

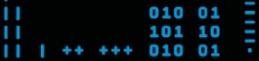
10  
01  
101101 1  
010 0  
0101 10010 01  
101 10  
010 0101 1  
10  
01 1

f g c z

# Shotgun metagenomics

Dr. Natalia Zajac

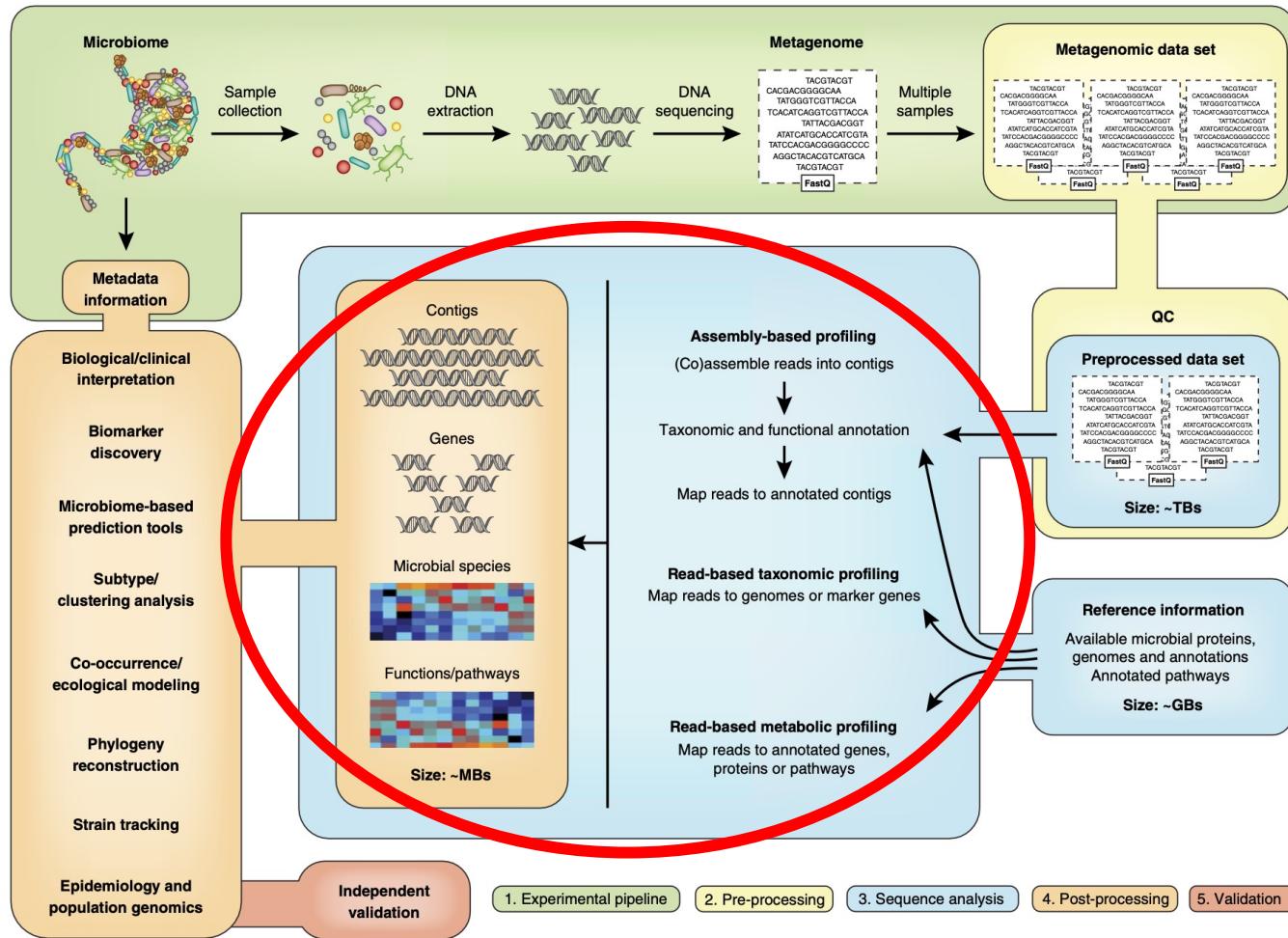
Metagenomics, 03.2023

10  
01  
010  
0101101  
1  
010  
10

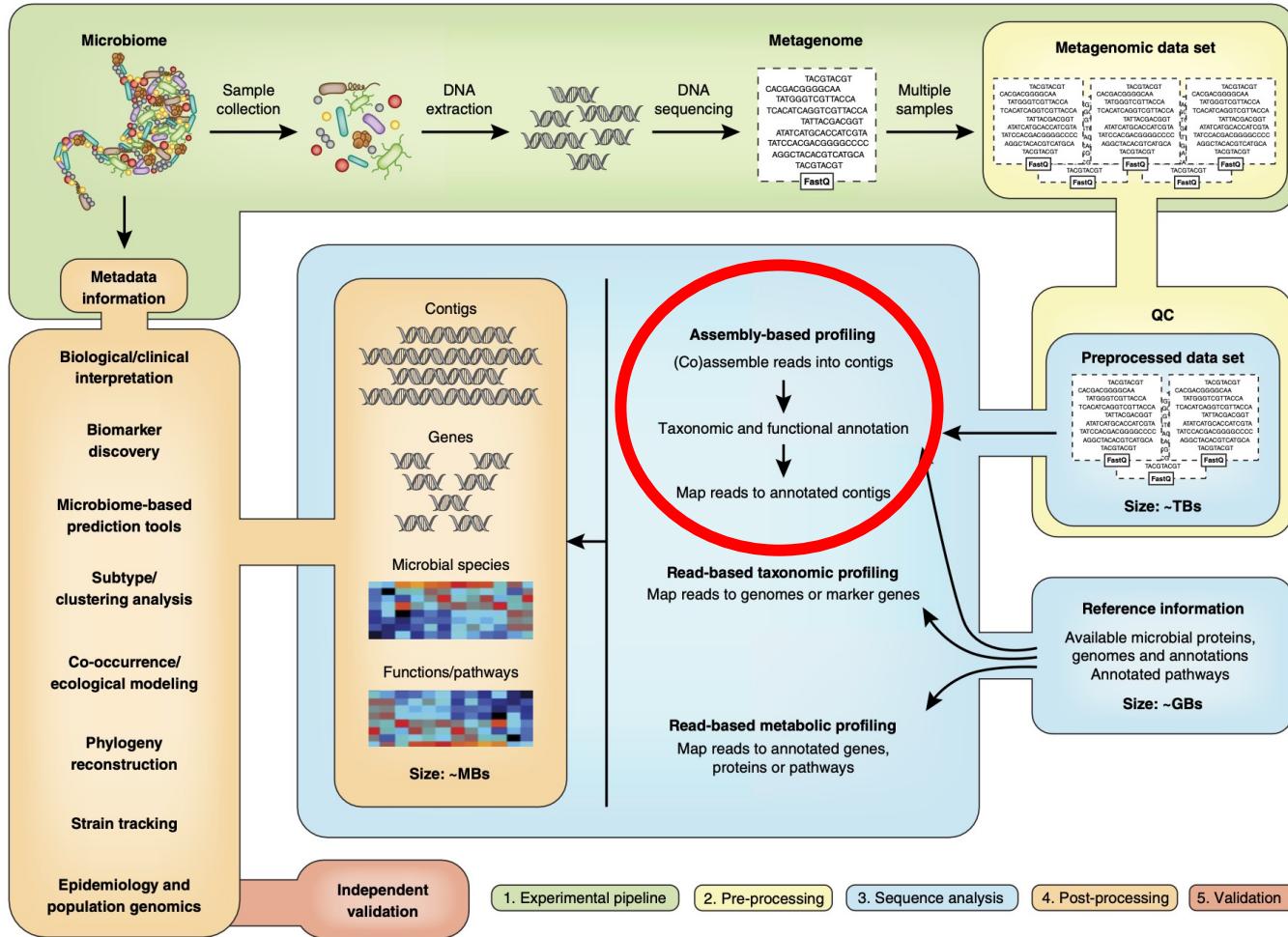
f g c z

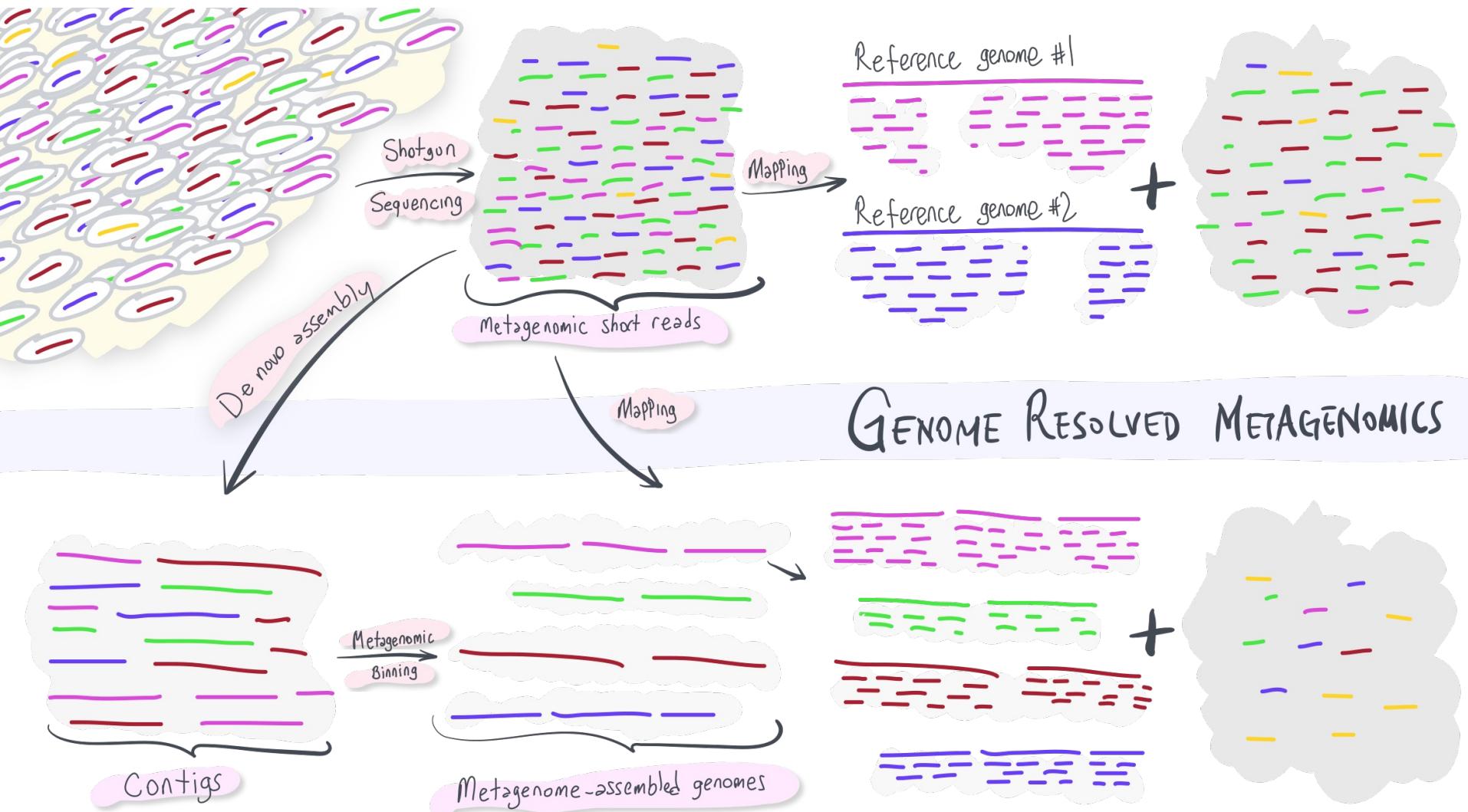
01  
1  
10  
01

# Overview

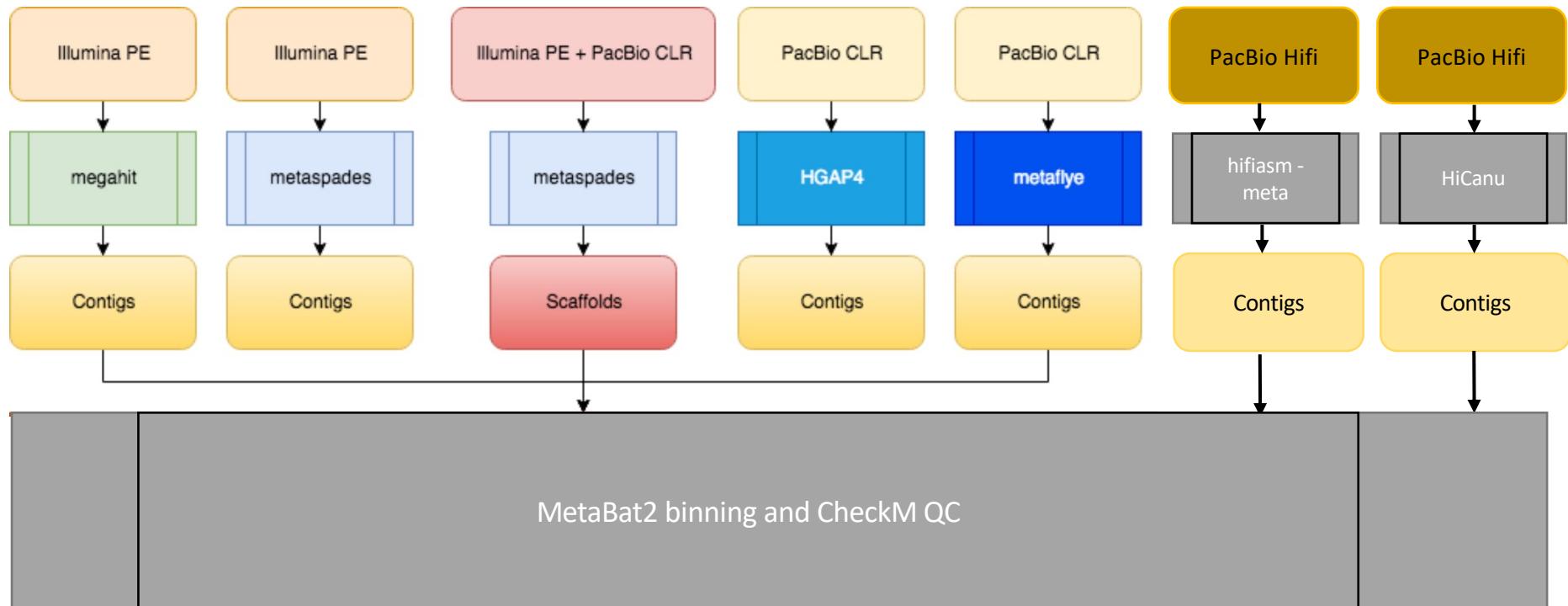


# Overview



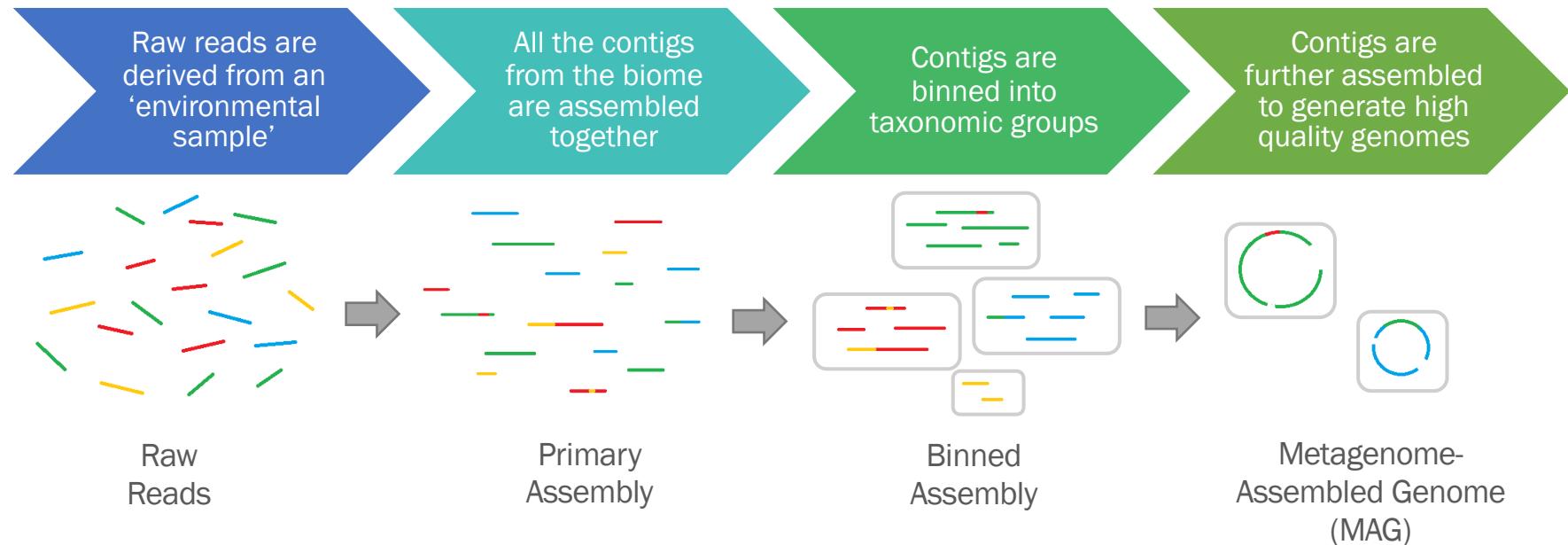


# Metagenome assembly and binning workflow



10  
01  
101101 1  
010 0  
0101 10010 01  
101 10  
010 01  
10 01  
01 1  
1  
0

## Structuring Metagenomic Studies: Types Of Metagenomic Assembly



10  
01  
101101 1  
010 0  
0101 10...  
1 + + + + 010 01

- - - - 101 10

...  
+ + + + - - -

f g c z

