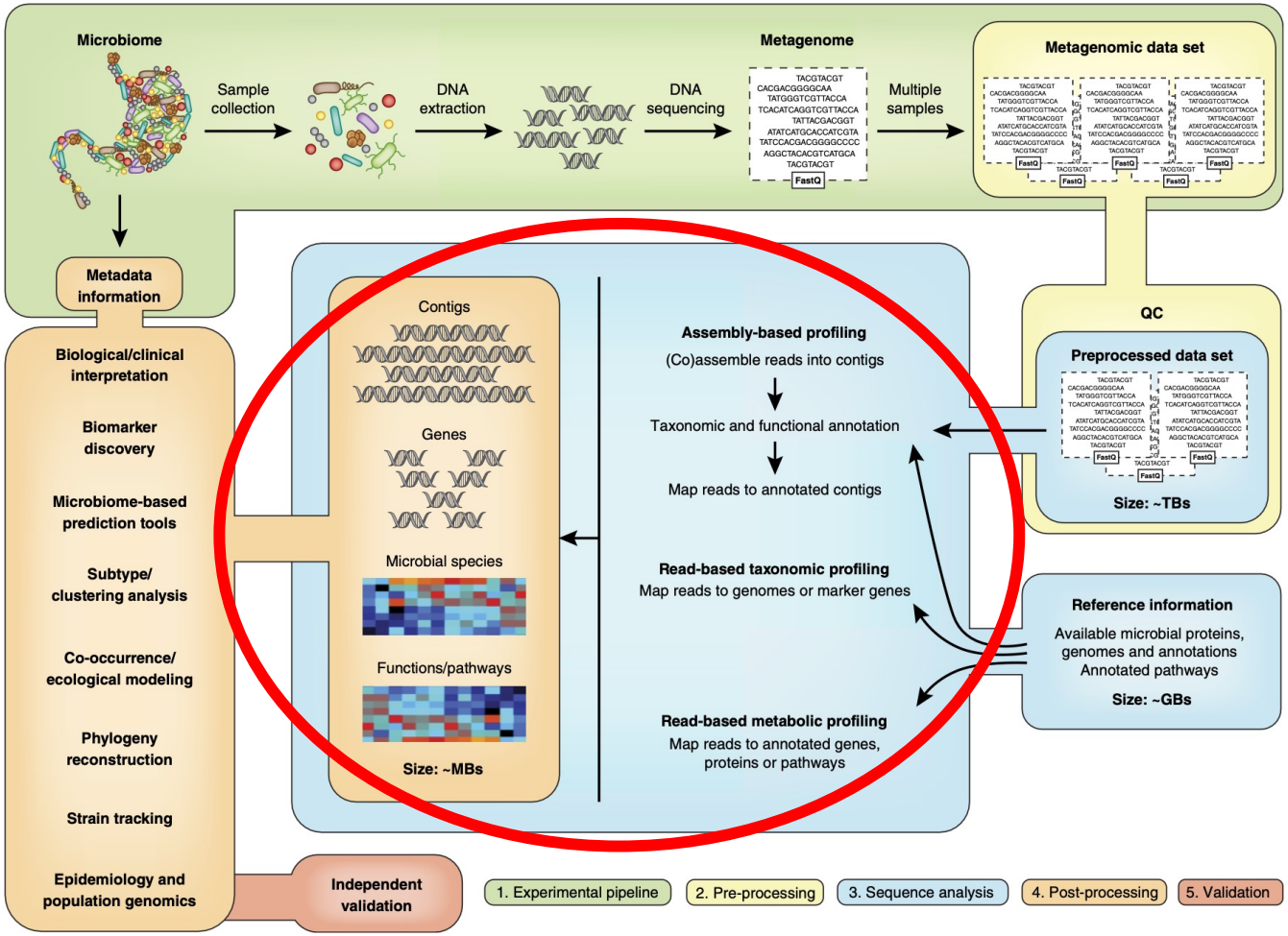
Shotgun metagenomics

Dr. Natalia Zajac

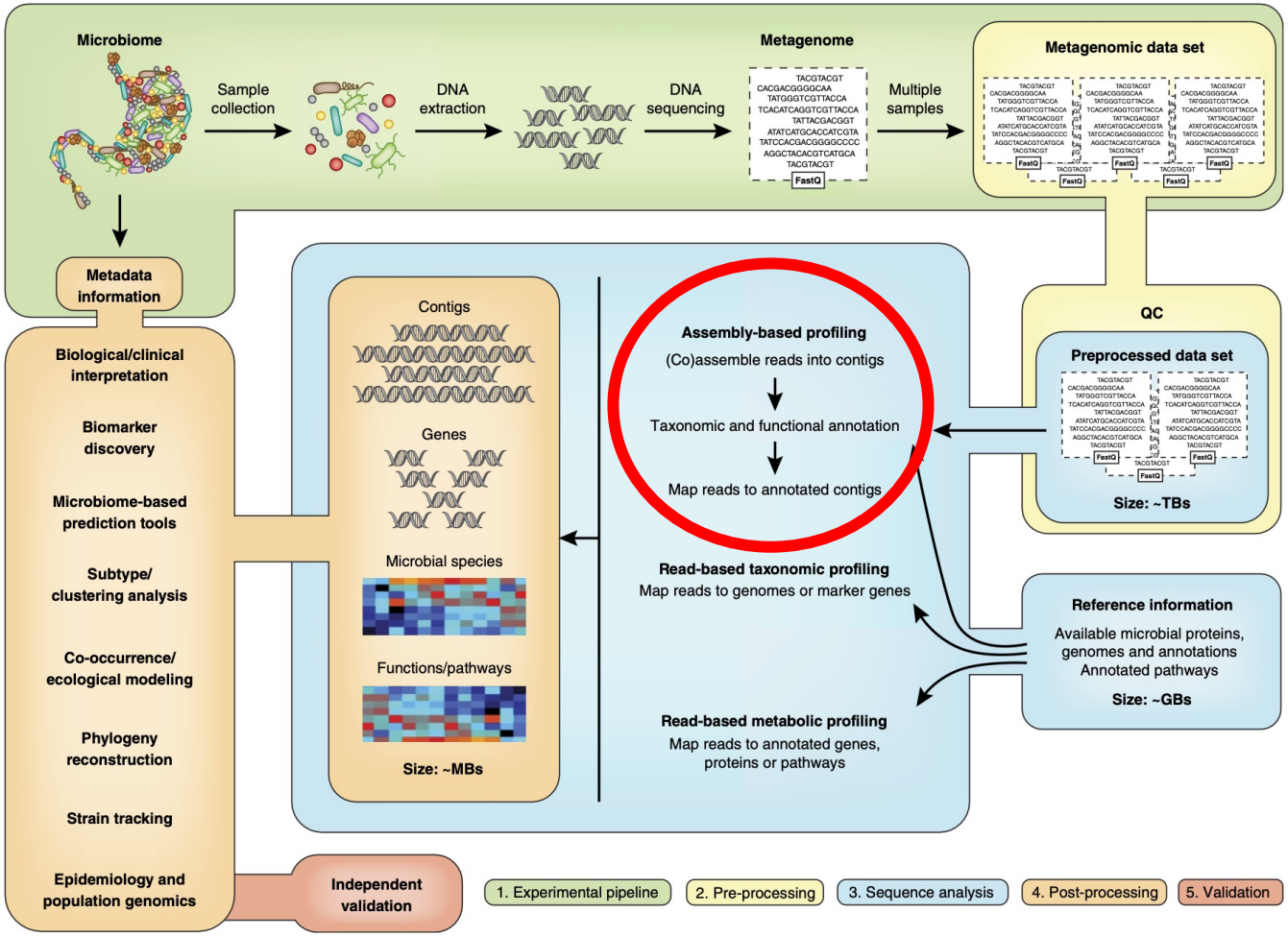
Metagenomics, 03.2023

Overview

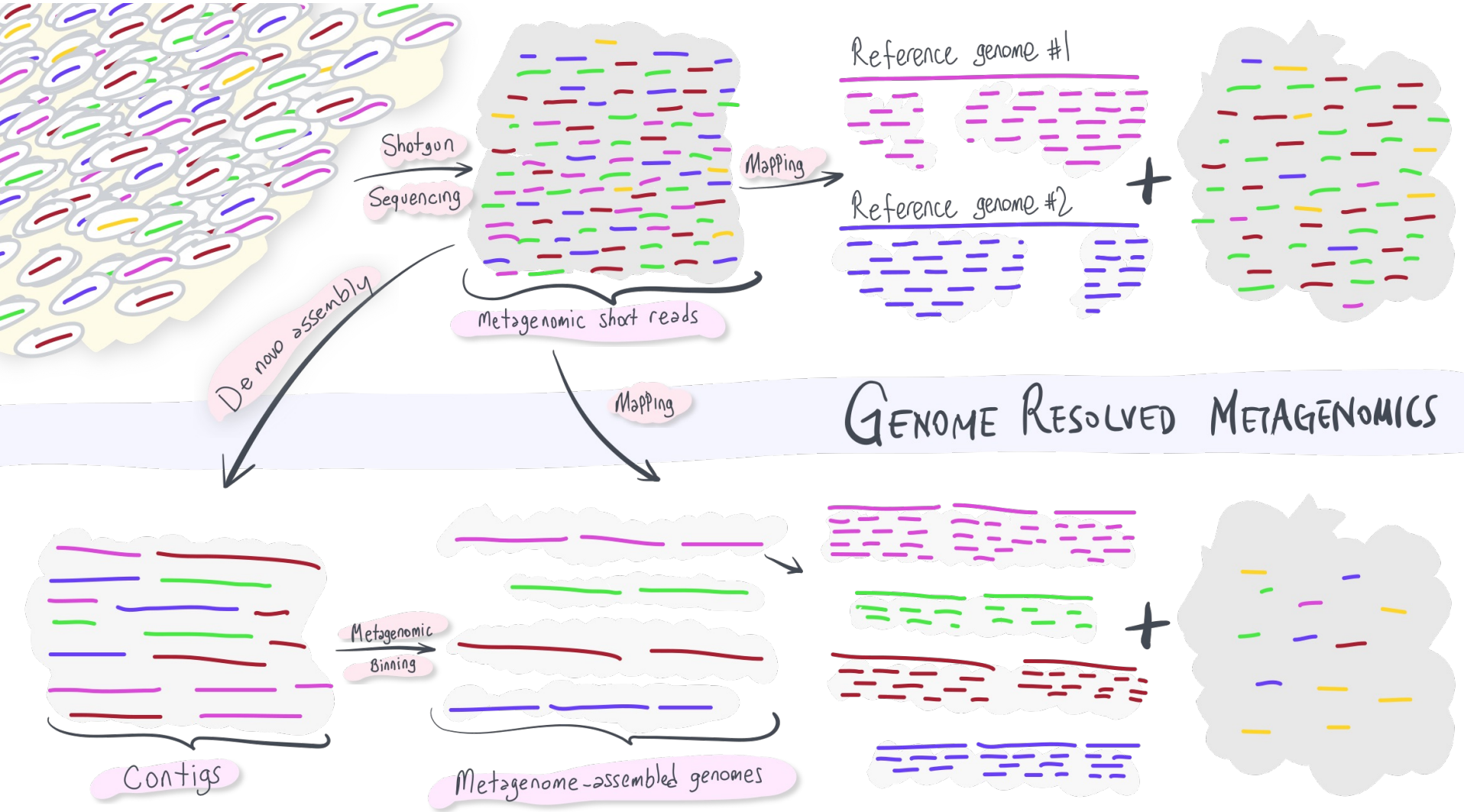


3. Metagenome assembly and annotation

Overview

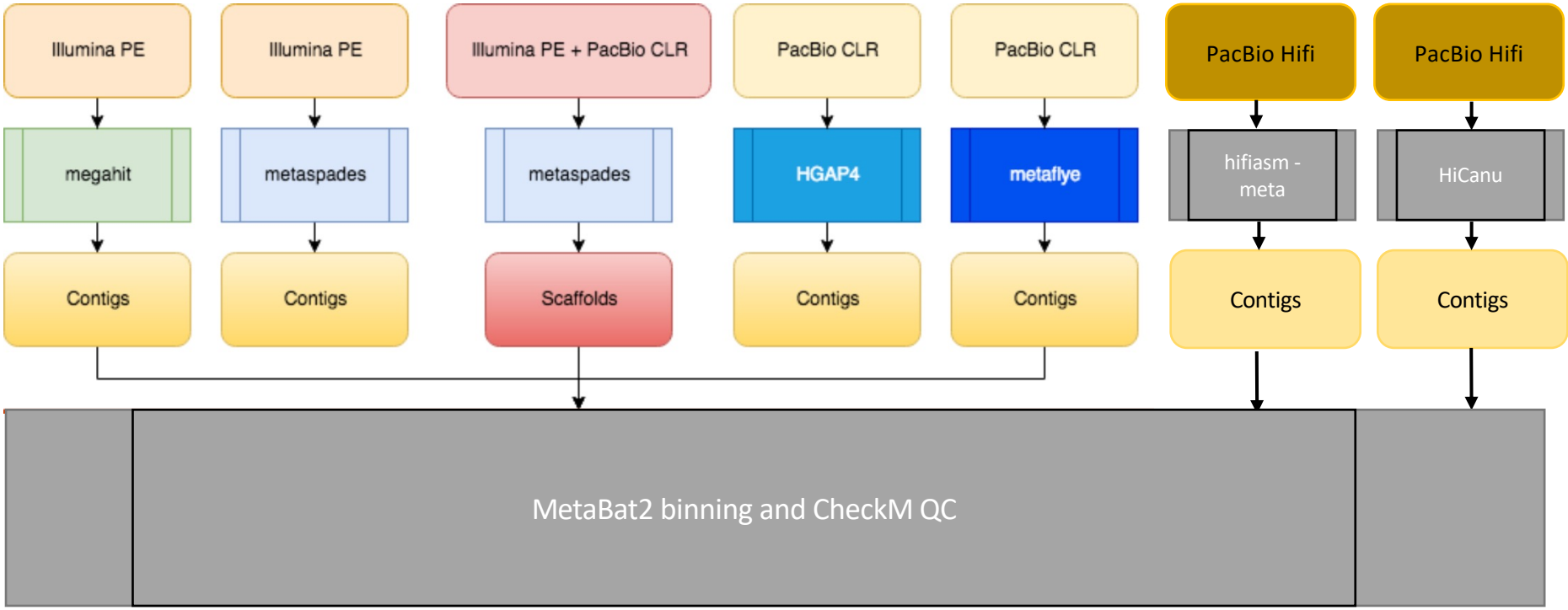


3. Metagenome assembly and annotation

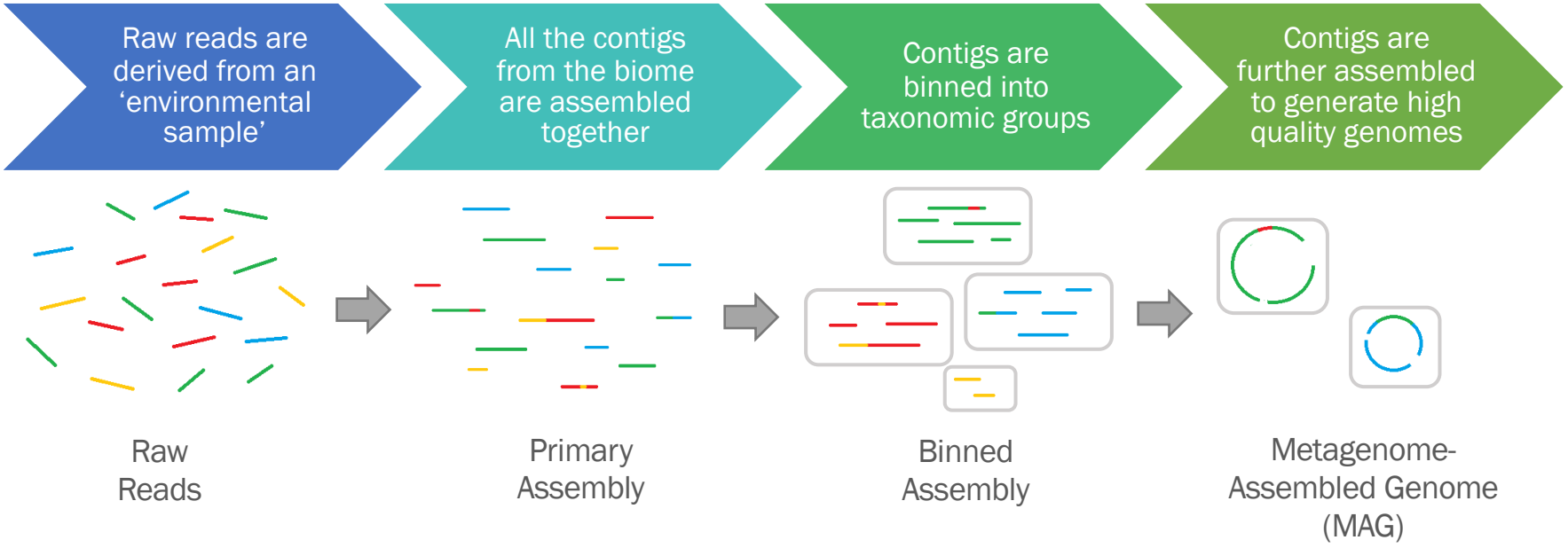


https://merenlab.org/momics/#genome-resolved-metagenomics

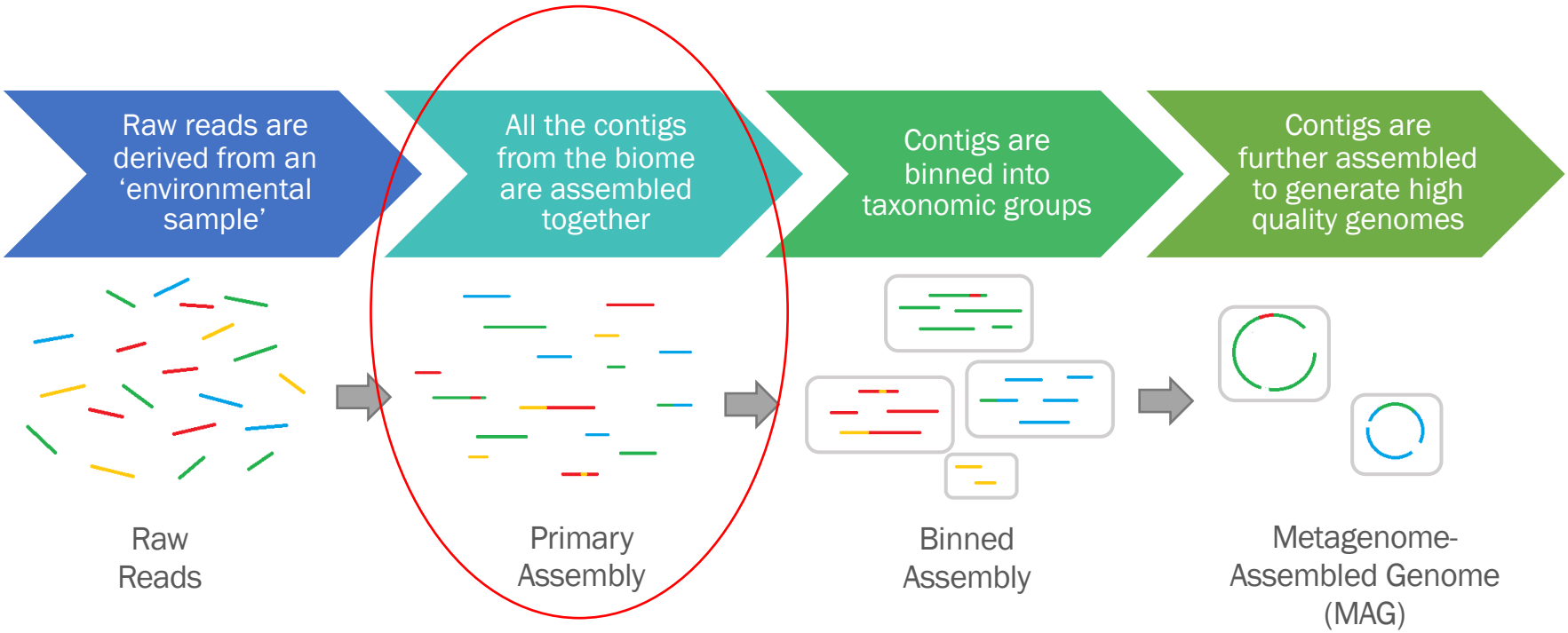
Metagenome assembly and binning workflow



Structuring Metagenomic Studies: Types Of Metagenomic Assembly



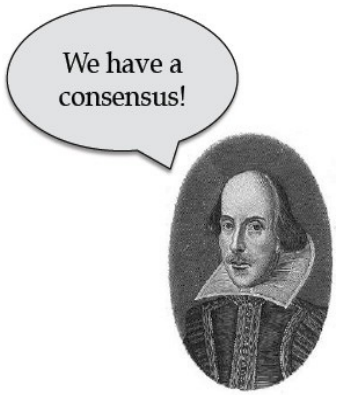
Structuring Metagenomic Studies: Types Of Metagenomic Assembly



Assembly – The Problem

- ***Reads***

```
ds, Romans, count
ns, countrymen, le
Friends, Rom
send me your ears;
crymen, lend me
```



- ***Overlaps***

```
Friends, Rom
     ds, Romans, count
           ns, countrymen, le
                 crymen, lend me
                       send me your ears;
```

- ***Majority consensus***

```
Friends, Romans, countrymen, lend me your ears;
```

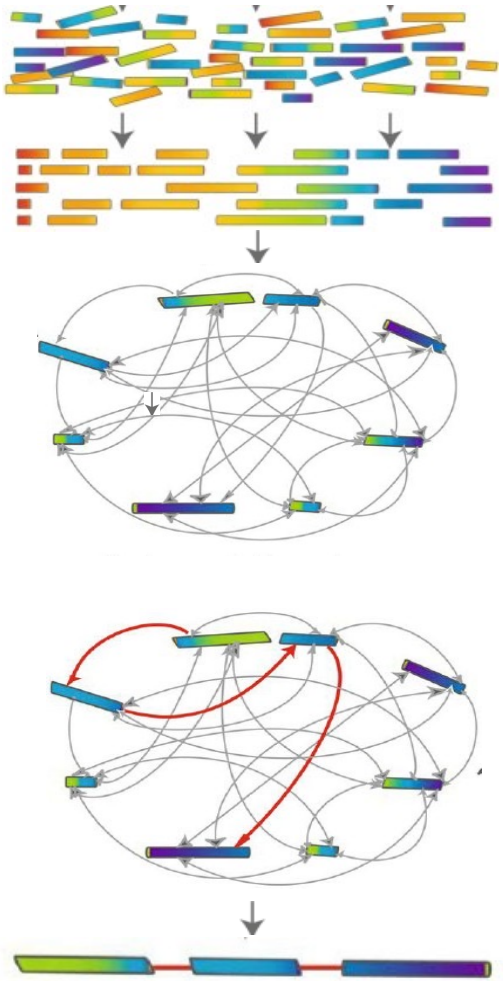
assembly algorithms

Overlap-Layout-Consensus

De Brujin graph

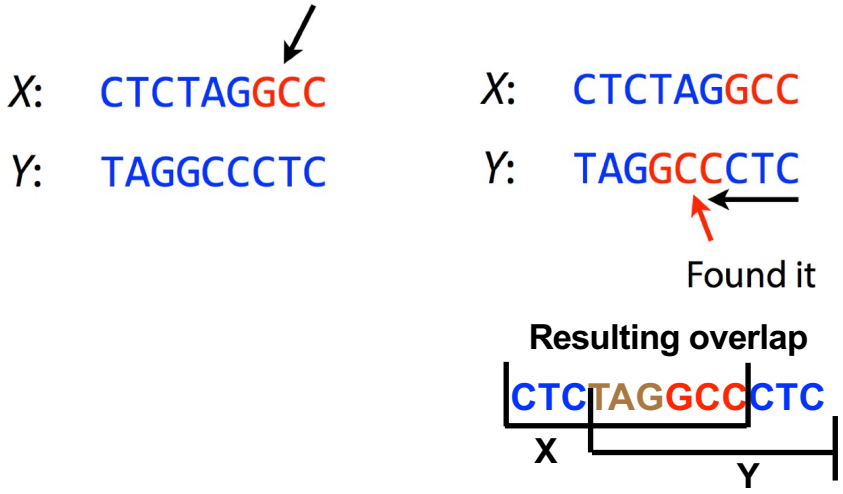
Overlap Layout Consensus - OLC

- Overlap
  - Find all overlaps between reads
  - Build overlap graph: nodes=reads, edges=overlaps
- Layout
  - Simplify graph: errors/SNPs, repeats
