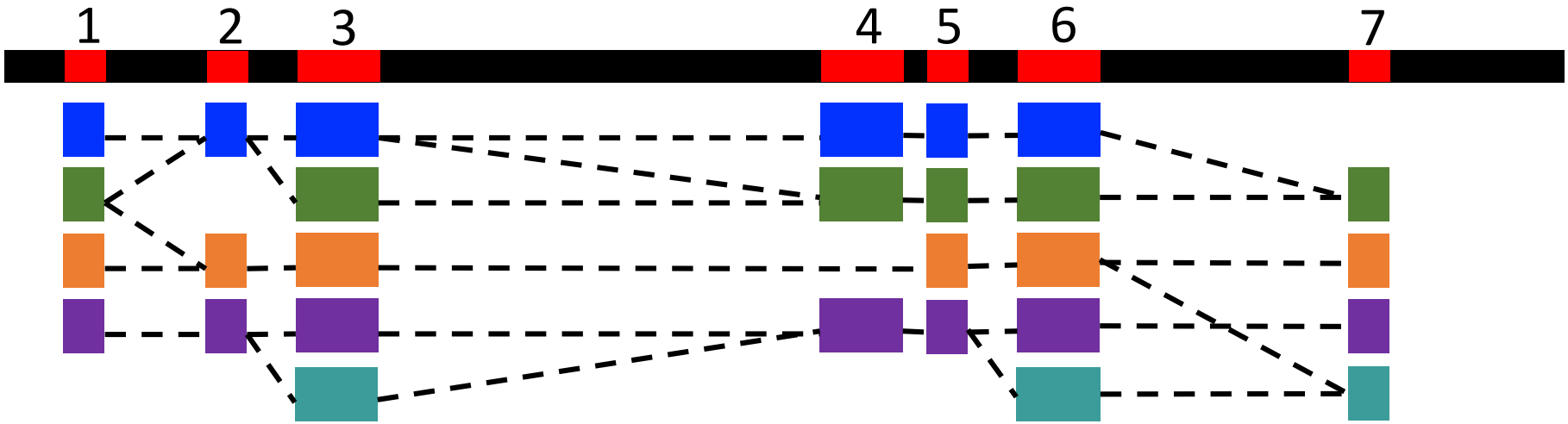
Create Consensus - Basic Idea

- For each reference range
  - Build a UPGMA tree based on pairwise distance between any two haplotypes
  - Apply a threshold cutoff (mxDiv)
  - Take the remaining clusters and merge haplotypes

Haplotypes at a single gene in the PHG

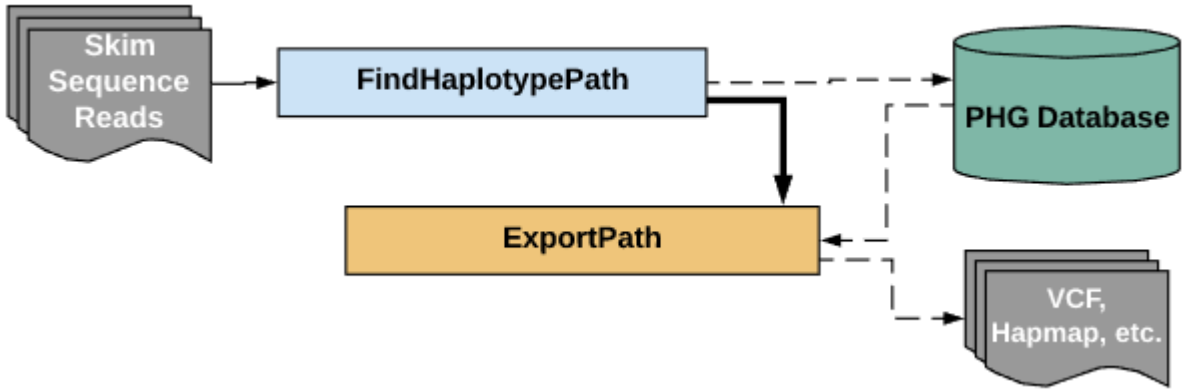


Consensus haplotypes across the genome



Haplotypes for new individuals are predicted based on similarity to genotypes in the graph

Phase 2: Path data for inferred genotypes



Storing Paths

Paths through the haplotype graph are stored in the DB during Phase 2 of the pipeline. Phase 2 of the pipeline does the following:

- Maps reads from skim sequences to stored haplotypes (consensus or raw)
- Uses stored graph data to infer genotypes
- Results are stored to a paths table: an ordered list of haplotype_ids (from haplotypes table) representing the path through the graph

Summary

- One database instance per crop.
- PHG database stores haplotype data from reference genomes, assemblies, GATK created raw haplotypes, and consensus haplotypes.
- Haplotype data is stored relative to a reference genome, and on a per reference range basis.
- PHG DB data is used to infer and store data regarding paths through the haplotype graph.
- API commands to store and access the data are available via TASSEL plugins.
- PHG data can be accessed from R.
- https://bitbucket.org/bucklerlab/practicalhaplotypegraph/wiki/Home

Appendix A: DB Schema

Database support:

- PostgreSQL
- SQLite

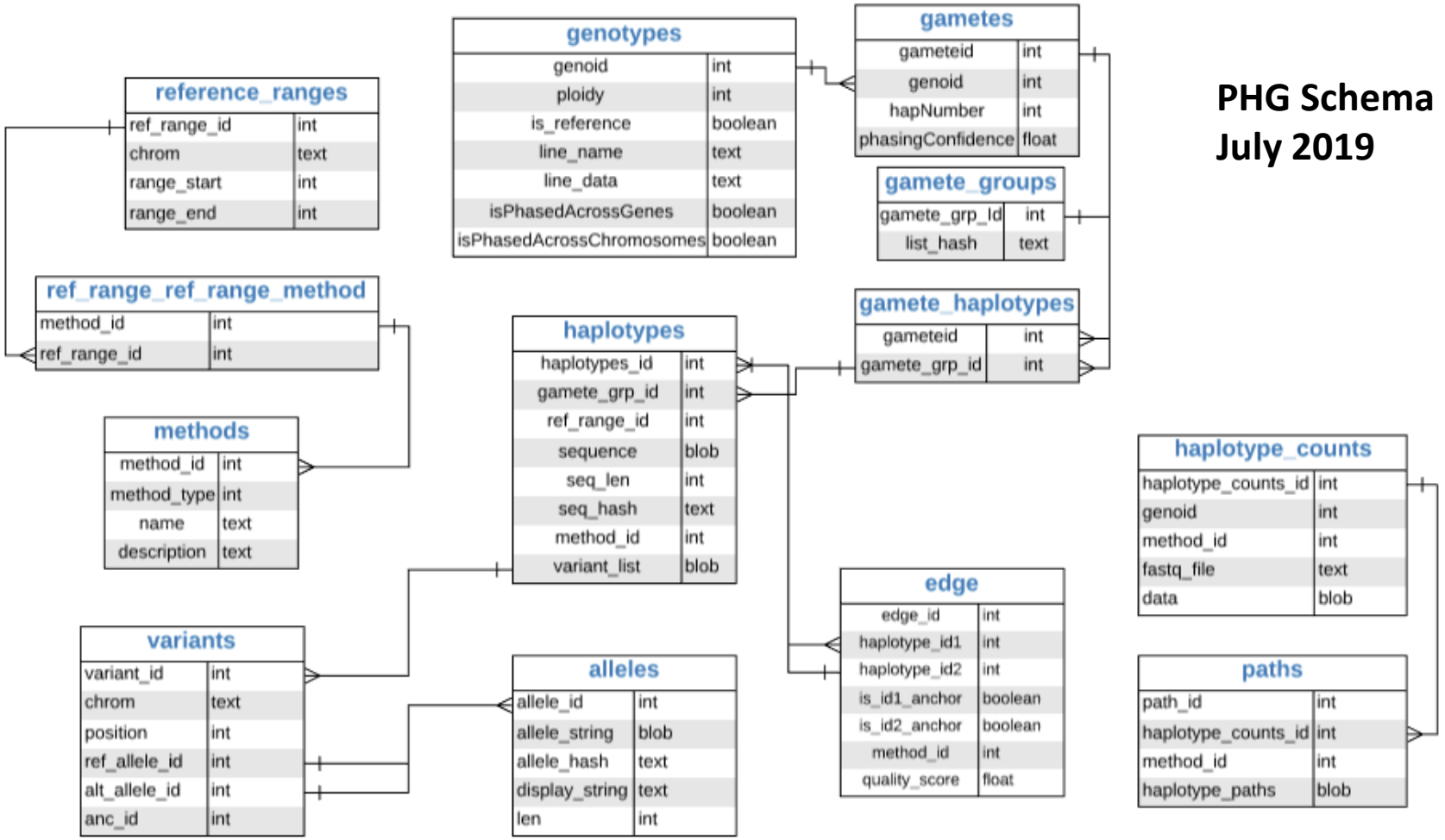
Sqlite vs PostgreSQL

Sqlite:

- Embedded, compact, serverless
- Single file output
- No built-in data encryption
- Good for single users and debugging

PostgreSQL:

- client/server implementation
- better security features
- Better for multi-user environment


PHG Schema
July 2019
genotypes
genoid int
ploidy int
is_reference boolean
line_name text
line_data text
isPhasedAcrossGenes boolean
isPhasedAcrossChromosomes boolean
gametes
gameteid int
genoid int
hapNumber int
phasingConfidence float
gamete_groups
gamete_grp_Id int
list_hash text
reference_ranges
ref_range_id int
chrom text
range_start int
range_end int
ref_range_ref_range_method
method_id int
ref_range_id int
gamete_haplotypes
gameteid int
gamete_grp_id int
haplotypes
haplotypes_id int
gamete_grp_id int
ref_range_id int
sequence blob
seq_len int
seq_hash text
method_id int
variant_list blob
methods
method_id int
method_type int
name text
description text
haplotype_counts
haplotype_counts_id int
genoid int
method_id int
fastq_file text
data blob
edge
edge_id int
haplotype_id1 int
haplotype_id2 int
is_id1_anchor boolean
is_id2_anchor boolean
method_id int
quality_score float
variants
variant_id int
chrom text
position int
ref_allele_id int
alt_allele_id int
anc_id int
alleles
allele_id int
allele_string blob
allele_hash text
display_string text
len int
paths
path_id int
haplotype_counts_id int
method_id int
haplotype_paths blob