01 1  
10  
01

0 0

1 1

0 0

1 1

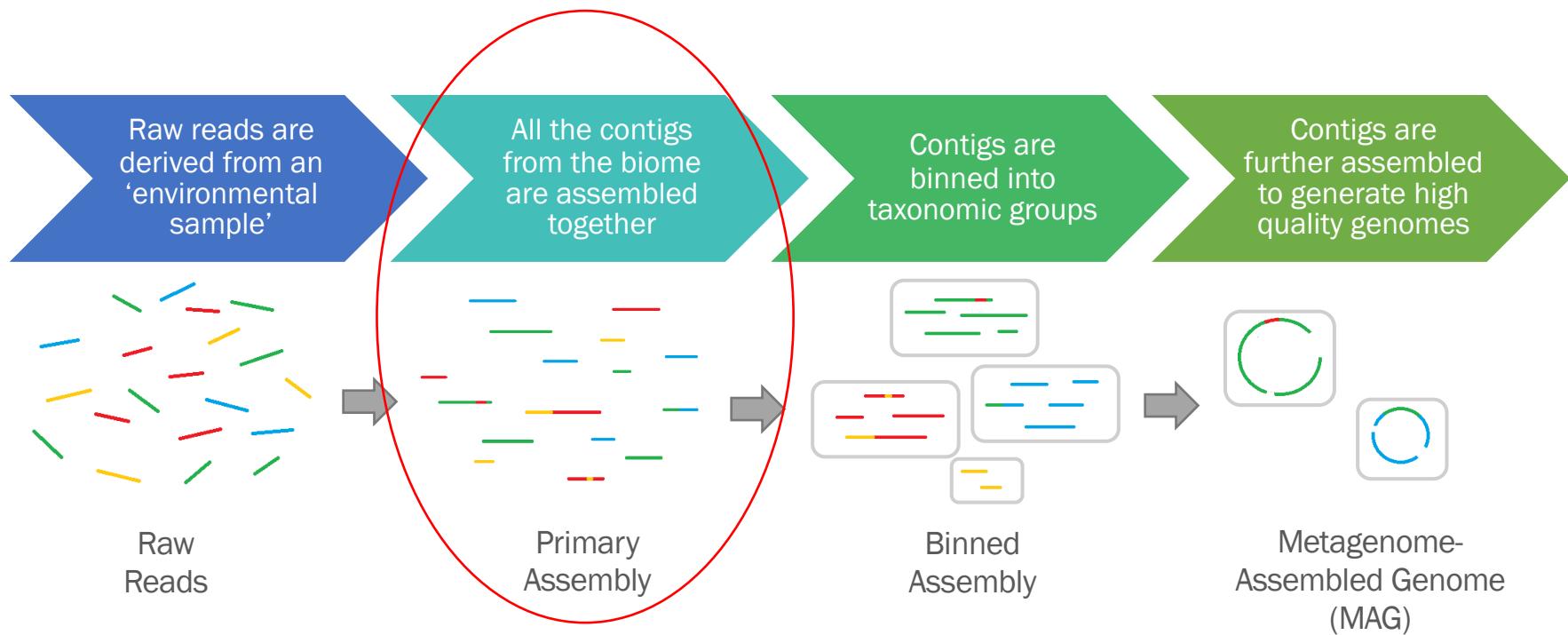
0 0

1 1

0 0

1 1

## Structuring Metagenomic Studies: Types Of Metagenomic Assembly



10  
01  
101101 1  
010 0  
0101 10...  
...101 01  
101 10  
010 01...  
...++  
++

010 01

...  
...++  
++

010 01

...  
...++  
++

010 01

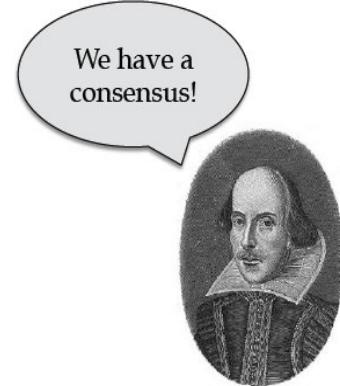
...  
...++  
++01 1  
1 0  
0 1

## Assembly – The Problem

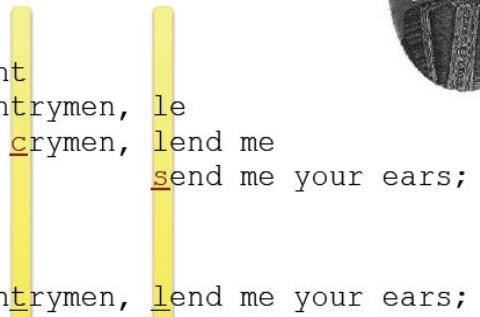
- **Reads**  
ds, Romans, count  
ns, countrymen, le  
Friends, Rom  
 send me your ears;  
 crymen, lend me

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

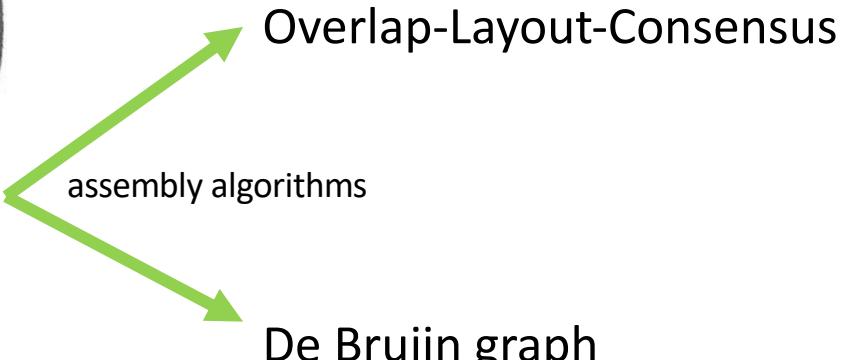
- **Majority consensus**  
Friends, Romans, countrymen, lend me your ears;

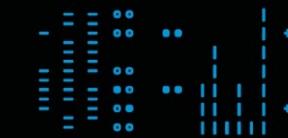


We have a  
consensus!