  - Define assembly path
- Consensus
  - Align reads along assembly path
  - Consensus bases are called using weighted voting

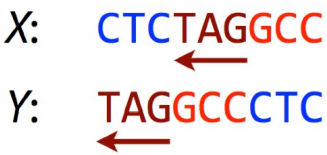


Overlap Layout Consensus - Overlap

Look for this in *Y*, going right-to-left



Extend to left; in this case, we confirm that a length-6 prefix of *Y* matches a suffix of *X*



- We need to do this for every pair of reads

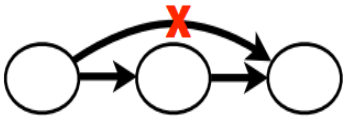
Overlapping is typically the slowest part of assembly

Consider a second-generation sequencing dataset with hundreds of millions or billions of reads!

Overlap Layout Consensus - Layout

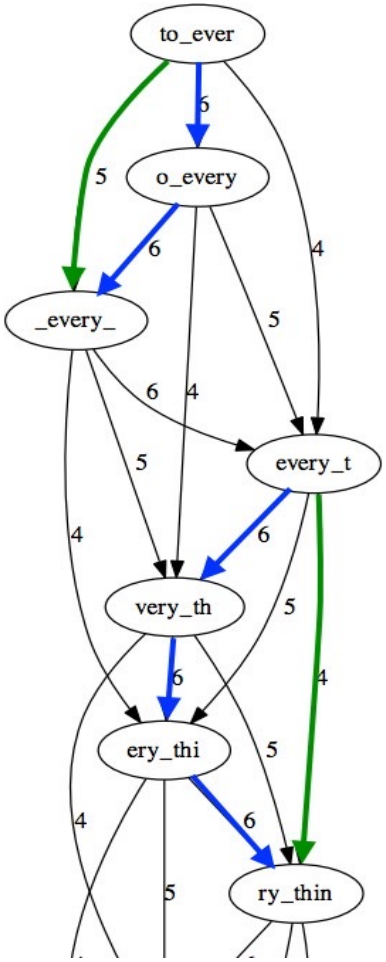
Pop bubbles

- Bubbles are formed with transitively inferable edges
  - Edges can be inferred from others



  - The green ones can be inferred from the blue ones

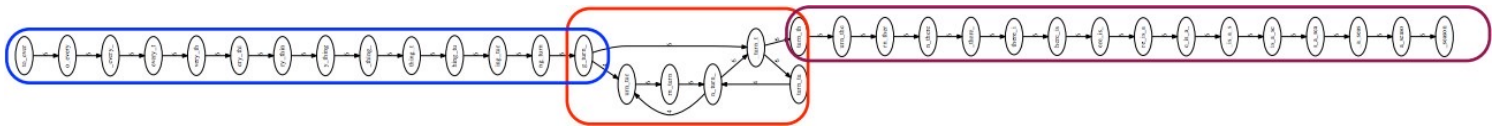
- Popping bubbles collapses small differences (i.e. minor heterozygosity)



Overlap Layout Consensus - Layout

Transverse only unambiguous paths

- Repeats, paralogous genes etc form ambiguous paths
  - **Branches**
- Report only paths corresponding to non-branching stretches
  - Reliable stretches of unique DNA sequences
- Leave ambiguous stretches unresolved



Contig 1 `to_every_thing_turn_`

Contig 2 `turn_there_is_a_season`

Unresolvable repeat

Overlap Layout Consensus - Consensus

At each position, ask: what nucleotide (and/or gap) is here?