le  
lend me  
send me your ears;

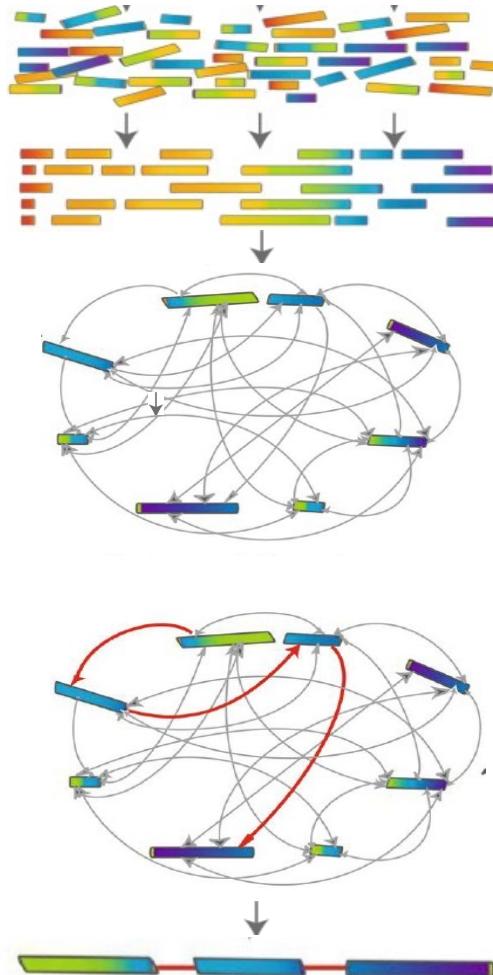


10  
01  
101101 1  
010 0  
0101 10  
010 01010 01  
101 10  
010 01  
10 01

f g c z

## 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





10  
01  
101

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

010 01  
101 10  
010 01  
10 01

## Overlap Layout Consensus - Overlap

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

X: CTCTAGG**GCC**  
Y: TAGGCC**CTC**

X: CTCTAG**G****GCC**  
Y: TAG**G****GCC****CTC**

X: CTC**TAGGCC**  
Y: TAGGCC**CTC**

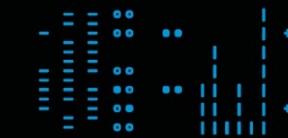
Found it

Resulting overlap  


- 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!

10  
01  
101101 1  
010 0  
0101 10...  
1 ++ +++ 010 01

- - - - -

f g c z

01 1  
10  
010 0  
1 1  
0 0  
1 1

## Overlap Layout Consensus - Layout

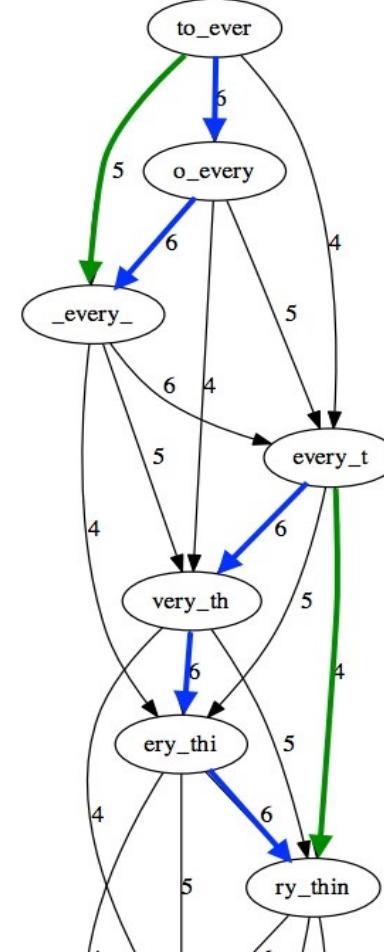
### Pop bubbles

- 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)

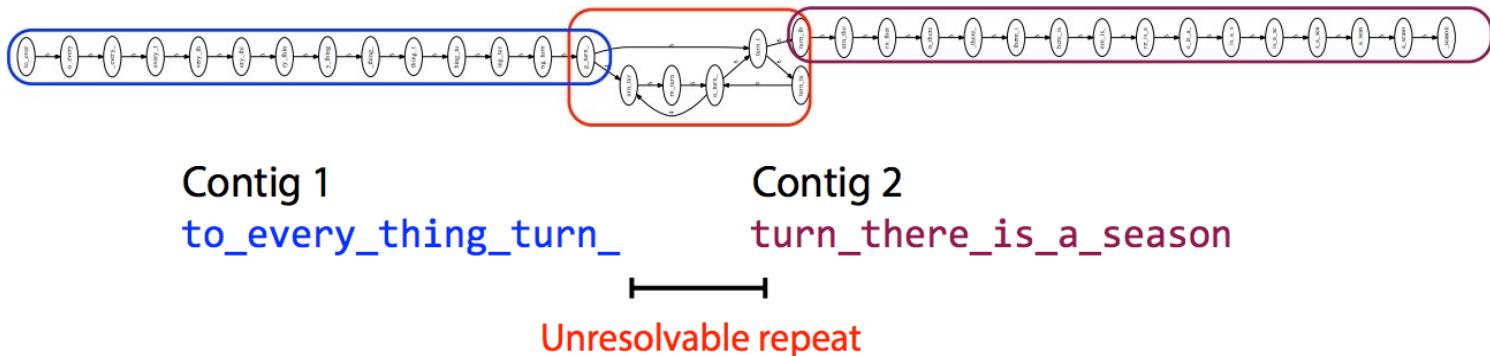




## Overlap Layout Consensus - Layout

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





## Overlap Layout Consensus - Consensus

TAGATTACACAGATTACTGA TTGATGGCGTAA CTA  
TAGATTACACAGATTACTGACTTGATGGCGTAACTA  
TAG TTACACAGATTATTGACTTCATGGCGTAA CTA  
TAGATTACACAGATTACTGACTTGATGGCGTAA CTA  
TAGATTACACAGATTACTGACTTGATGGCGTAA CTA

↓      ↓      ↓      ↓      ↓

TAGATTACACAGATTACTGACTTGATGGCGTAA CTA

Take reads that make up a contig and line them up

Take *consensus*, i.e. majority vote

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

Hamiltonian path

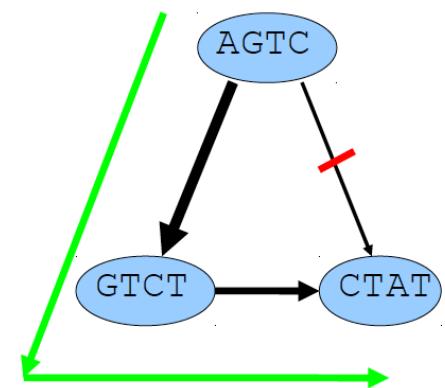
a path that visits each node exactly once

Optimal path shown in green

Un-traversed weak overlap in red

Consensus is read by outputting the overlapped nodes along the path

aGTCTCTat



### 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

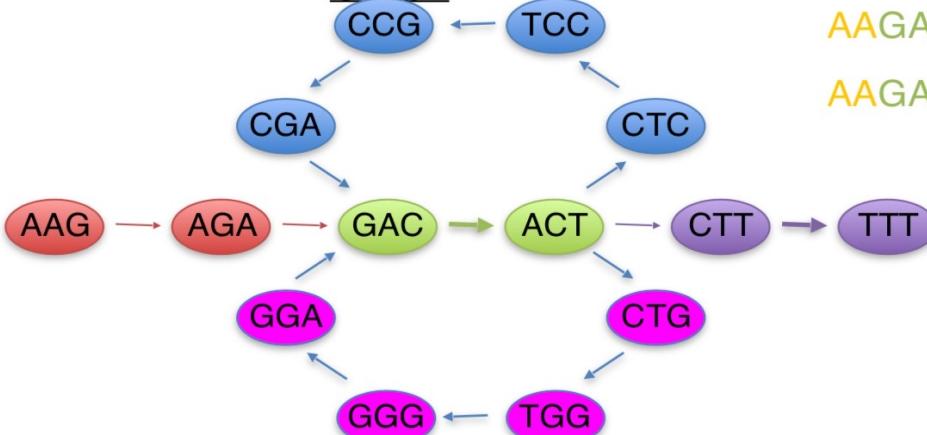
## De Brujin graph - DBG

- 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

### Reads

AAGA  
ACTT  
ACTC  
ACTG  
AGAG  
CCGA  
CGAC  
CTCC  
CTGG  
CTTT  
...

### de Bruijn Graph



### Potential Genomes

AAGACTCCGACTGGGACTTT  
AAGACTGGGACTCCGACTTT  
...

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**



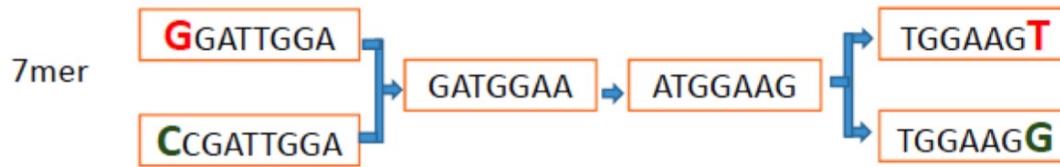
## De Brujin graph - DBG

### Impact of kmer length

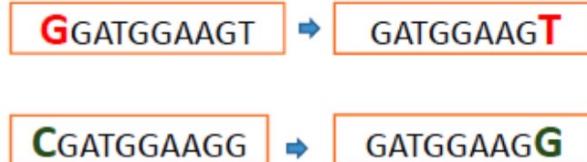
- Short kmers would collapse paralogous regions, and result in more branches

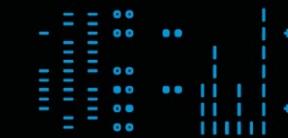
Paralogous regions

**G**GATGGAAG**TCG**..... **C**GATGGAAG**GAT**



9mer



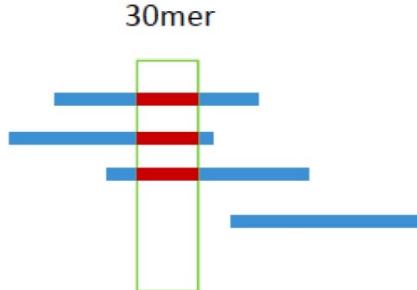
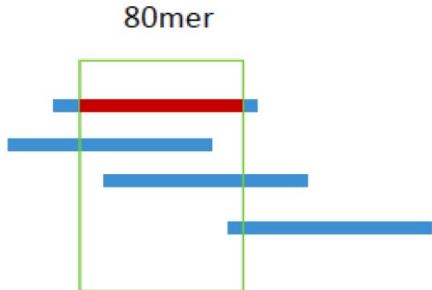
10  
01  
10101 1  
101 10  
010 01  
10 01

## 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

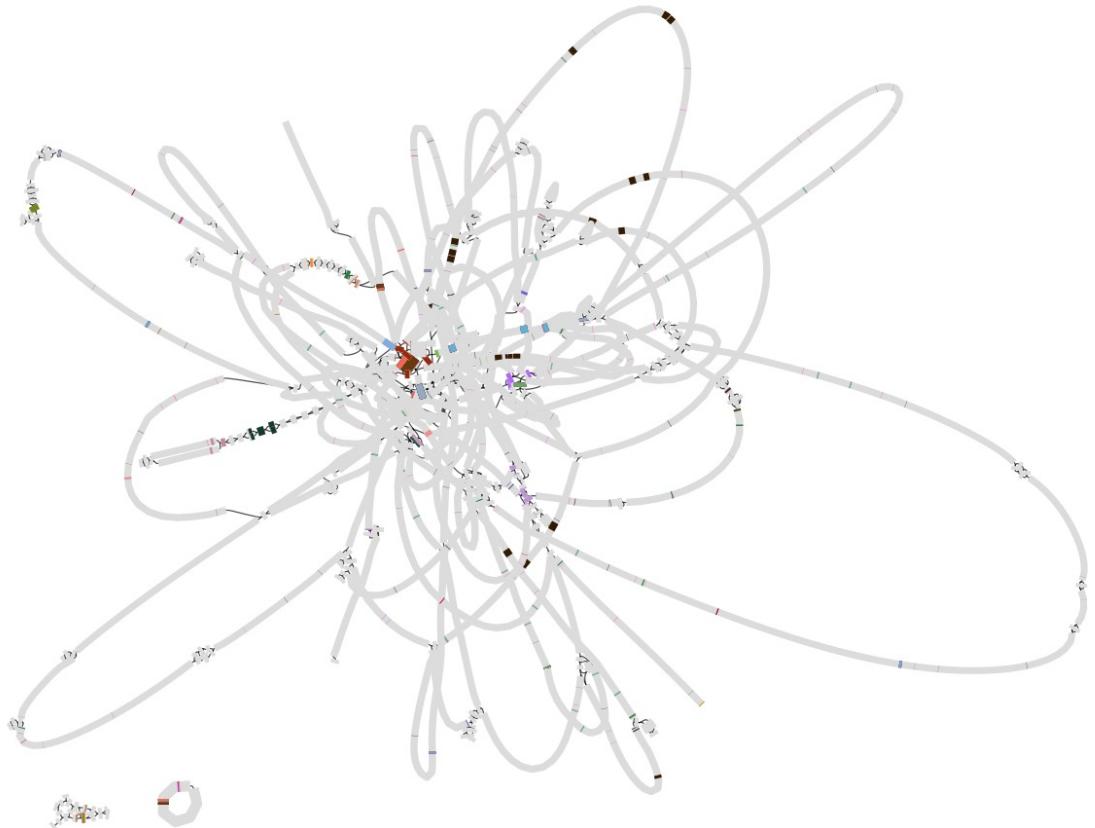


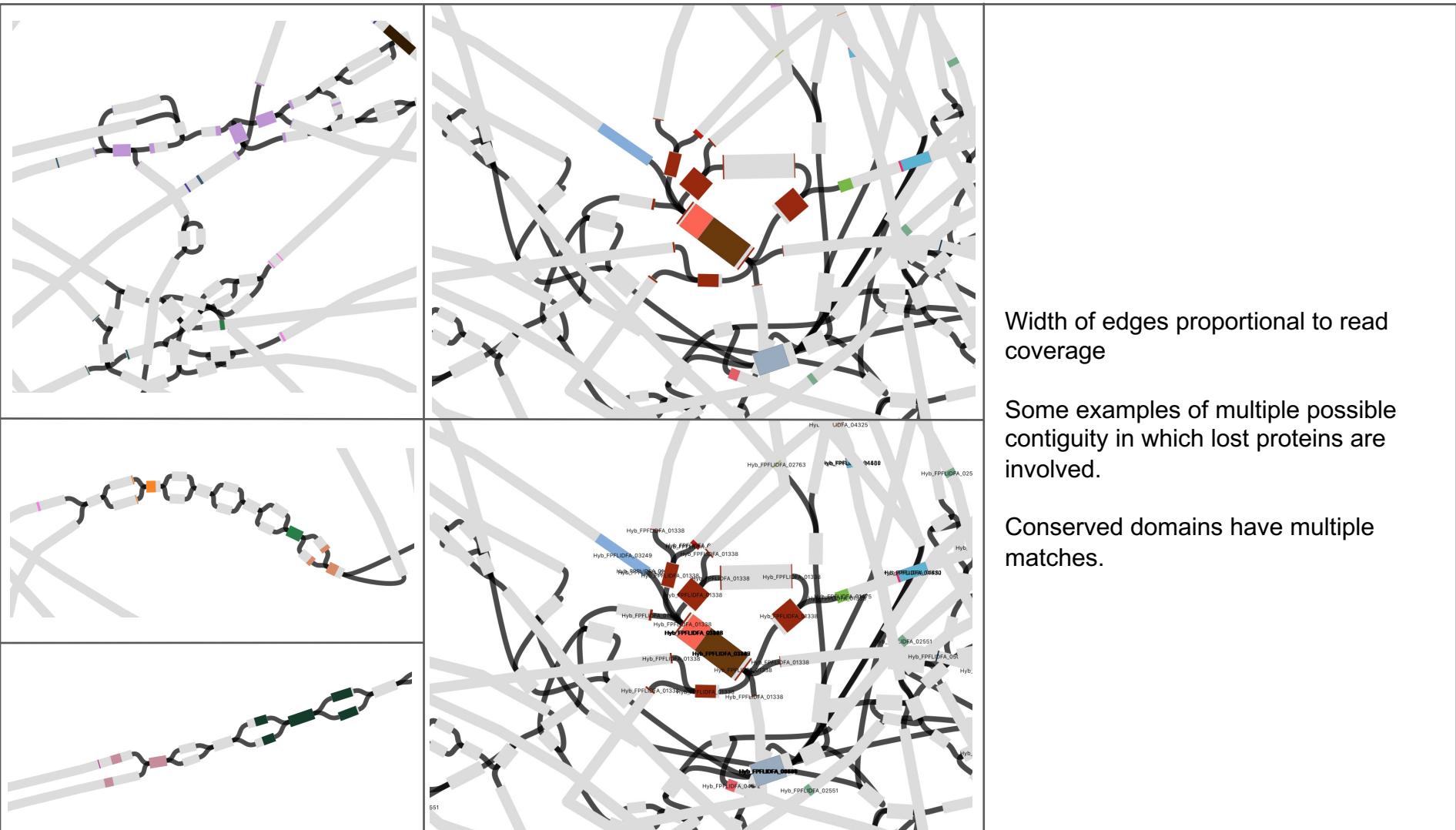
Read depth: 3

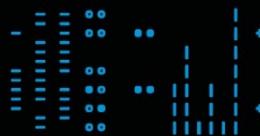


## De Bruijn graph – Visualising assembly graph

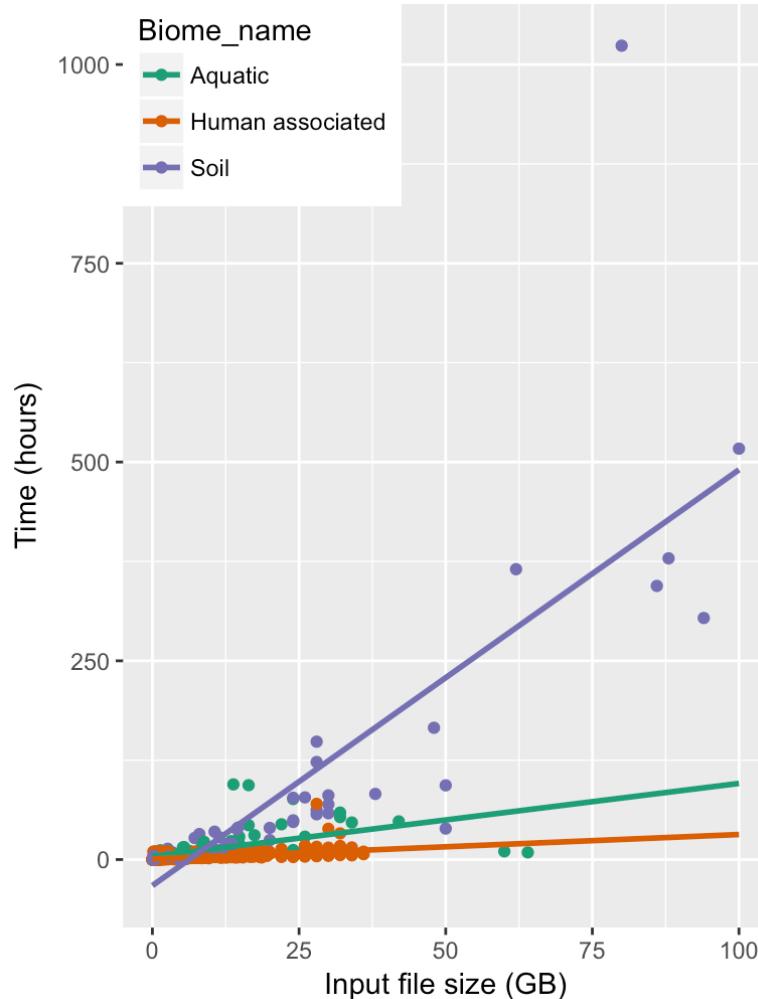
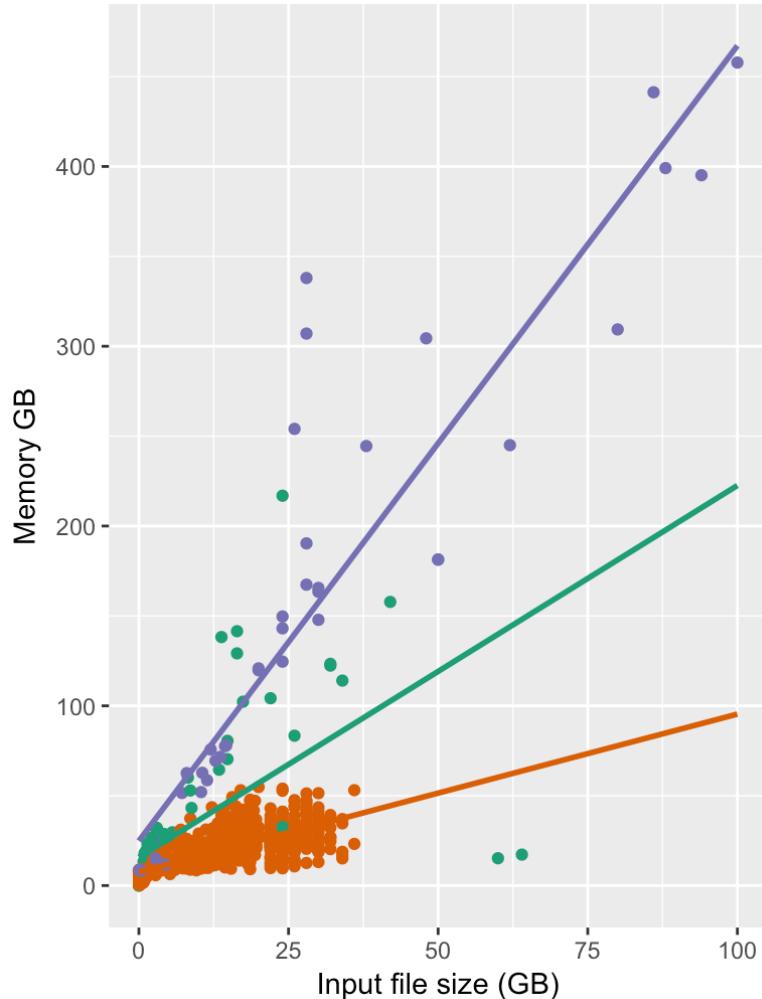
- Tools such as metaSPAdes convert the de Bruijn graph to an assembly graph
- Bandage allows the visualisation and querying of this graph