Complications: (a) sequencing error, (b) ploidy

Say the true genotype is AG, but we have a high sequencing error rate and only about 6 reads covering the position.

Overlap Layout Consensus – Example

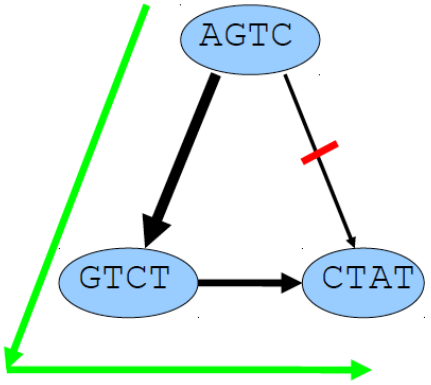
- Find a path which visits each node once
  - Hamiltonian path/cycle is NP-hard (this is bad)
  - solution will be a set of paths which terminate at decision points
- Form a consensus sequences from paths
  - use all the overlap alignments
  - each of these collapsed paths is a contig

Optimal path shown in green

Un-traversed weak overlap in red

Consensus is read by outputting the overlapped nodes along the path

`aGTCTCTat`



Hamiltonian path

a path that visits each node exactly once

NP-hardness

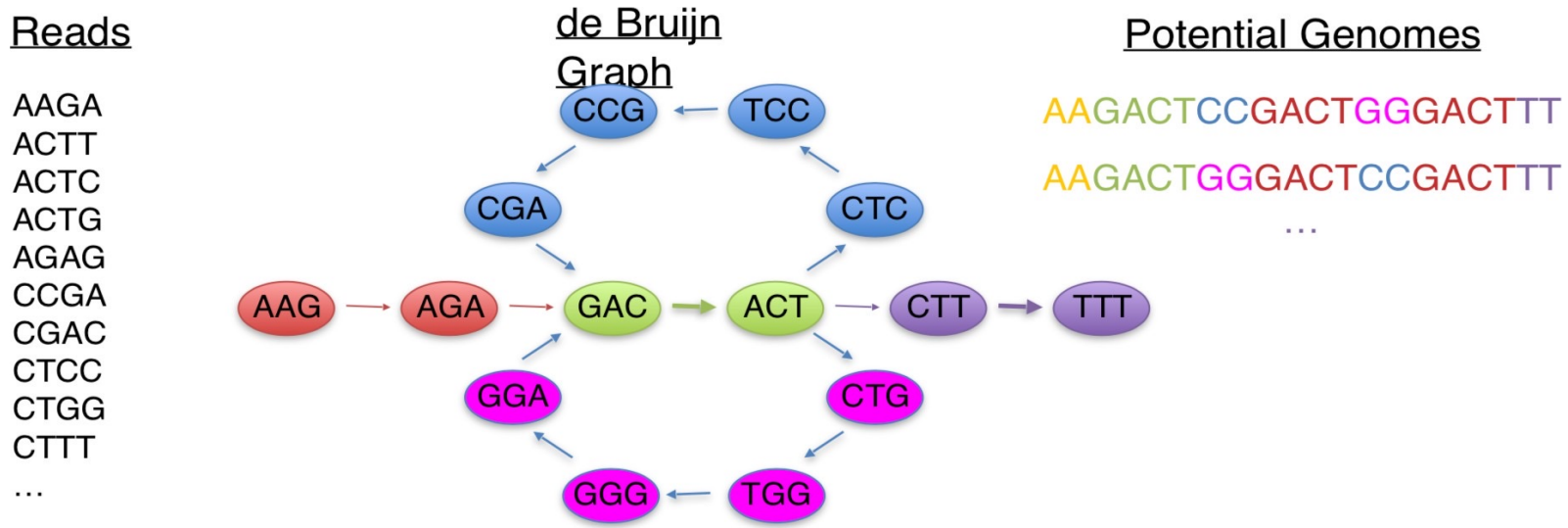
The Hamiltonian path (traveling salesman) optimization problem is NP-hard. Unless P=NP, no polynomial algorithm (in $|V| = n$: number of nodes) exists that guarantees to find an optimal solution.

Number of possible paths: up to $(|V| - 1)!/2$.

Heuristic algorithms generate solutions for large instances of reasonable quality

De Brujin graph - DBG

- Reads are split into K-mers (sequences of length K)
- K-mers represent the edges of the graph
- The nodes are the suffix and the prefix of length K-1 shared by edges



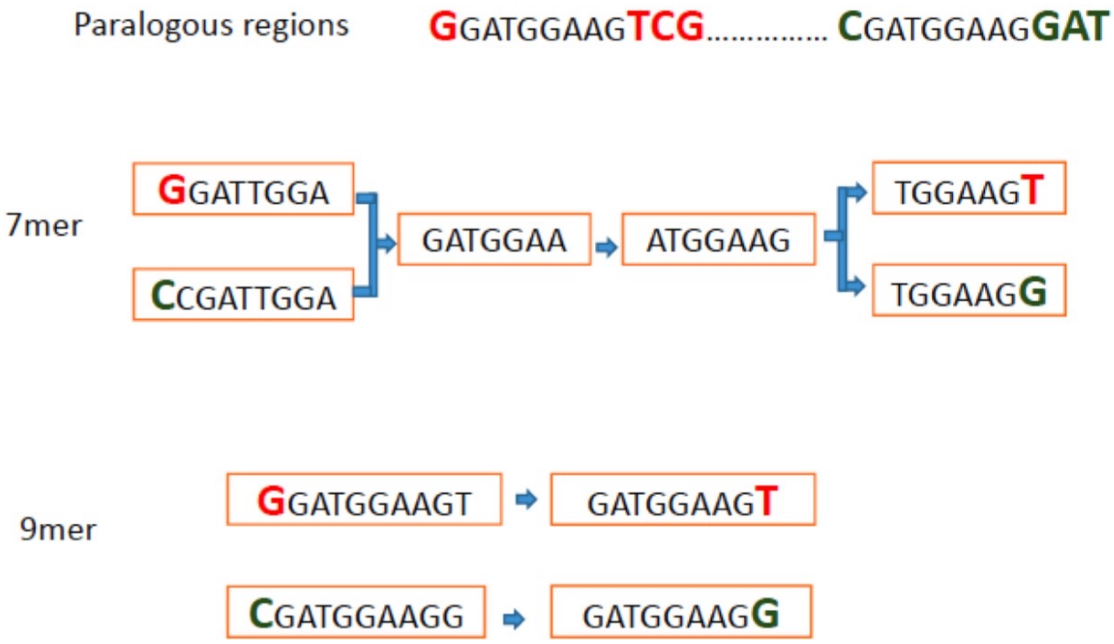
The assembly is an Eulerian path (it contains all the edges)

A directed, connected graph is Eulerian if and only if it has at most 2 semi-balanced nodes and all other nodes are balanced

De Brujin graph - DBG

Impact of kmer length

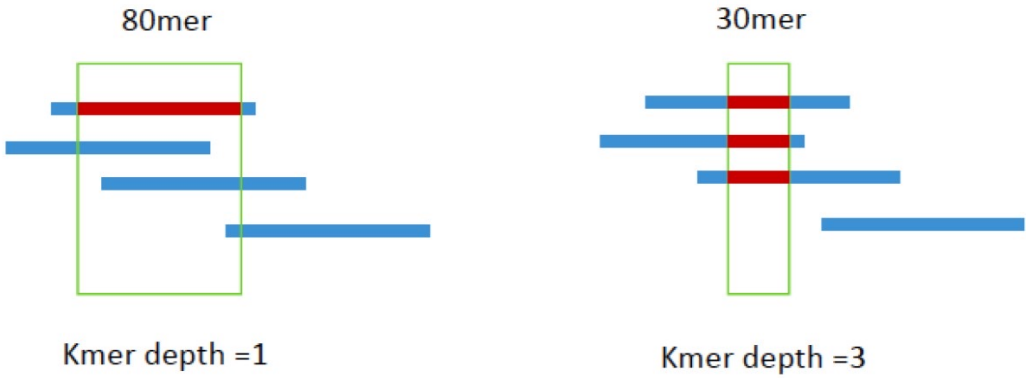
- Short kmers would collapse paralogous regions, and result in more branches



De Brujin graph - DBG

Impact of kmer length

- Longer kmers - lower kmer depth

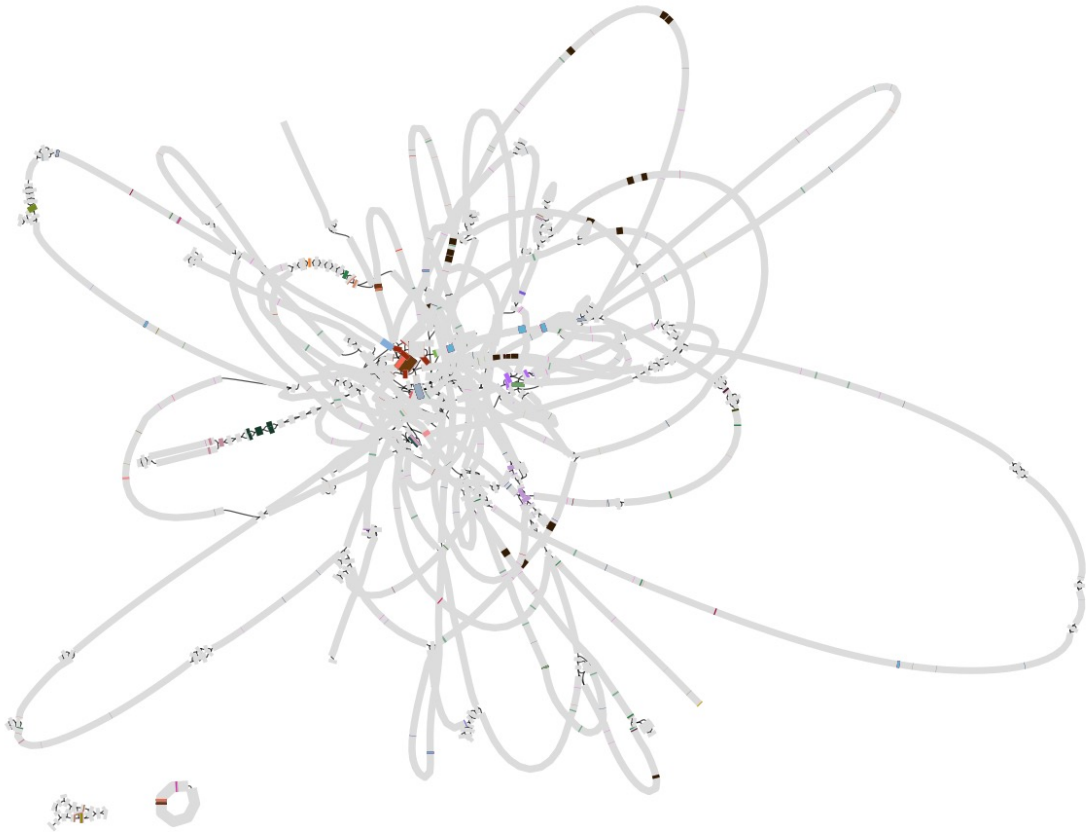


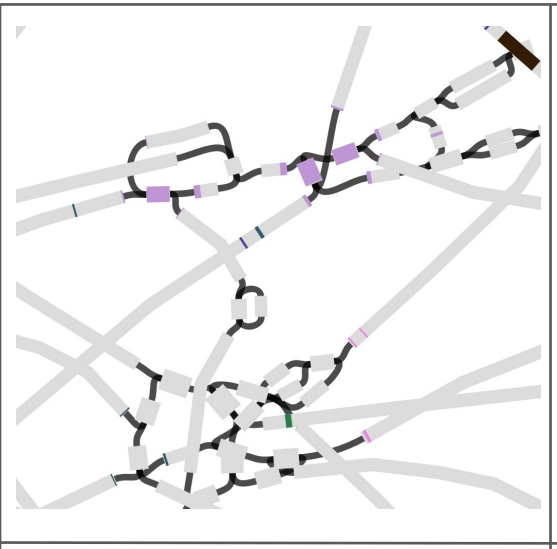
- A selection of k-mers is always used to get the optimal assembly.
- Megahit and Metaspades are DBG assemblers.
- Optimal k-mer length has to be chosen for low- and high- abundance species

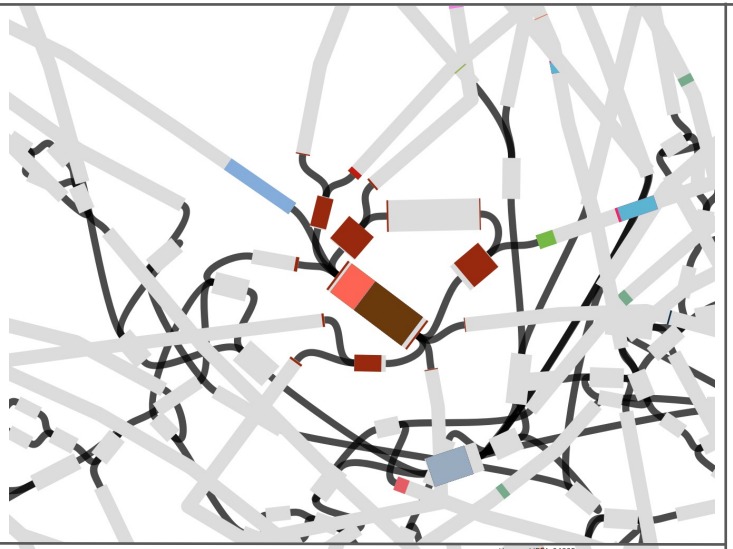
De Brujin graph – Visualising assembly graph