10  
01  
101

## Computational resources limiting assemblies

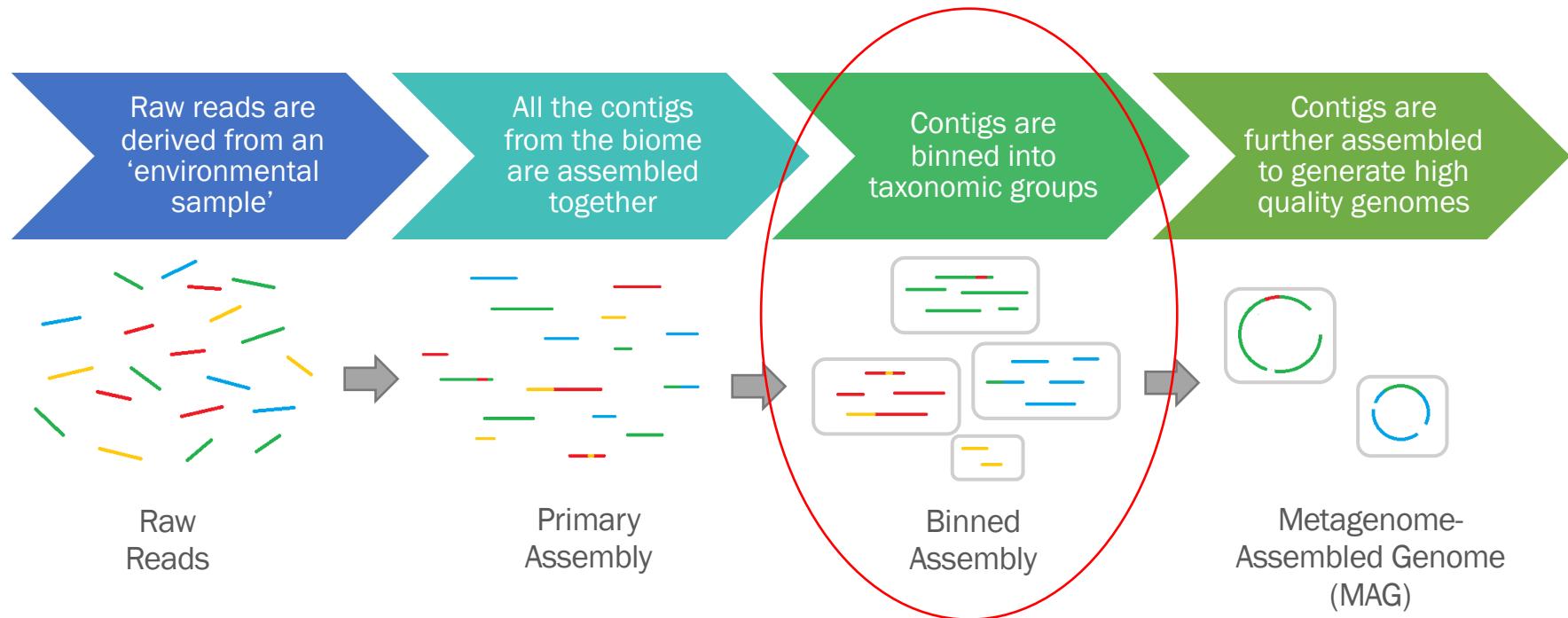


10  
01  
101

## Complications in assemblies

- Different abundance
  - Highly abundant genomes are over-sequenced
    - higher coverage  $\neq$  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

## Structuring Metagenomic Studies: Binning contigs





10  
01  
101

01 1  
01 01  
101 10  
10  
01 1

## Binning contigs: Two ways

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

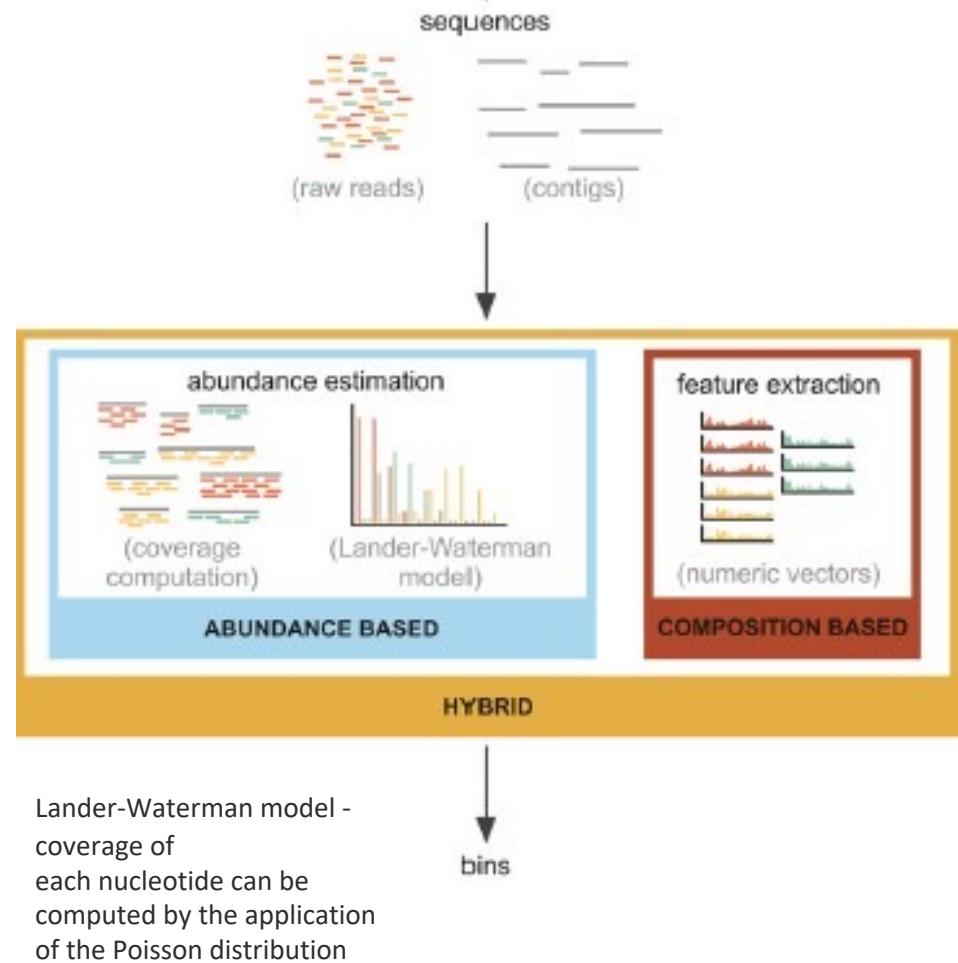
## Binning contigs: Taxonomy-independent binning

### 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 abundance rather than content

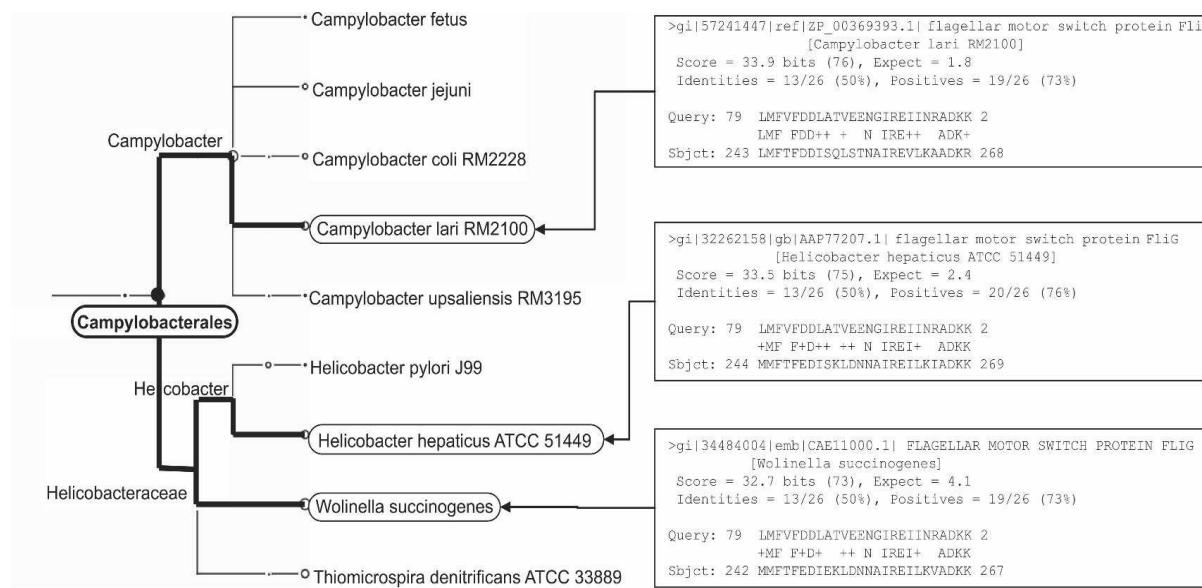


10  
01  
101101 1  
010 0  
101 10  
0101 10  
...  
++ +++ 010 01  
- - - f g c z  
01 1  
10  
01

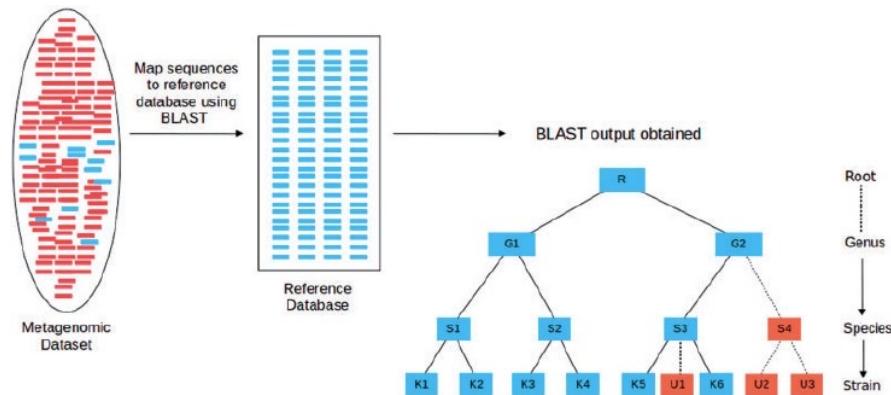
# Taxonomy dependent binning (LCA)

## 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)



# Taxonomy dependent binning (LCA)



## PRO

Leveraging existing information

Amenable to shorter reads

Rare microorganisms identifiable provided sufficient coverage

## CONS

Computationally intensive/time consuming (alignment and eventual downstream assembly)

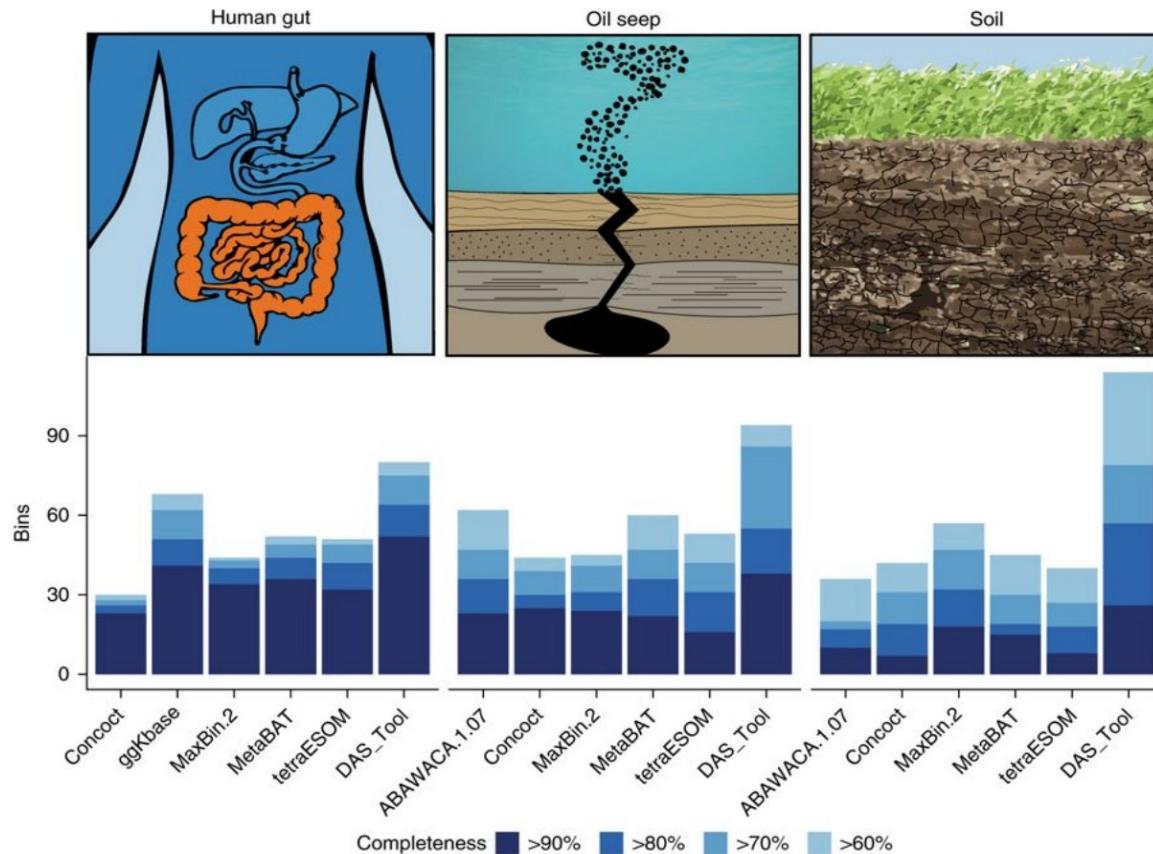
Matching bit-scores sometimes inadequate metric