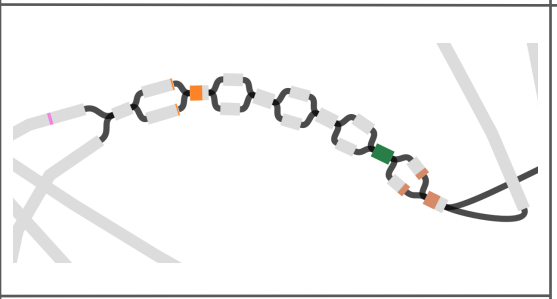
- Tools such as metaSPAdes convert the de Bruijn graph to an assembly graph

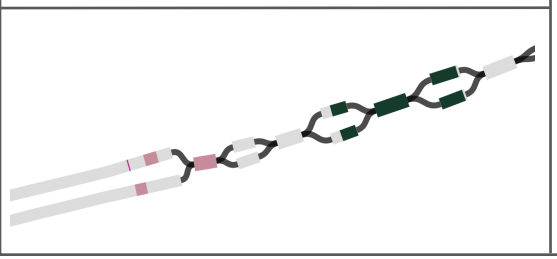
- Bandage allows the visualisation and querying of this graph

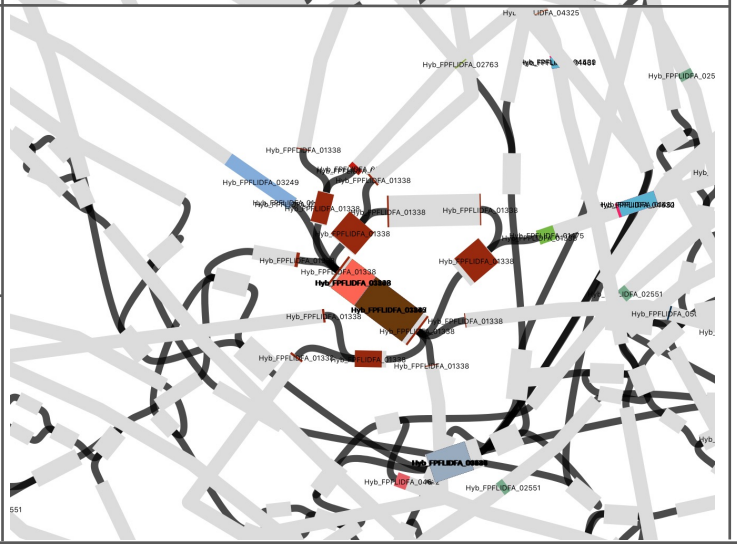










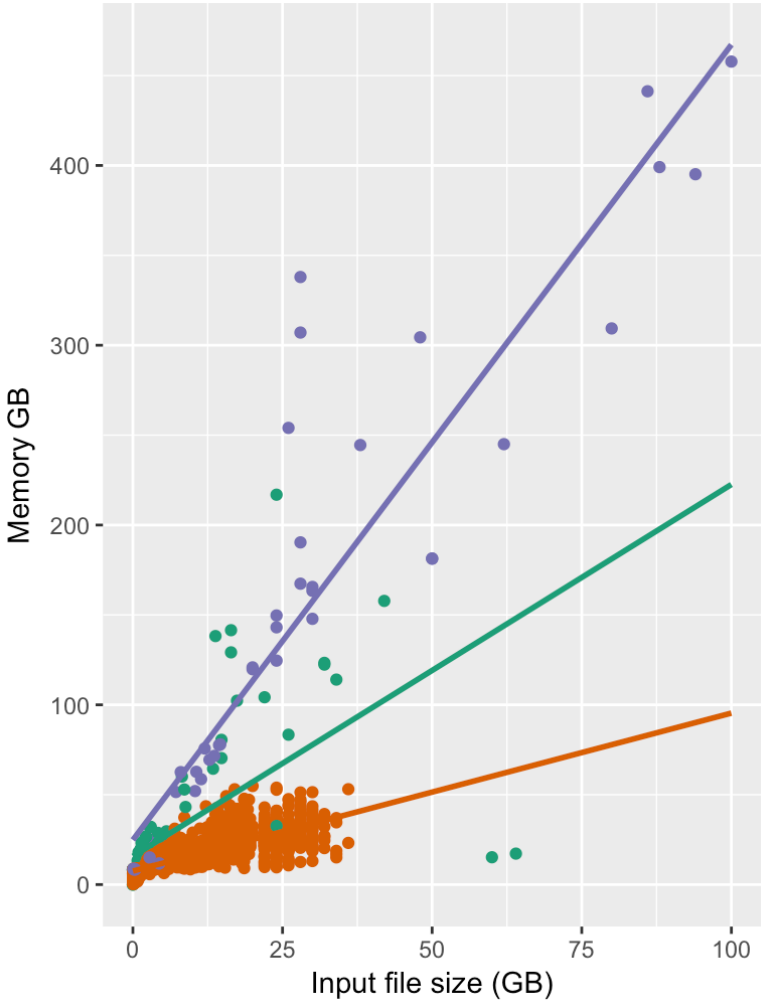


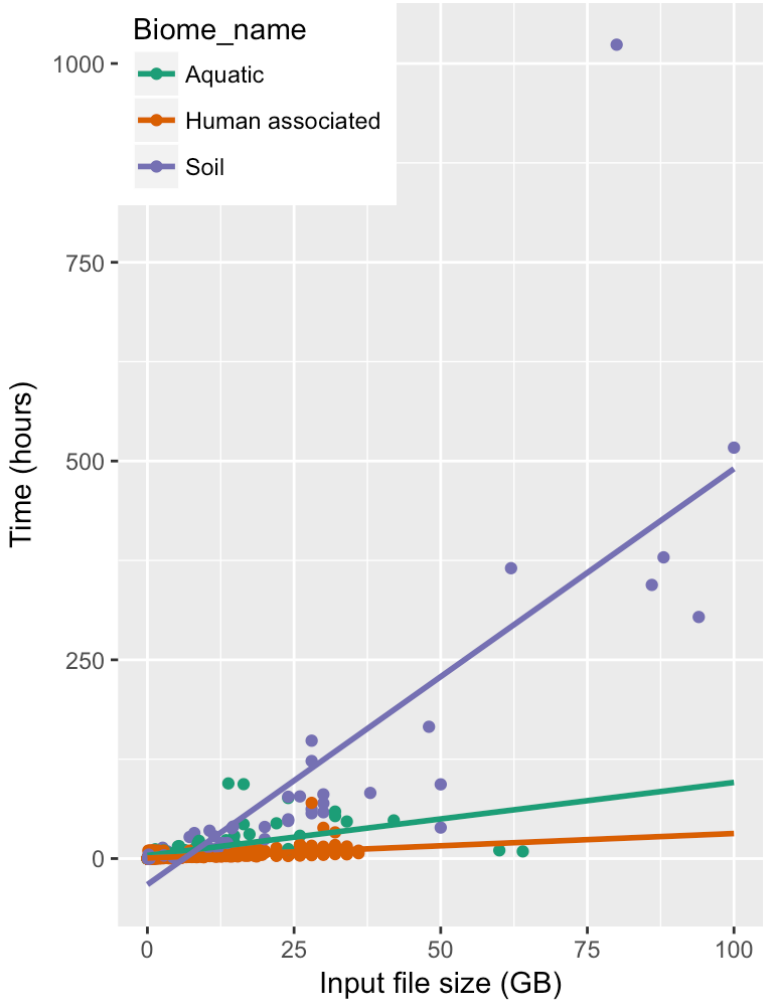
Width of edges proportional to read coverage

Some examples of multiple possible contiguity in which lost proteins are involved.

Conserved domains have multiple matches.

Computational resources limiting assemblies



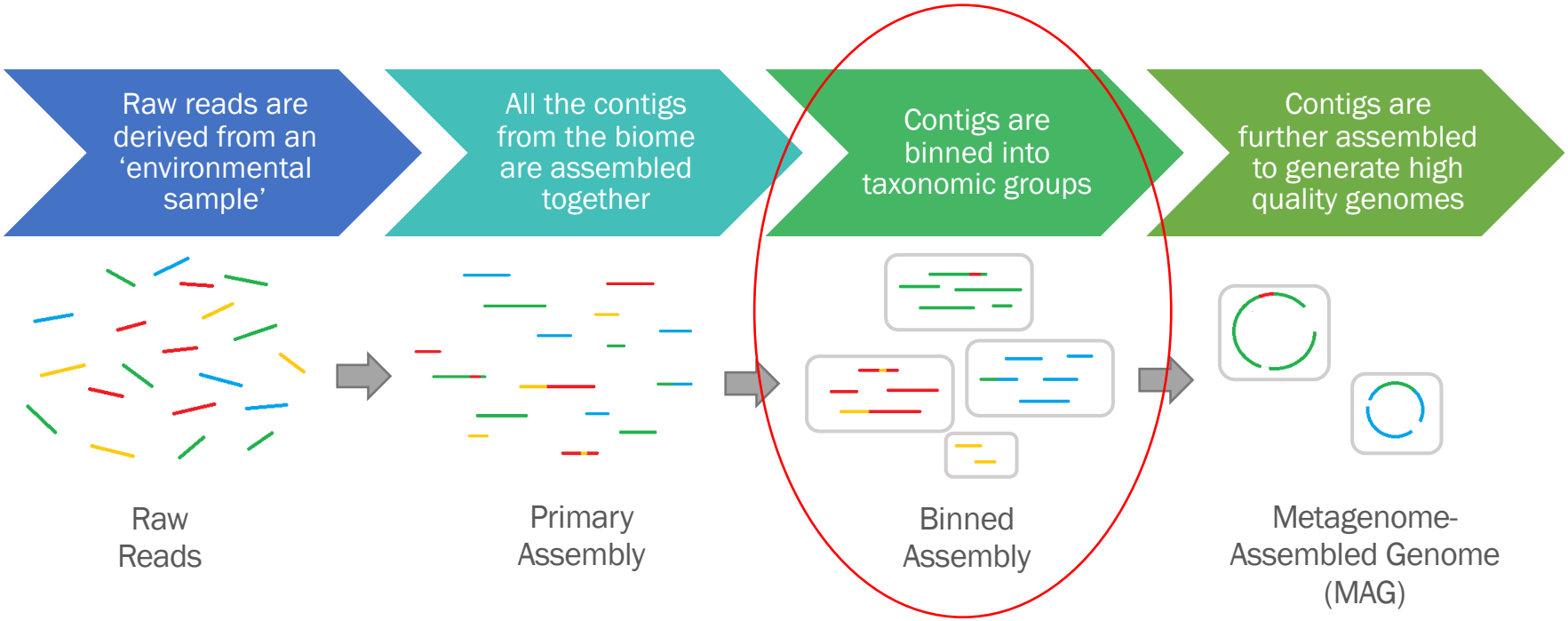


Complications in assemblies

- Different abundance
  - Highly abundant genomes are over-sequenced
    - higher coverage ≠ repeats
  - Low abundant genomes – low coverage kmers are real
- Different relatedness
  - Unique but highly conserved regions in a single genome might be repetitive in a metagenome
- Simple coverage statistics can no longer be used to detect the repeats
- Distinguishing true biological differences from sequencing errors becomes nearly impossible

Ghurye JS. 2016. Yale J Biol Med.

Structuring Metagenomic Studies: Binning contigs



Binning contigs: Two ways

- Taxonomy-independent binning – unsupervised approach
- Taxonomy-dependent binning – supervised approach

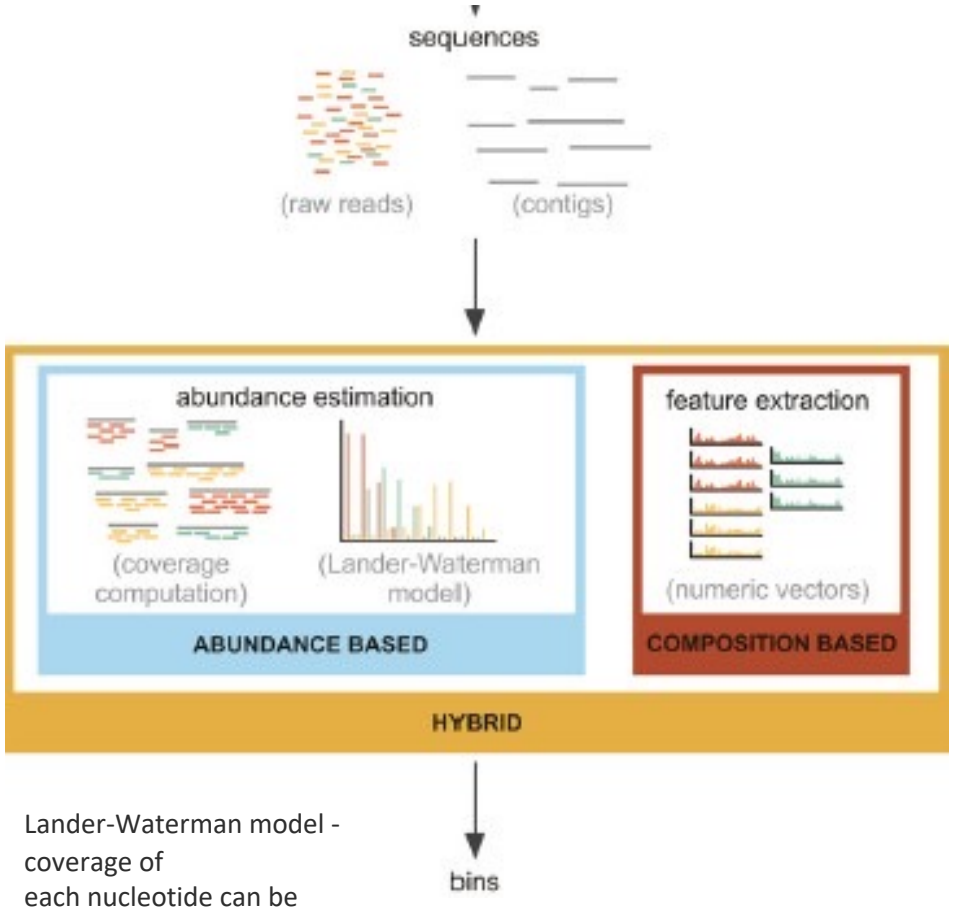
Binning contigs: Taxonomy-independent binning

Composition-based

- Assumes nucleotide composition as closest proxy for species similarity - based on an assumption that the genome composition is unique for each taxon
- Can also be reference agnostic, no assembly
- Only reads information (e.g., GC %, K- mers)
- Fast (no need to align)

Abundance-based

- Assumes sequence abundances as closest proxy for species similarity
- *De novo* assembly for initial contigs
- Abundance of contigs via read mapping
- Binning of contigs
- Concerned with k-mer abudance rather than conent

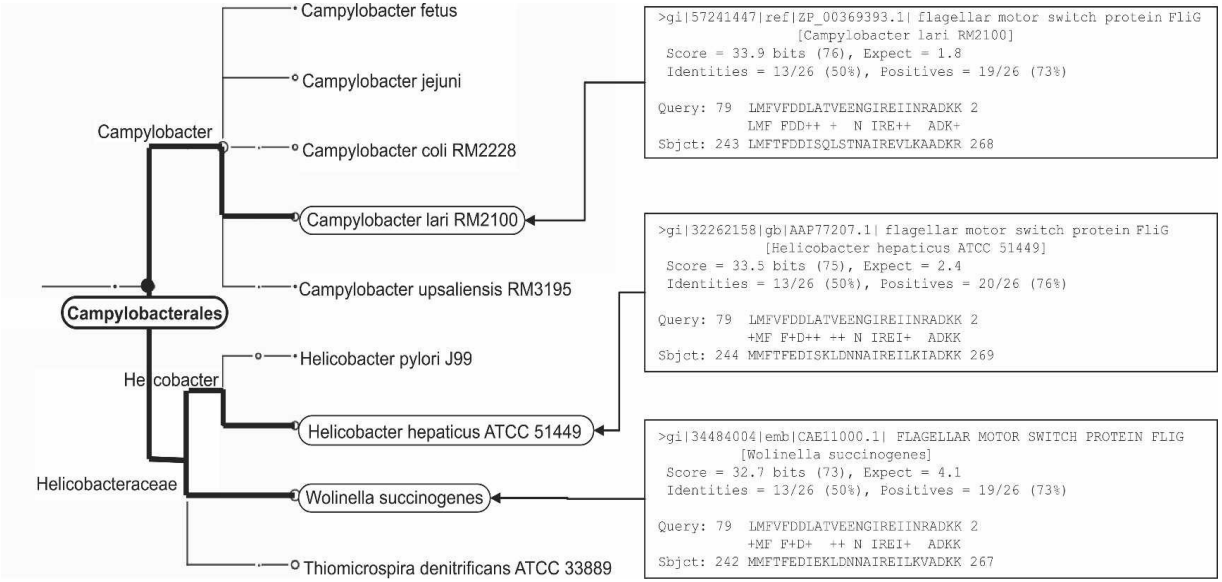


Lander-Waterman model - coverage of each nucleotide can be computed by the application of the Poisson distribution

Taxonomy dependent binning (LCA)

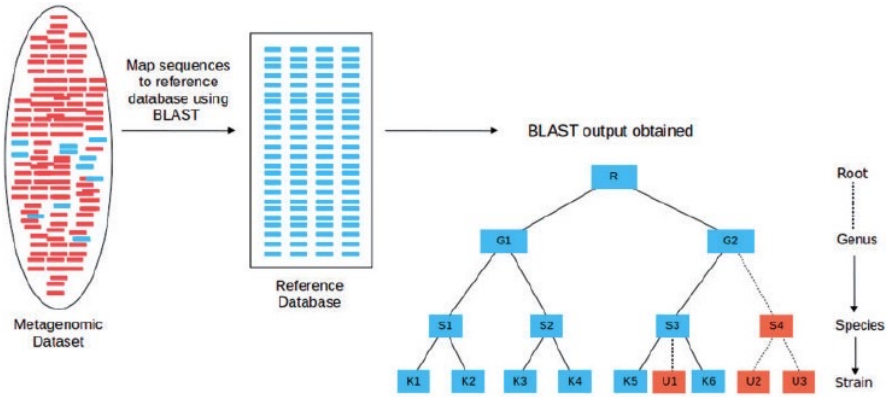
Sequence-based

- Assumes similarity to sequences in a database as closest proxy for species similarity
- Multiple potential hits usually reported (30-50)
- Lowest common ancestors approach (LCA) to resolve ambiguities
- Standard LCA limited in the case of long/multiple ORF contigs (e.g., Eukaryote metagenomes)
- LCA* recently proposed (Hanson et al., Bioinformatics 2016)



Huson et al., Gen. Res., 2007

Taxonomy dependent binning (LCA)

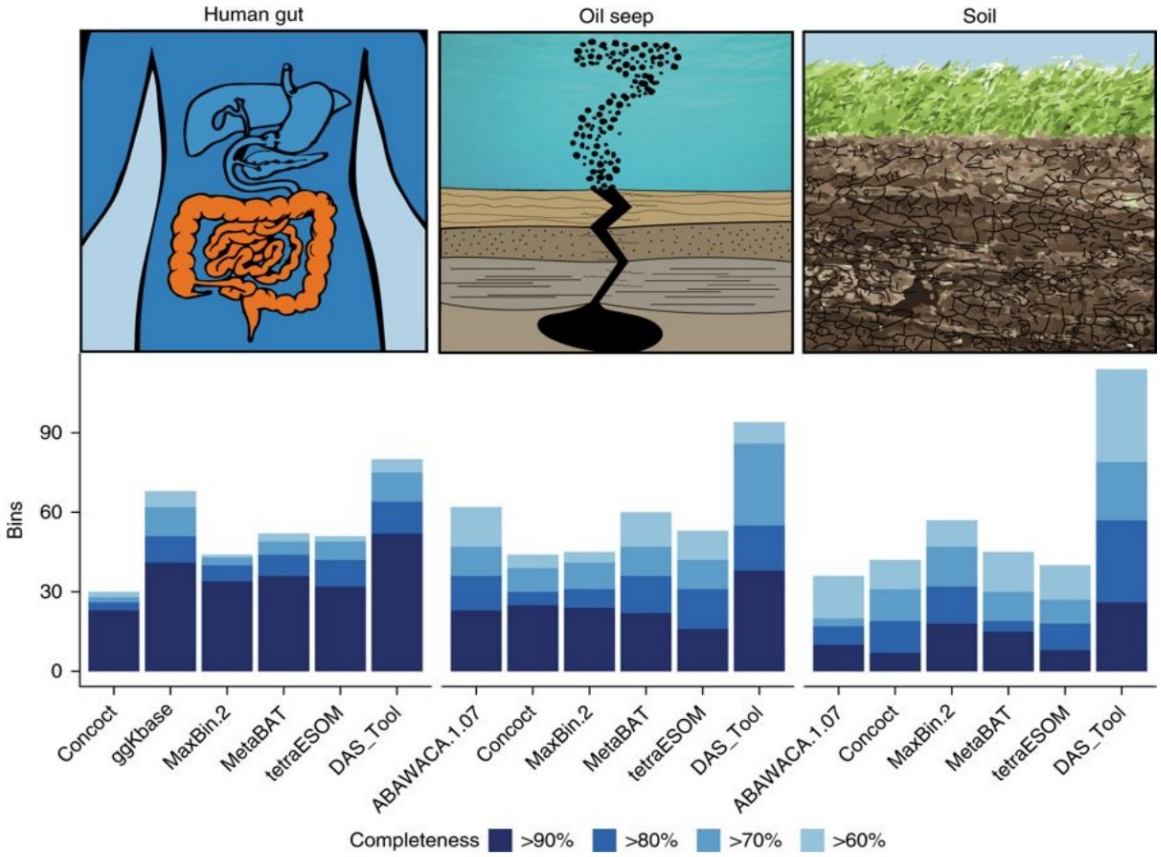


PRO	CONS
Leveraging existing information	Computationally intensive/time consuming (alignment and eventual downstream assembly)
Amenable to shorter reads	Matching bit-scores sometimes inadequate metric
Rare microorganisms identifiable provided sufficient coverage	Problematic with sample from mostly unexplored environments

Reads originate from	Significant BLAST Hits	Assingment Strategies	
		Best BLAST Hit Approach	LCA
K1	K1, K2, K3	K1 (√)	G1 (√)
U1	K5, K6	K5 (X)	S3 (√)
U2 and U3	K5, K6	K5 (X)	S3 (X)

Binning contigs: Selection of methods

- Use multiple binning methods when possible



Sieber CMK et al. 2018.Natural Microbiology

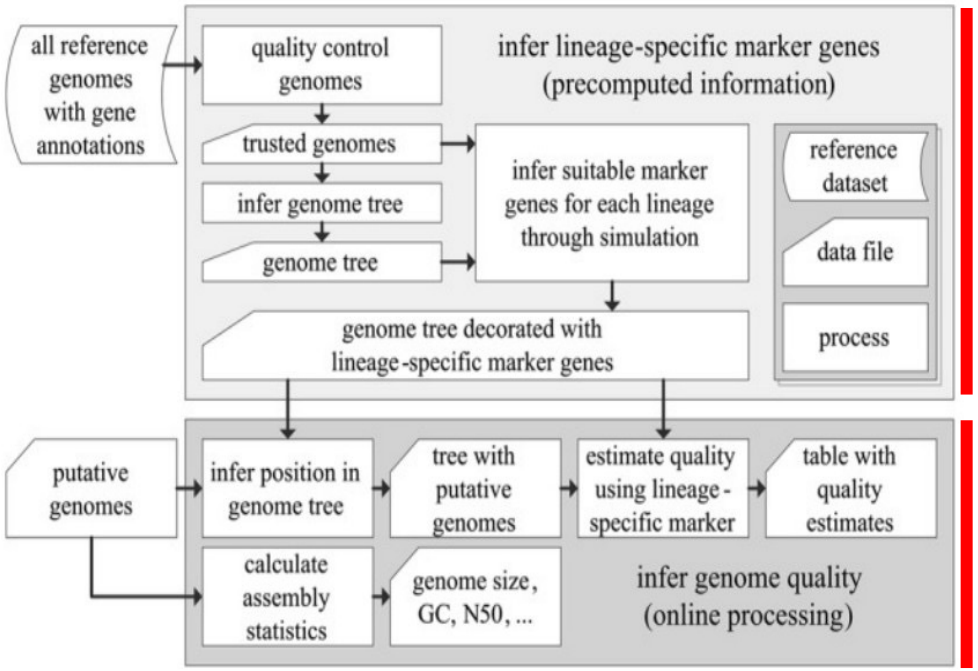
Binning contigs

Assessing the quality of reconstructed genome bins

- Estimate completeness and cleanness
  - The number and copy number of marker genes
  - Universal and lineage specific marker gene sets

- checkM
  - Utilize the set of marker genes specific to the position of a genome within a reference genome tree
  - Can distinguish contamination introduced by multiple strains from that introduced by more divergent taxa: amino acid identity (AAI) between multicopy genes
    - a gene identified as single copy in ≥97% of genomes is considered to be a marker gene
    - often encode essential functions and are frequently organized into operons
    - often also used collocated marker sets

Binning contigs: Assessing quality with CheckM

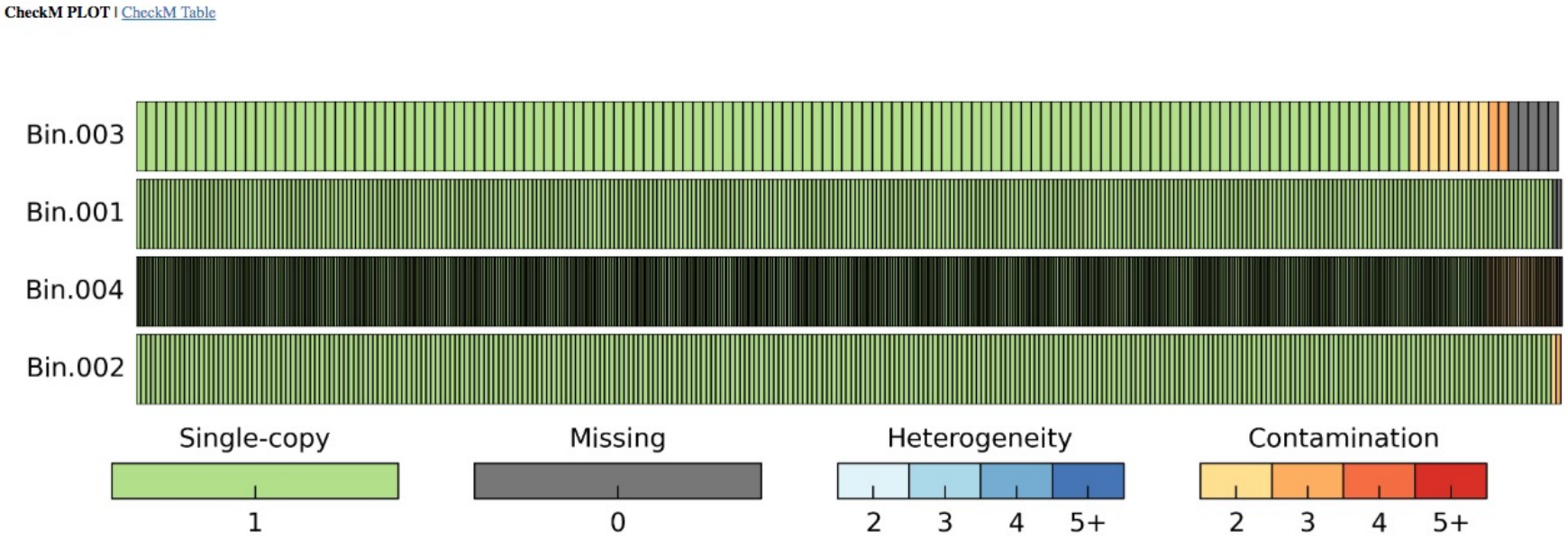


Creating a CheckM database

Assesing your bins against the database

Parks DH et al. 2015. Genome Research

Binning contigs: Assessing quality with CheckM



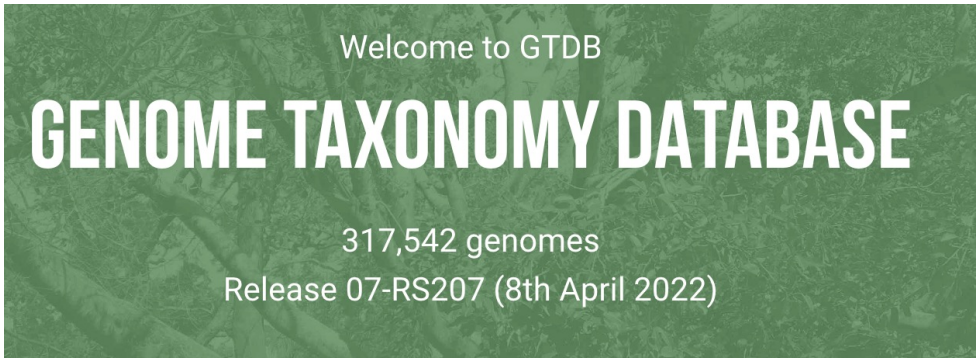
What makes a good binned assembly

- N50
  - minimum contig length that contains 50% of the assembled bases
- Long contigs (subjective)
- Does the assembly contain what you expect
  - MetaQUAST, BLAST
- Assembly Likelihood Evaluation framework (ALE) - https://bioinformaticshome.com/tools/wga/descriptions/ALE-Assembly.html
  - tool can be used to detect errors in metagenomes as well by pinpointing single base errors, indels, genome re-arrangement and chimera
  - without the need of a reference genome