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



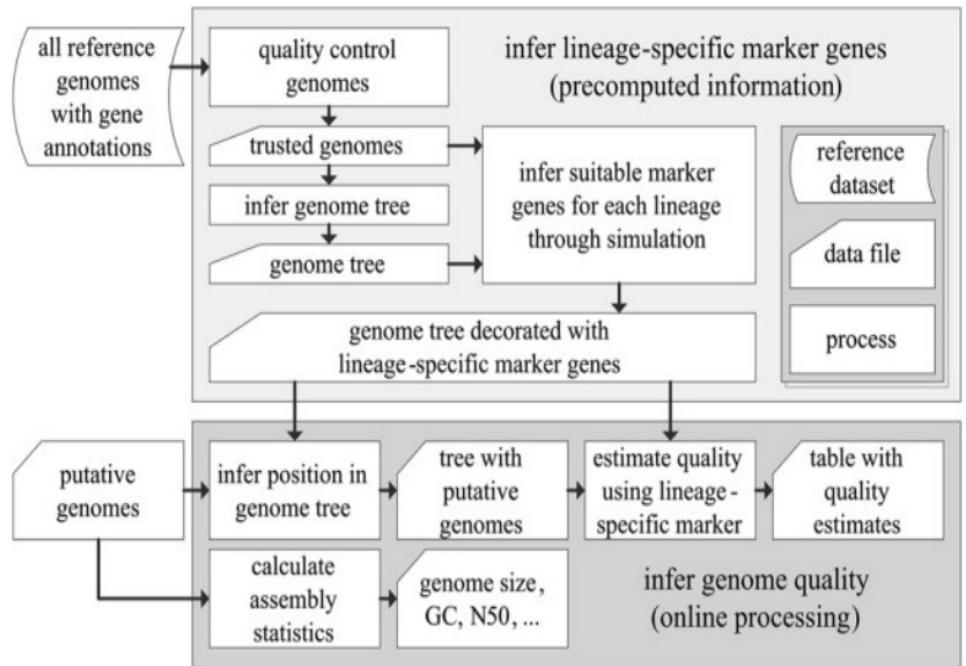


## Binning contigs

### Assessing the quality of reconstructed genome bins

- Estimate completeness and cleanliness
  - 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 multicity genes
    - a gene identified as single copy in  $\geq 97\%$  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

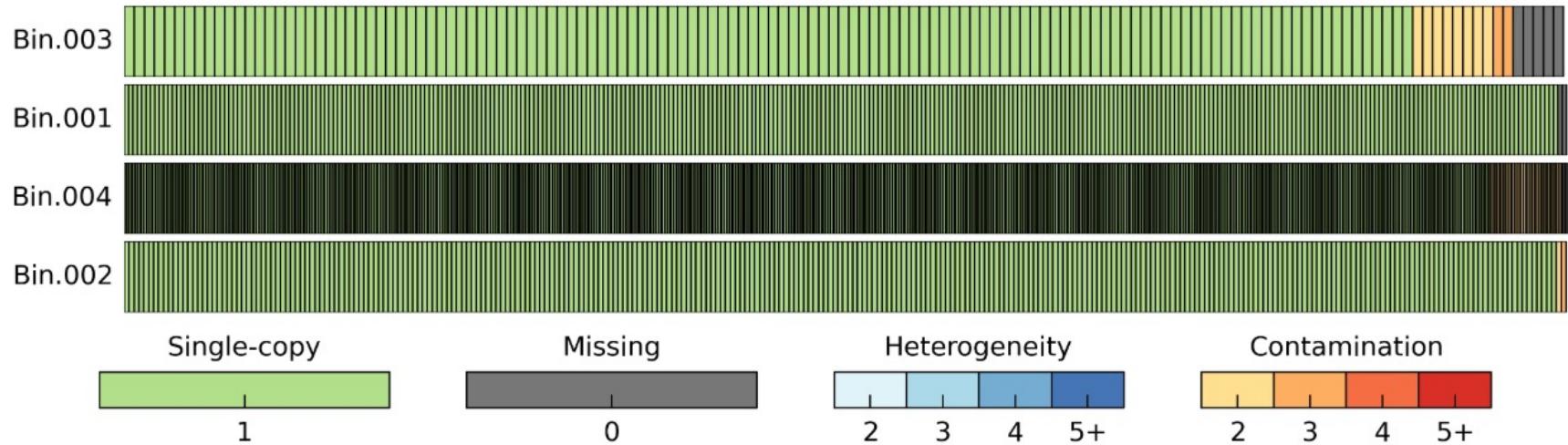
Assesing your bins against the database

Parks DH et al. 2015. Genome Research



## Binning contigs: Assessing quality with CheckM

[CheckM PLOT](#) | [CheckM Table](#)





10  
01  
101

01 1  
01 01  
101 10  
10 01  
01 1

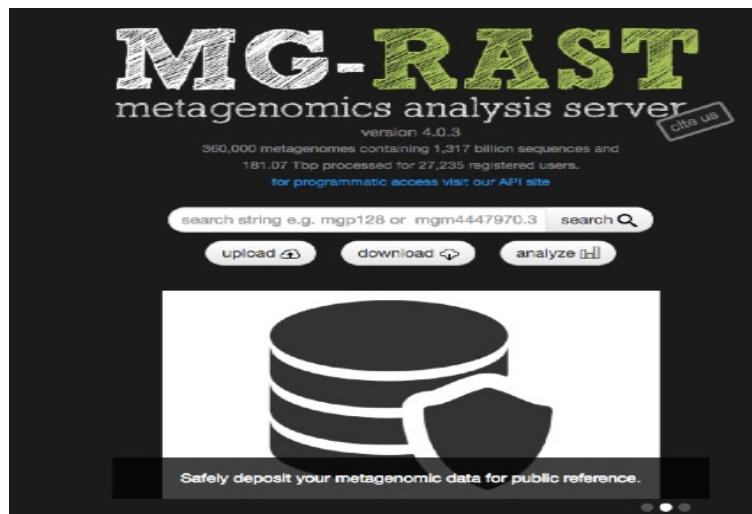
## What makes a good binned assembly

- 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

10  
01  
10101 01  
101 10  
010 01  
101 10  
010 01  
10 01  
01 1

## Tools for robust taxonomic classification

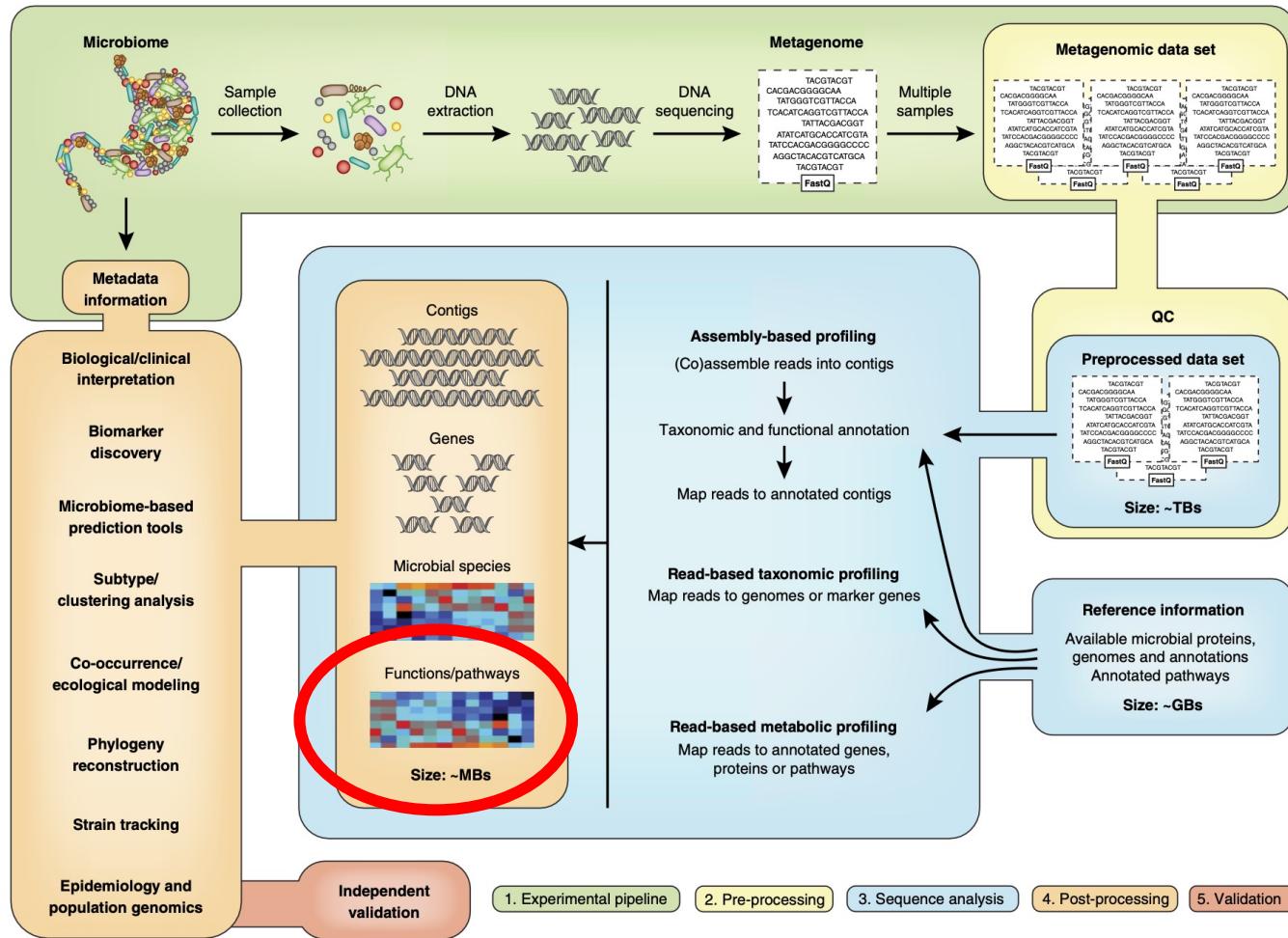
MG-RAST (GenBank annotation)  
(<https://www.mg-rast.org/>)



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



# Overview



## 3. Metagenome assembly and annotation

10  
01  
0101  
1001  
01  
010 01  
101 10  
010 01  
10  
01  
01

# 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.

10  
01  
10101  
01  
010101  
1  
010  
10010  
01  
101  
1001  
1  
01  
10  
01

# Protein prediction: Prokka

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

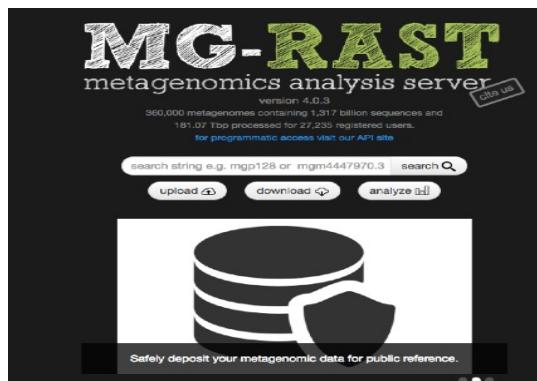
1. An optional **user-provided set of annotated proteins**. These are expected to be trustworthy curated datasets and will be used as the primary source of annotation. They are searched using BLAST+ blastp ([Camacho \*et al.\*, 2009](#)).
2. All bacterial **proteins in UniProt** ([Apweiler \*et al.\*, 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 hmmsearch from the HMMER 3.1 package ([Eddy, 2011](#)).
5. If no matches can be found, label as 'hypothetical protein'.