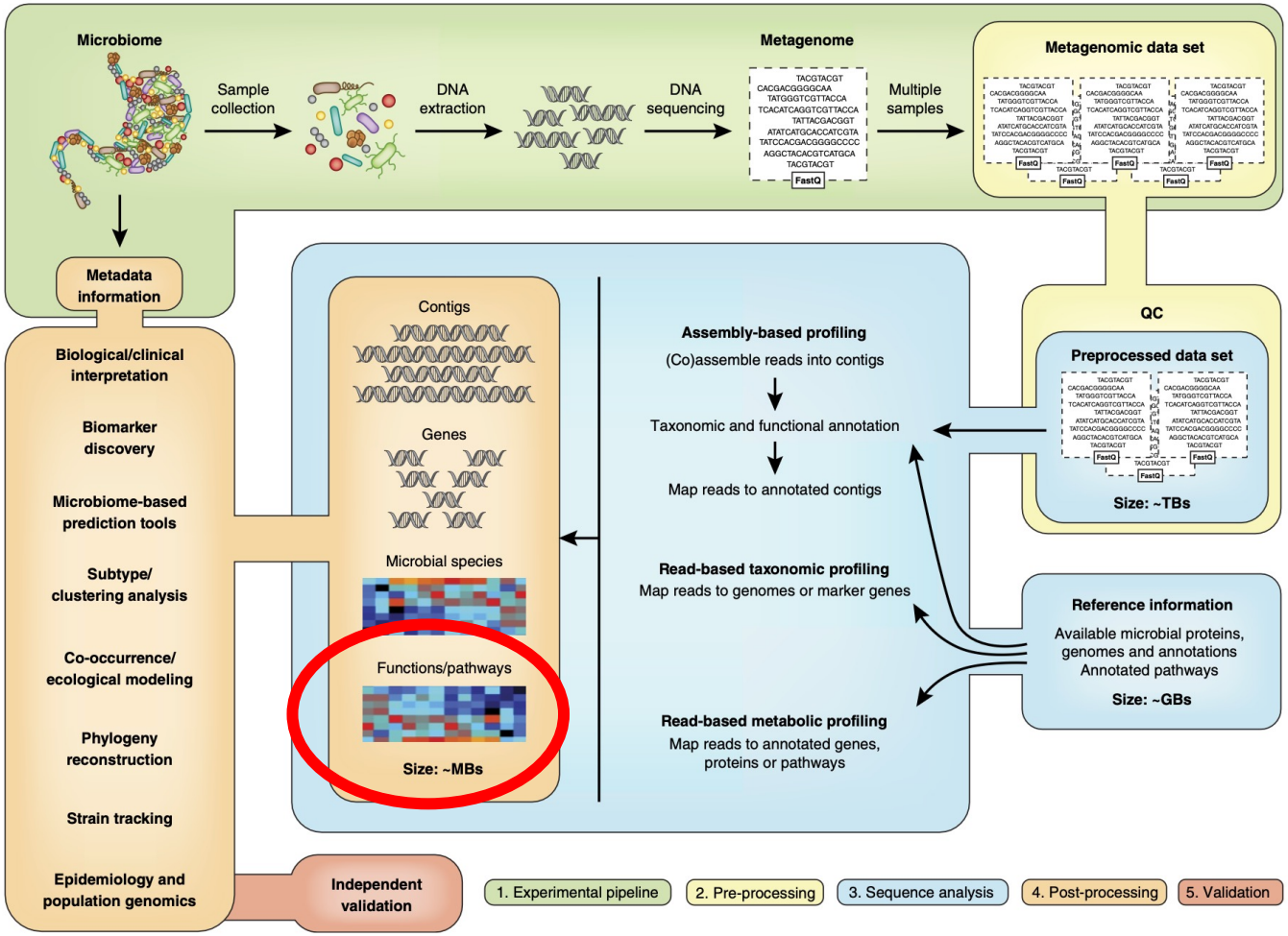
Tools for robust taxonomic classification

MG-RAST (GenBank annotation)
(https://www.mg-rast.org/)

GTDBTk
(https://github.com/Ecogenomics/GTDBTk)
(https://gtdb.ecogenomic.org/)



Overview



3. Metagenome assembly and annotation

Protein prediction: Prodigal

What does Prodigal do?

- **Predicts protein-coding genes**: Prodigal provides fast, accurate protein-coding gene predictions in GFF3, Genbank, or Sequin table format.
- **Handles draft genomes and metagenomes**: Prodigal runs smoothly on finished genomes, draft genomes, and metagenomes.
- **Runs quickly**: Prodigal analyzes the *E. coli K-12* genome in 10 seconds on a modern MacBook Pro.
- **Runs unsupervised**: Prodigal is an unsupervised machine learning algorithm. It does not need to be provided with any training data, and instead automatically learns the properties of the genome from the sequence itself, including genetic code, RBS motif usage, start codon usage, and coding statistics.
- **Handles gaps, scaffolds, and partial genes**: The user can specify how Prodigal should deal with gaps and has numerous options for allowing or forbidding genes to run into or span gaps.
- **Identifies translation initiation sites**: Prodigal predicts the correct translation initiation site for most genes, and can output information about every potential start site in the genome, including confidence score, RBS motif, and much more.
- **Outputs detailed summary statistics for each genome**: Prodigal makes available many statistics for each genome, including contig length, gene length, GC content, GC skew, RBS motifs used, and start and stop codon usage.

Protein prediction: Prokka

Prokka trustworthy databases, moving from medium-sized to domain-specific databases in a hierarchical manner.

1.An optional **user-provided set of annotated proteins**. These are expected to be trustworthy curated datasets and will be used as the primary source of annotation. They are searched using BLAST+ blastp (Camacho *et al.* , 2009).

2.All bacterial **proteins in UniProt** (Apweiler *et al.* , 2004) that have real protein or transcript evidence and are not a fragment. This is ~16 000 proteins, and typically covers >50% of the core genes in most genomes. BLAST+ is used for the search.

3.All proteins from **finished bacterial genomes in RefSeq** for a specified genus. This captures domain-specific naming, and the databases vary in size and quality, depending on the popularity of the genus. BLAST+ is used for this and is optional.

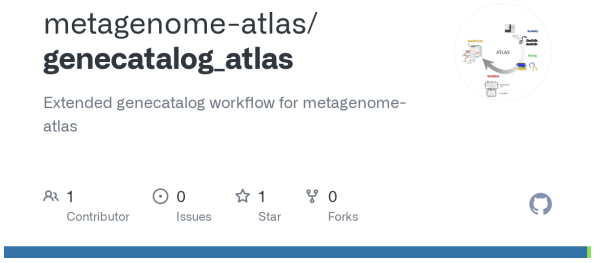
4.A series of hidden Markov model profile databases, including **Pfam** (Punta *et al.* , 2012) and **TIGRFAMs** (Haft *et al.* , 2013). This is performed using hmmscan from the HMMER 3.1 package (Eddy, 2011).

5.If no matches can be found, label as ‘hypothetical protein’.

Annotation

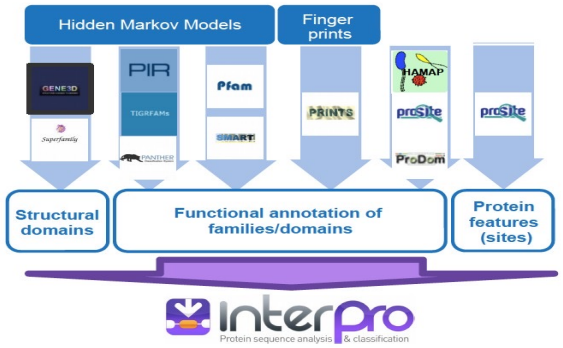
- After binning/reads assembly (preferred)
  - Large contigs (30,000+ bp) expected to have optimal results
  - Genome annotation tools used quite successfully

A lot of pipelines do it automatically as a next step of the workflow:

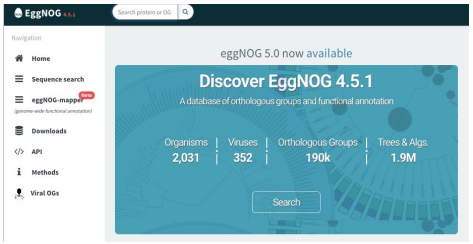


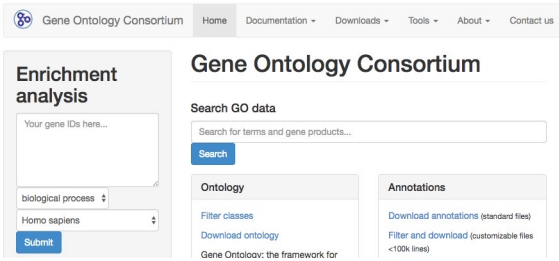
Annotation

Annotation of protein domains (Interpro scan)

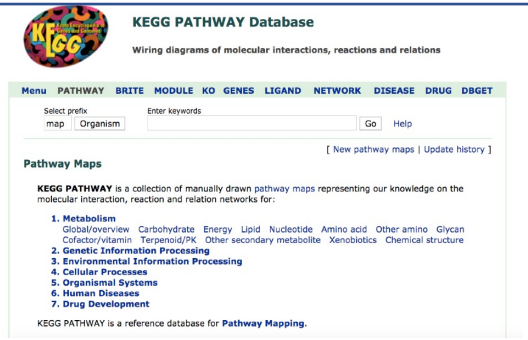


Orthology based gene funtion prediction (EggNOG, GO Consotrium, OMA GO annotation, Pannzer2)



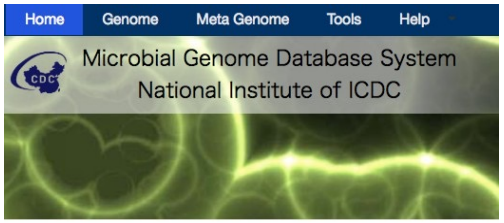


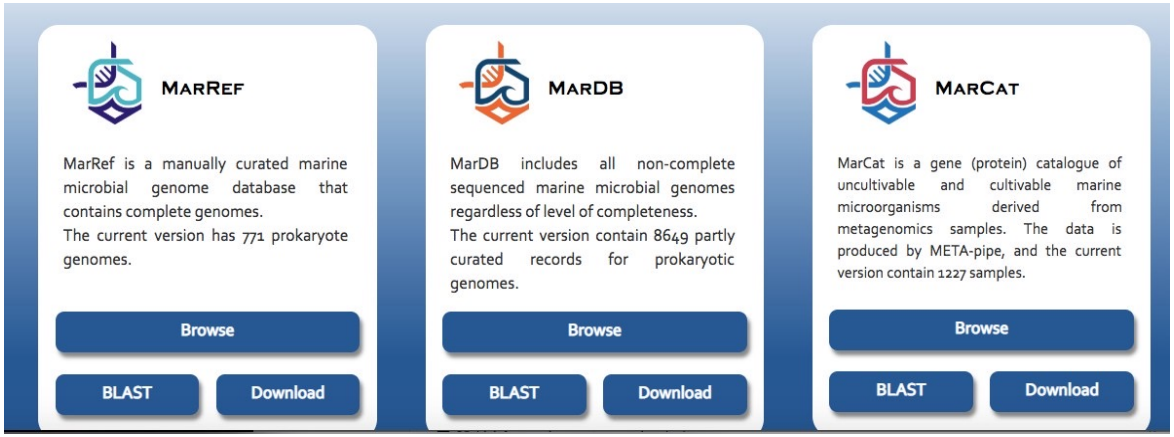
Metabolic pathway prediction (KEGG)