10  
01  
101010 01  
101 10  
010 01  
10 01  
01 1

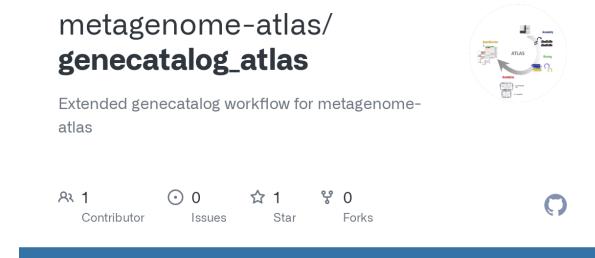
# Annotation

- 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:



The MG-RAST homepage features a large "MG-RAST" logo at the top left. Below it, the text "metagenomics analysis server" and "version 4.0.3". A sub-header states "360,000 metagenomes containing 1,317 billion sequences and 181.07 Tbp processed for 27,235 registered users." A "click us" button is present. Below this, there's a search bar with placeholder text "search string e.g. mgp128 or mgm4447970.3" and a "search" button. Below the search bar are three buttons: "upload", "download", and "analyze". At the bottom left is a large icon of a database cylinder with a shield, and the text "Safely deposit your metagenomic data for public reference".



The GitHub page for "metagenome-atlas/genecatalog\_atlas" shows the repository name at the top. Below it is a description: "Extended genecatalog workflow for metagenome-atlas". There's a circular profile picture of a brain-like structure. Below the description are four metrics: "Contributor" (1), "Issues" (0), "Star" (1), and "Forks" (0). A blue progress bar is at the bottom.

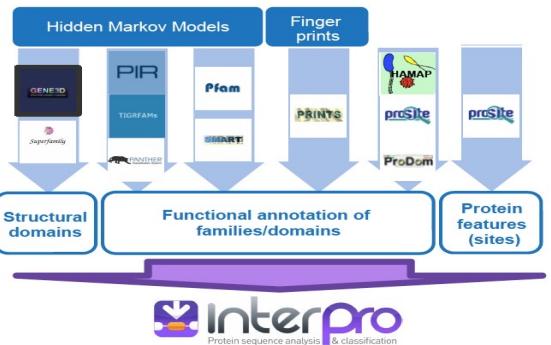


The SEED homepage has a blue header with the text "The SEED" and "Home of the SEED". Below the header is a green navigation bar with buttons for "page", "discussion", and "view source". The main content area features a circular logo with a green microorganism and the text "The SEED". To the right, there's a "navigation" sidebar with links to "Home of the SEED", "Manifesto", "SEED People", and "Contact". Below the sidebar is a section titled "Home of the SEED" with the subtext "(Redirected from Main Page)" and a "Jump to: navigation, search" link.

10  
01  
101

# Annotation

## Annotation of protein domains (Interpro scan)



Orthology based gene function prediction  
(EggNOG, GO Consortium, OMA GO  
annotation, Pannzer2)

The screenshot shows the EggNOG 4.5.1 homepage. At the top, it says 'eggNOG 5.0 now available'. Below that is a section titled 'Discover EggNOG 4.5.1' with statistics: Organisms (2,031), Viruses (352), Orthologous Groups (190k), and Trees & Algs (1.9M). The left sidebar includes links for Home, Sequence search, eggNOG-mapper, Downloads, API, Methods, and Viral OGs. The right sidebar provides search and filter options for Gene Ontology Consortium data.

The screenshot shows the Gene Ontology Consortium website. It features a search bar for 'Gene IDs here...', dropdown menus for 'Ontology' (set to 'biological process') and 'Organism' (set to 'Homo sapiens'), and a 'Submit' button. To the right, there's a sidebar for 'Enrichment analysis' and another for 'Annotations'.

**PANNZER**  
RAPID FUNCTIONAL ANNOTATION SERVER  
POWERED BY **SANS**

## Metabolic pathway prediction (KEGG)

The screenshot shows the KEGG PATHWAY Database homepage. It features a navigation menu with links to PATHWAY, BRITE, MODULE, KO, GENES, LIGAND, NETWORK, DISEASE, DRUG, and DBGET. Below the menu is a search bar with fields for 'Select prefix map', 'Organism', 'Enter keywords', and 'Go'. A link to 'Help' is also present. The main content area is titled 'Pathway Maps' and contains a brief description of KEGG PATHWAY. A sidebar on the right lists various biological categories: 1. Metabolism, 2. Genetic Information Processing, 3. Environmental Information Processing, 4. Cellular Processes, 5. Organismal Systems, 6. Human Diseases, and 7. Drug Development. At the bottom, it says 'KEGG PATHWAY is a reference database for Pathway Mapping.'

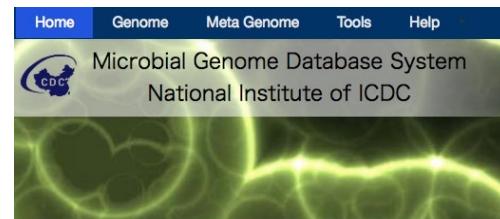
10  
01  
101010 01  
101 10  
010 01  
10  
01  
1

# Annotation

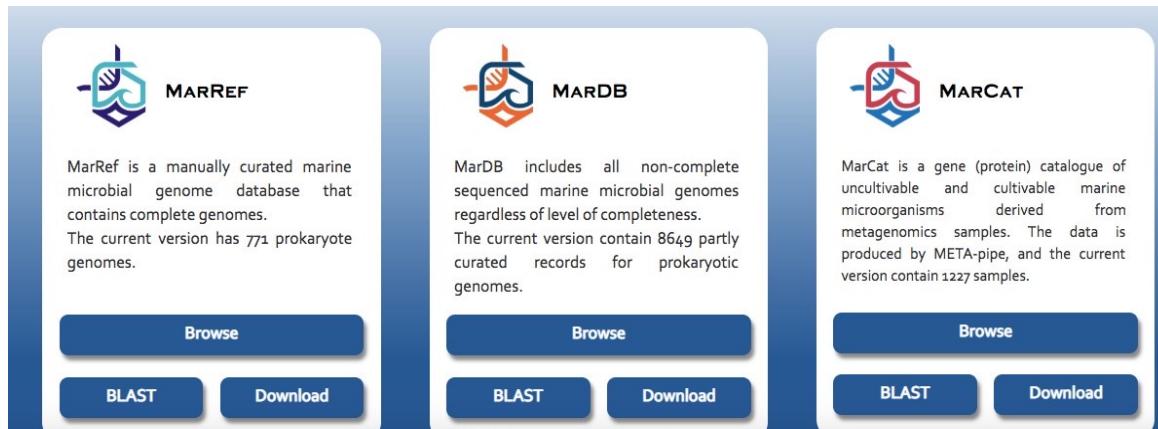
## Using ecosystem-specific databases for annotation and taxonomy



The screenshot shows the Terragenome website. The header includes links for HOME, ABOUT, WORKSHOPS, METADATA STANDARDS, PROJECT DIRECTORY, and A. The main title "Terragenome" is displayed in large yellow text above a photograph of a field with young plants. Below the title, it says "International Soil Metagenome Sequencing Consortium".



The screenshot shows the Microbial Genome Database System homepage. The header includes links for Home, Genome, Meta Genome, Tools, and Help. The main title is "Microbial Genome Database System" and "National Institute of ICDC". Below the title is a green abstract graphic.



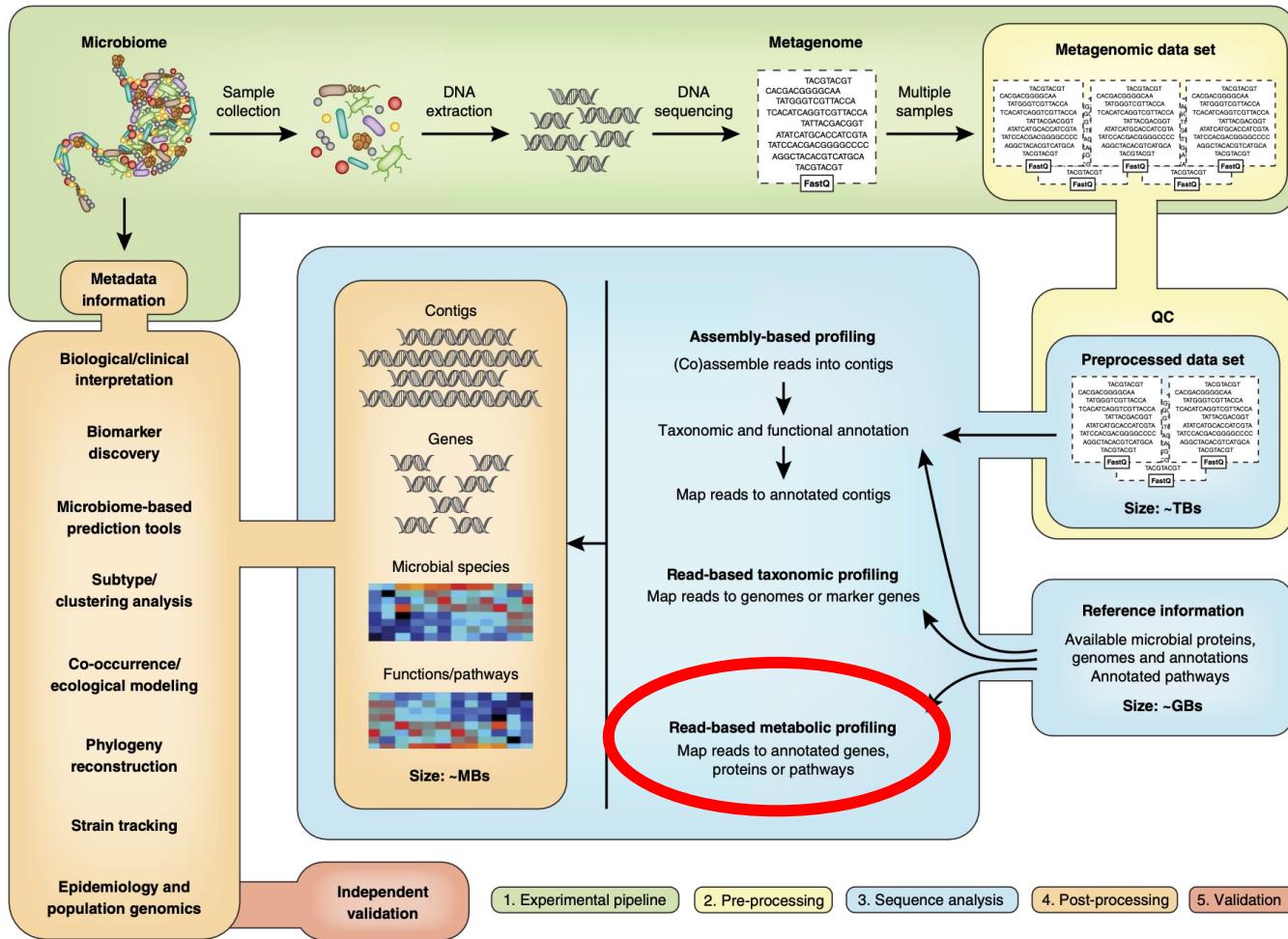
The screenshot shows the homepage of three marine microbial genome databases: MarRef, MarDB, and MarCat. Each section has a logo, title, brief description, and buttons for Browse, BLAST, and Download.