Annotation

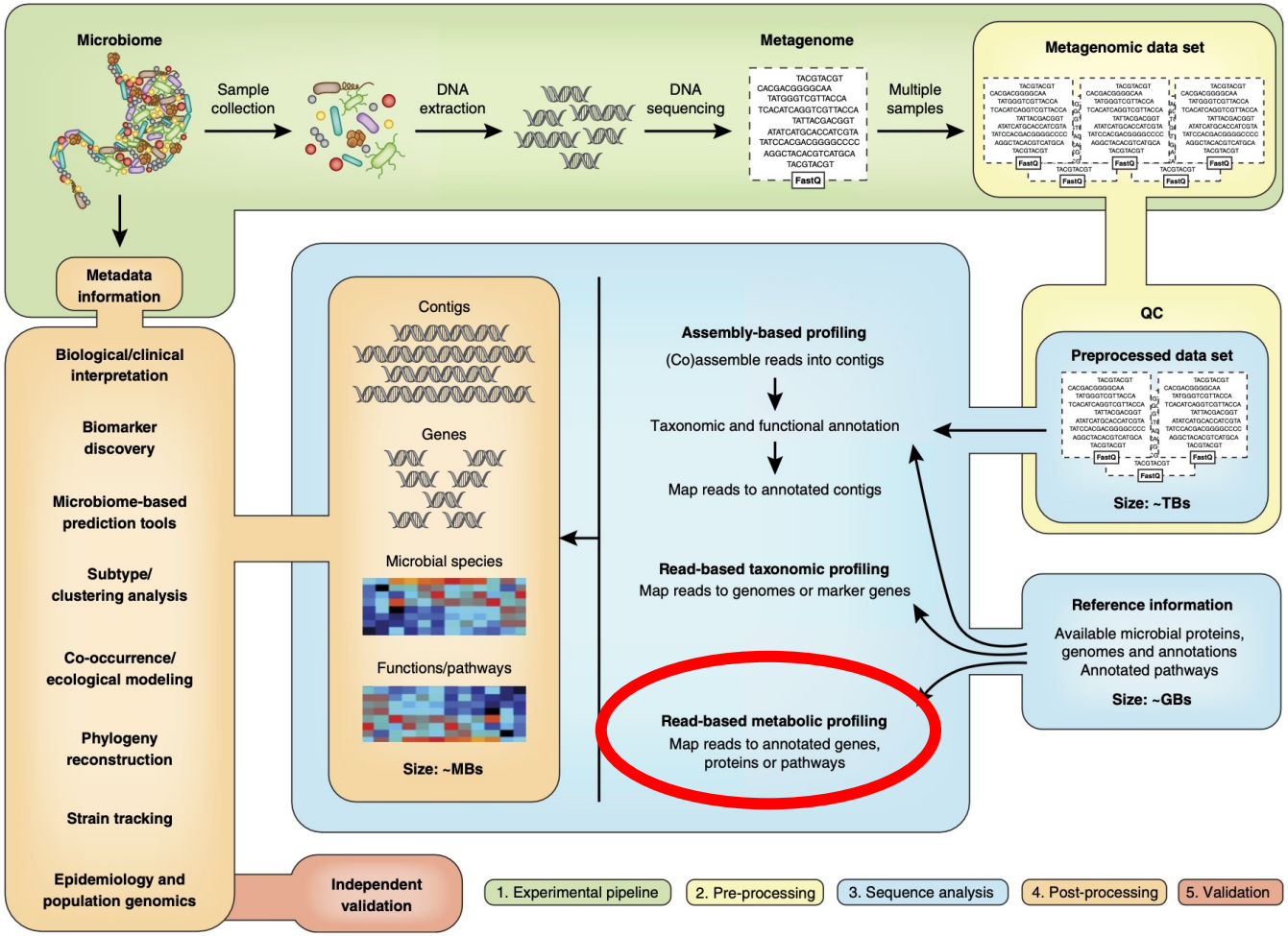
Using ecosystem-specific databases for annotation and taxonomy





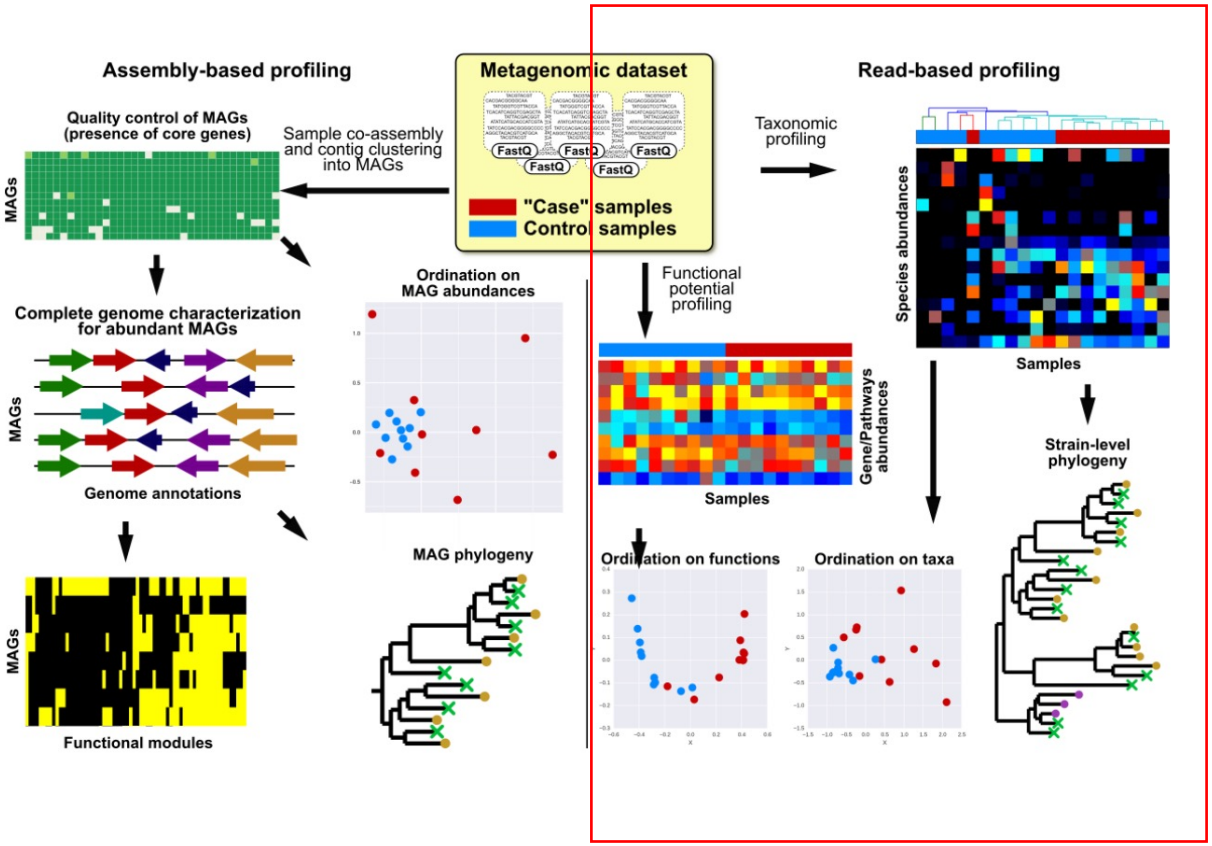
Microbial Genome Database

Overview



4. Assembly – free taxonomic profiling and annotation

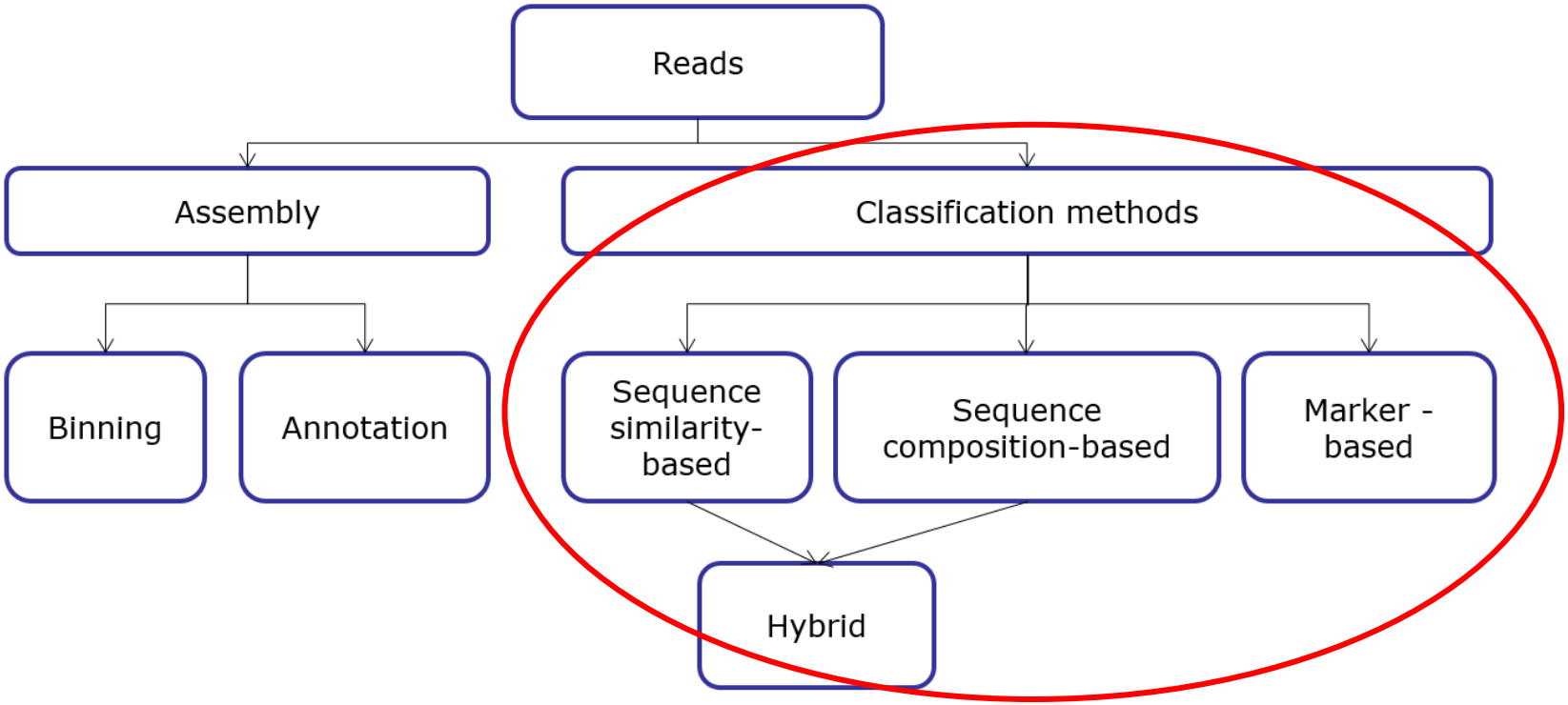
Assembly-free approach



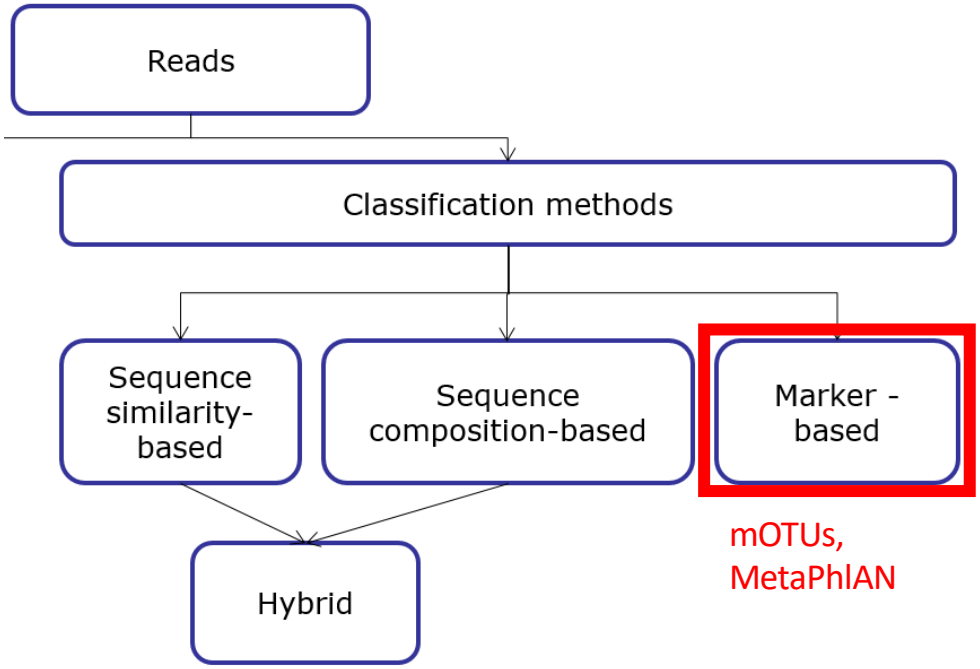
Why?

- speed up computation – assembly methods take a long time
- make it possible to profile low-abundance organisms that cannot be assembled de novo – but database dependent
- might have improved sensitivity
- can answer questions about presence/absence

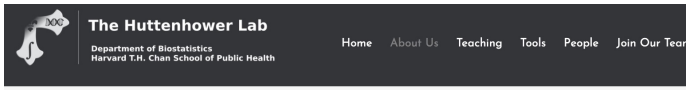
Assembly-free approach: taxonomy



Assembly-free approach: taxonomy



mOTUs, MetaPhlAN

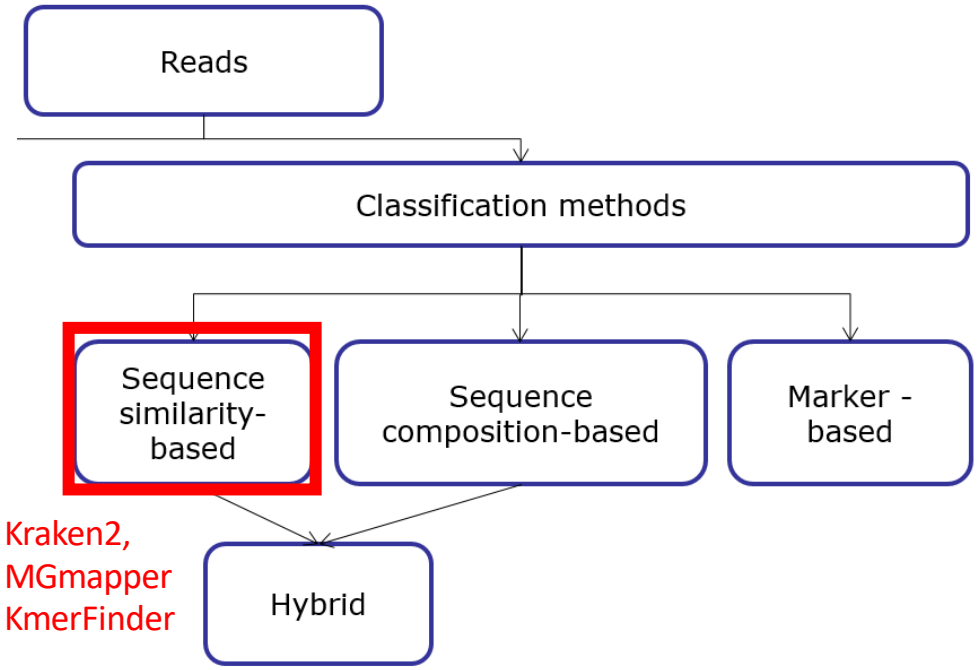


MetaPhlAn 3.0

(http://huttenhower.sph.harvard.edu/metaphlan/)

- 2,887 genomes available from the Integrated Microbial Genomes (IMG) system – 2 million potential markers

Assembly-free approach: taxonomy



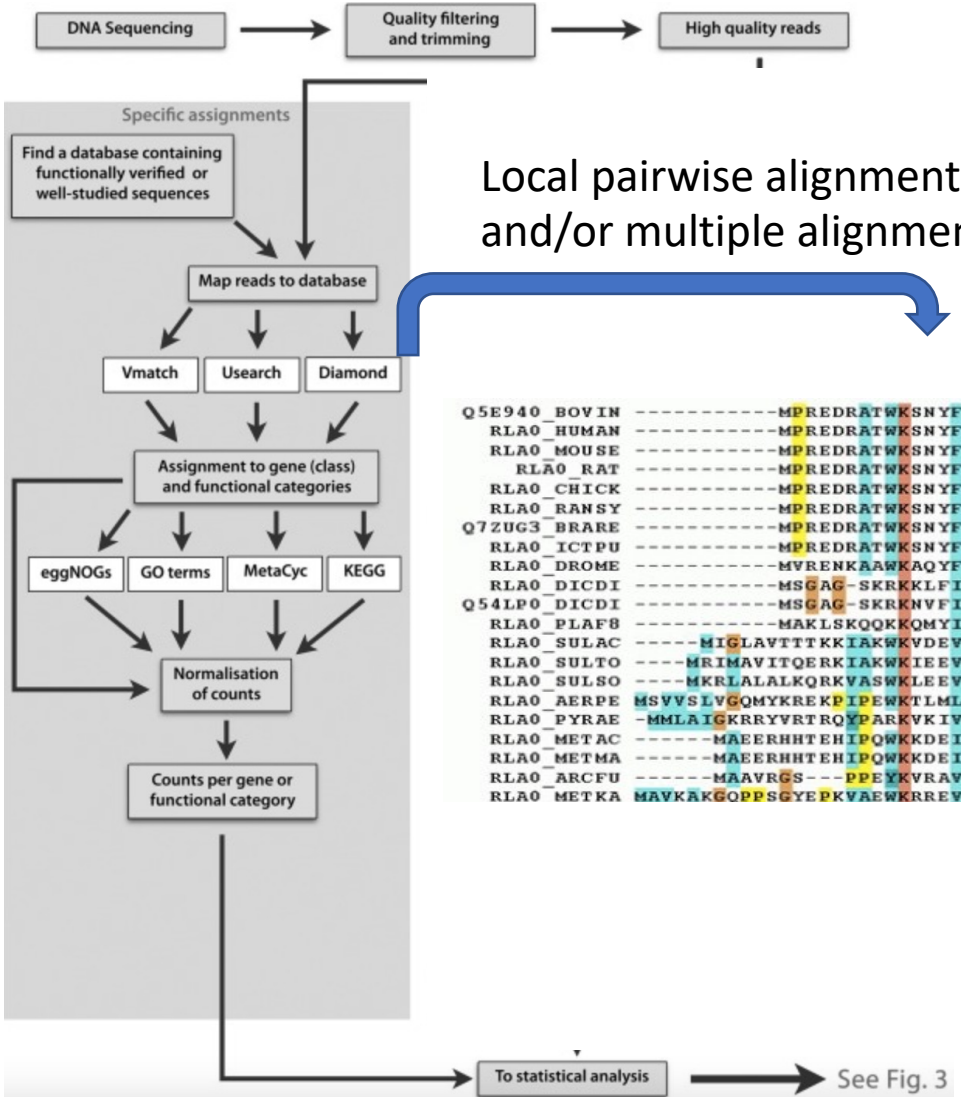
Kraken2,
MGmapper
KmerFinder

(**https://ccb.jhu.edu/software/kraken2/**)

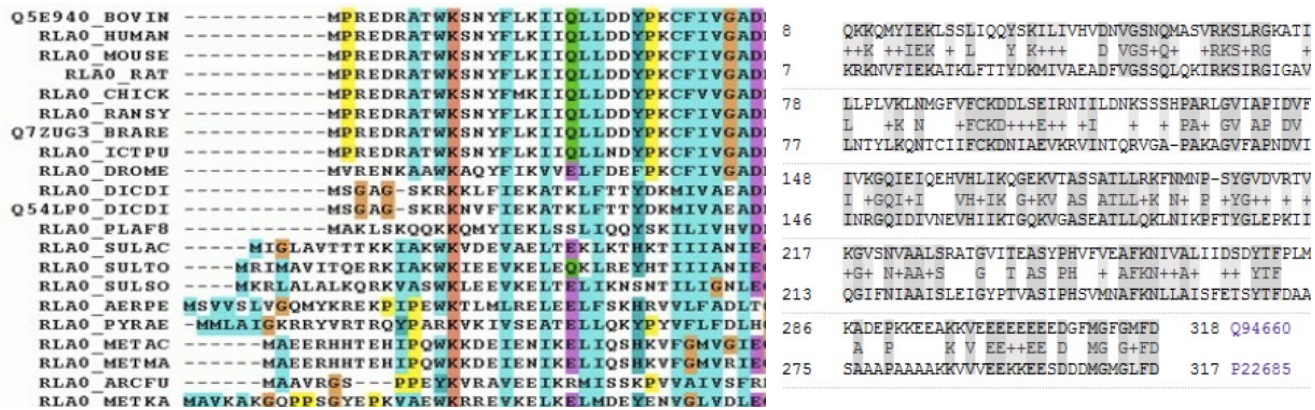
KmerFinder 3.2

(https://cge.cbs.dtu.dk/services/KmerFinder/)

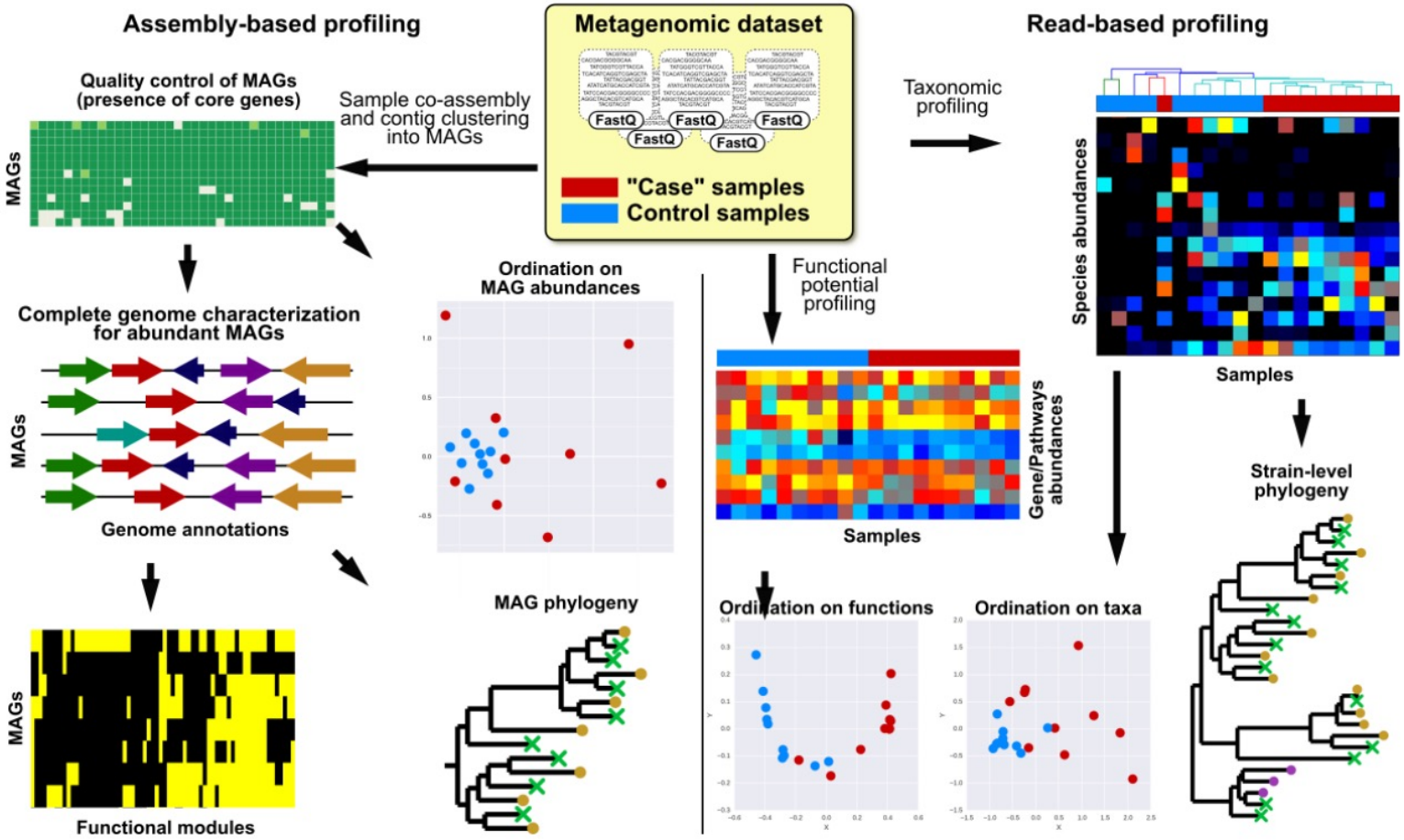
Assembly-free approach: function



Local pairwise alignment
and/or multiple alignment

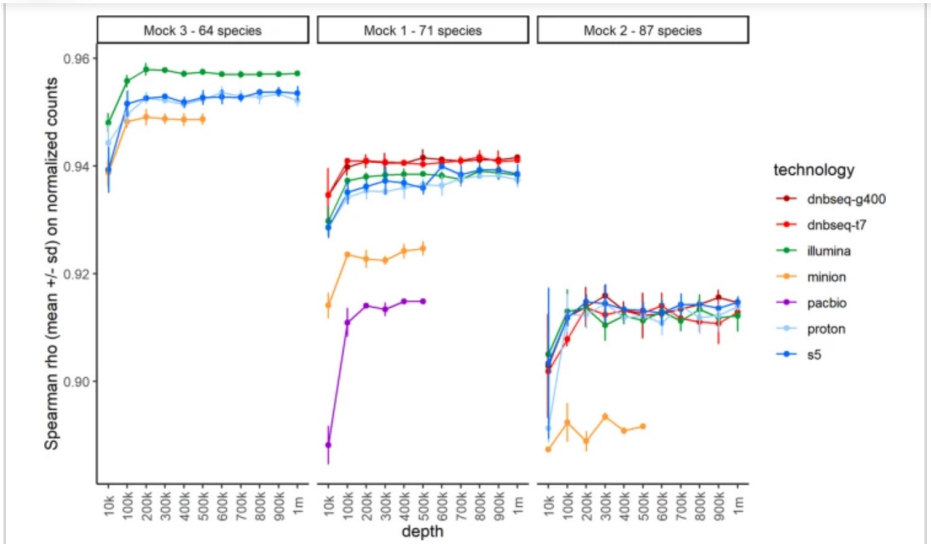


Conclusion


Assembly-based profiling
Metagenomic dataset
Read-based profiling
Quality control of MAGs (presence of core genes)
Sample co-assembly and contig clustering into MAGs
FastQ
"Case" samples
Control samples
Taxonomic profiling
MAGs
Species abundances
Samples
Ordination on MAG abundances
Functional potential profiling
Complete genome characterization for abundant MAGs
Gene/Pathways abundances
Strain-level phylogeny
Genome annotations
MAG phylogeny
Ordination on functions
Ordination on taxa
Functional modules

Illumina/ONT/Pacbio – Benchmarking Nov 2022

Benchmarking second and third-generation sequencing platforms for microbial metagenomics



Sequencer	Ion Proton P1 (spades)	Ion S5 (spades)	Illumina HiSeq 3000 (spades)	DNBSeq G400 (spades)	DNBSeq T7 (spades)	ONT MinION R9 (metaflye)	PacBio Sequel II (metaflye)
Nb Reads (M)	20	20	2×10	2×10	2×10	0.696	0.524
Nb Contigs	45,510	43,879	40,147	44,887	44,603	1,283	**437**
Largest Contig (bp)	384,996	794,907	1,599,668	1,063,396	1,002,925	4,324,150	**7,147,004**
N50 (bp)	7,847	9,089	13,707	8,519	8,184	759,940	**2,013,697**
Genome Fraction(%)	54.767	55.257	61.897	49.397	47.365	44.955	**68.197**
Mismatches per 100kbps	83.29	89.12	47.55	77.22	107.52	339.99	**18.3**
Indels Per 100kbps	77.8	50.03	3.53	3.23	3.67	764.45	11.76
Fully Unaligned Contigs	1,497	1,339	975	735	1,368	231	**6**
Fully Unaligned Length (bp)	900,150	821,545	620,805	426,856	711,992	6,279,694	**134,713**
NB full genome*	5	5	12	7	7	22	**36**

https://www.nature.com/articles/s41597-022-01762-z

Bash commands cheat sheet

pwd
– show the path to the current directory, check where you are
cd <dir>
– go into this directory
cd ..
– exit this directory/go to the previous directory
ls
– check the contents of the directory
ls -alt
– check the contents of the directory but also file sizes/date of creation/last author
cat <file>
– show the contents of the file
head <file>
– show only the 10 first lines of a file
tail <file>
– show only the 10 last lines of a file
grep <pattern> <file>
– find a pattern/phrase/word in a file
cp <file1> <file2>
– copy one file into another file
mv <file1 <file2>
– rename a file

Tutorial

- Time to get some work done! Lets go back to https://github.com/zajacn/metagenomics_course_FGCZ