- MarRef**: MarRef is a manually curated marine microbial genome database that contains complete genomes. The current version has 771 prokaryote genomes.
  - Browse
  - BLAST
  - Download
- MarDB**: MarDB includes all non-complete sequenced marine microbial genomes regardless of level of completeness. The current version contain 8649 partly curated records for prokaryotic genomes.
  - Browse
  - BLAST
  - Download
- MarCat**: MarCat is a gene (protein) catalogue of uncultivable and cultivable marine microorganisms derived from metagenomics samples. The data is produced by META-pipe, and the current version contain 1227 samples.
  - Browse
  - BLAST
  - Download



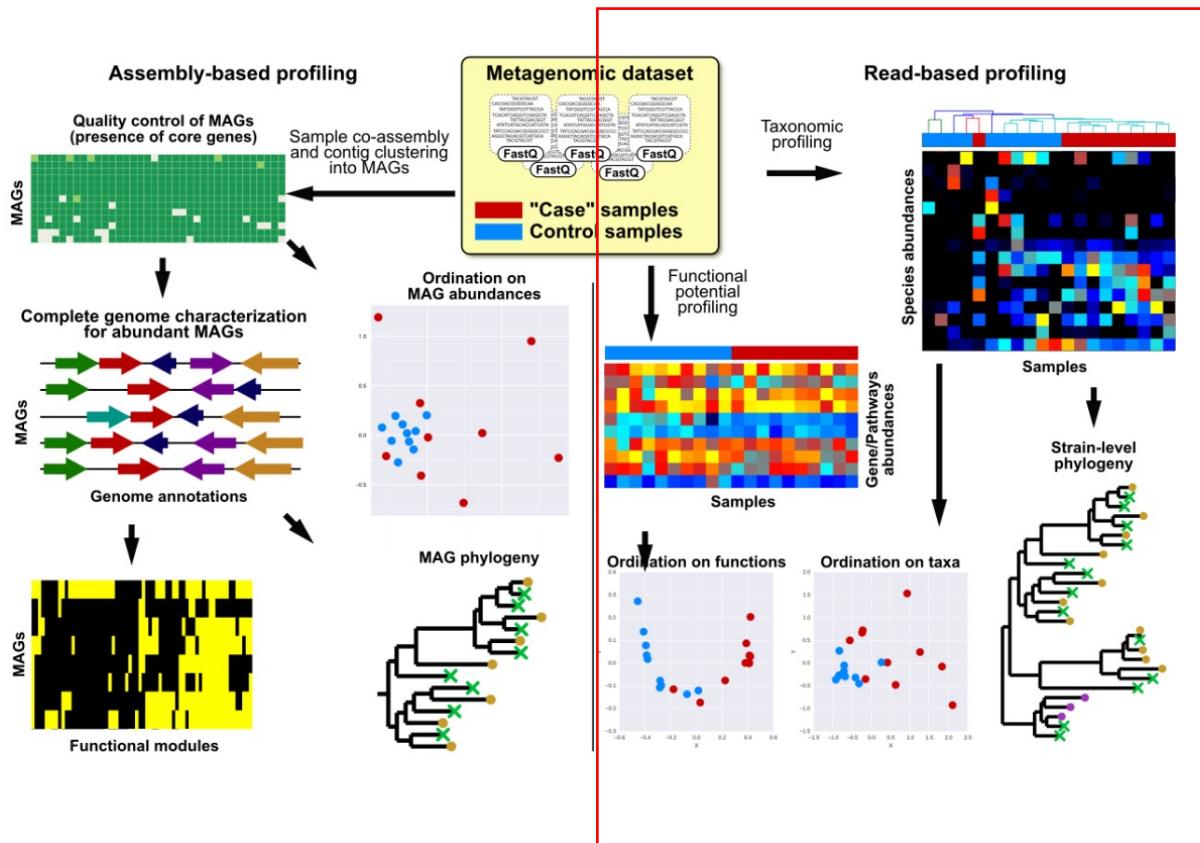


# Overview



**4. Assembly – free taxonomic profiling and annotation**

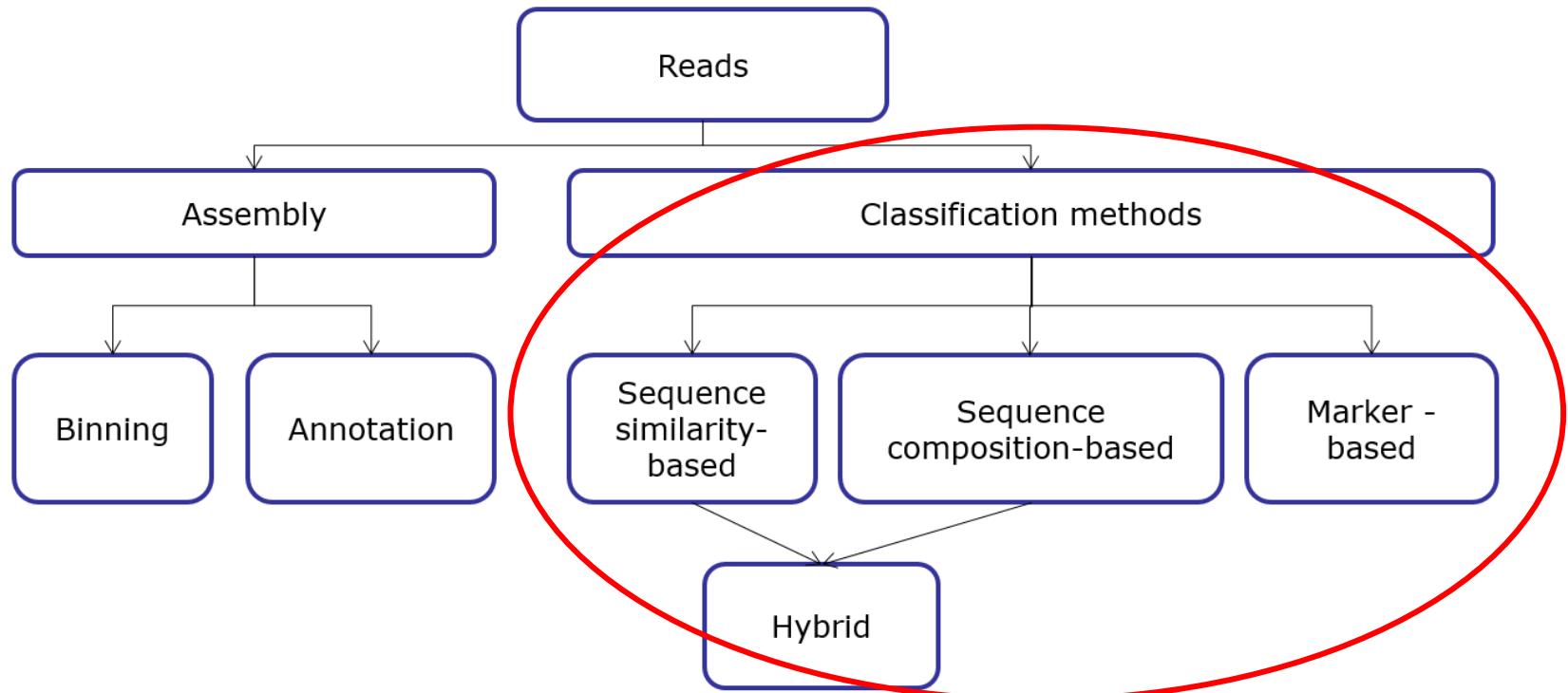
# Assembly-free approach



Why?

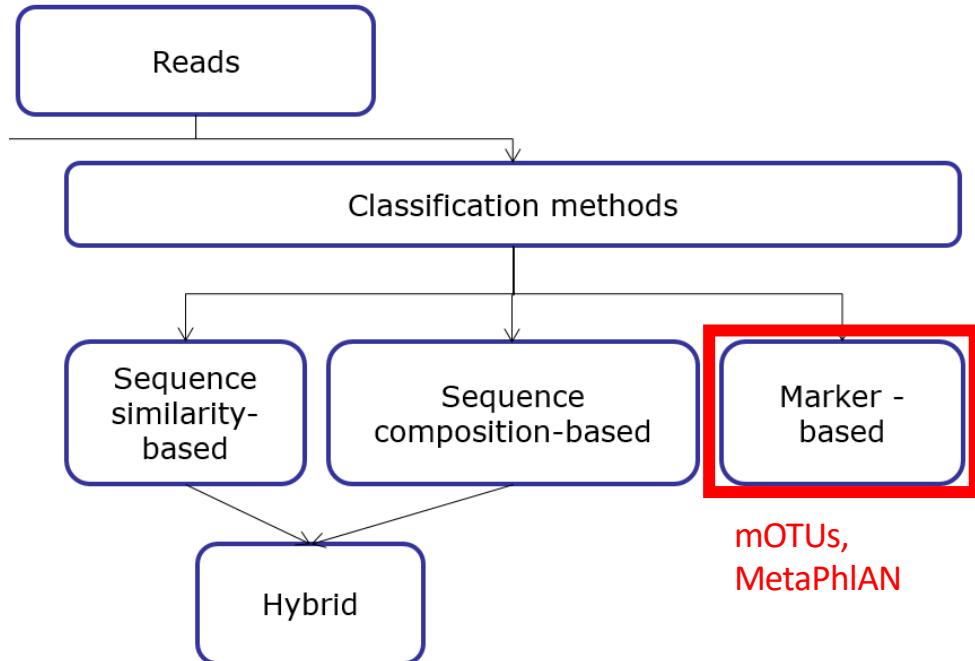
- speed up computation – assembly methods take a long time
- make it possible to profile low-abundance organisms that cannot be assembled de novo – but database dependent
- might have improved sensitivity
- can answer questions about presence/absence

# Assembly-free approach: taxonomy



10  
01  
10101 01  
101 10  
10  
01 01  
10  
01 01

# Assembly-free approach: taxonomy

[Home](#) [About Us](#) [Teaching](#) [Tools](#) [People](#) [Join Our Team](#)

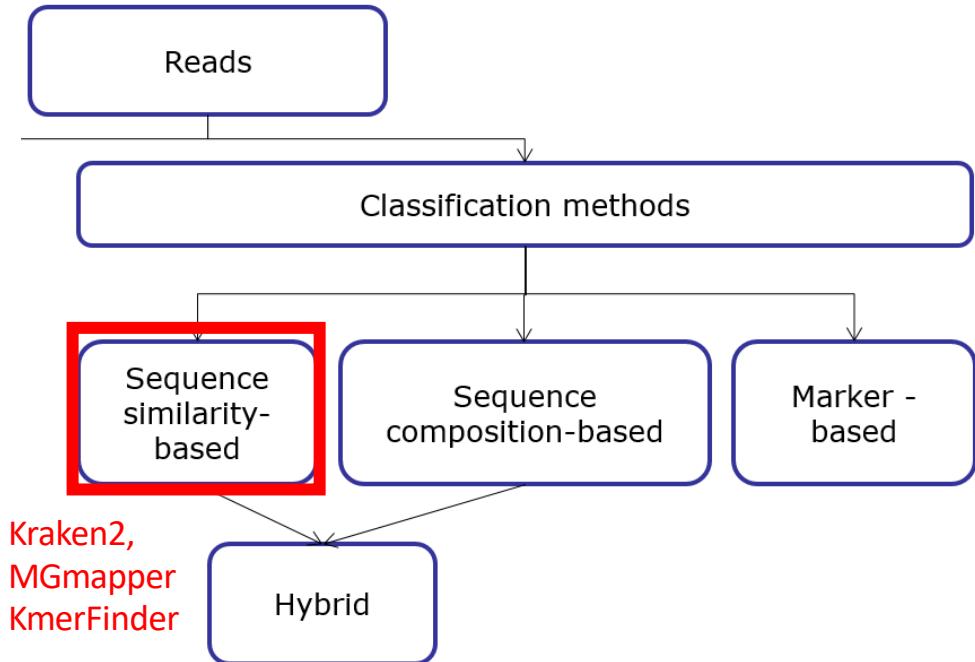
## MetaPhlAn 3.0

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

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

10  
01  
101101 1  
010 0  
0101 10010 01  
101 10  
010 01  
10 01  
01 1

# Assembly-free approach: taxonomy



## Kraken 2

Taxonomic Sequence Classification System



Home

Manual

Downloads

CCB &gt; Software &gt; Kraken2

(<https://ccb.jhu.edu/software/kraken2/>)

## Center for Genomic Epidemiology

Home

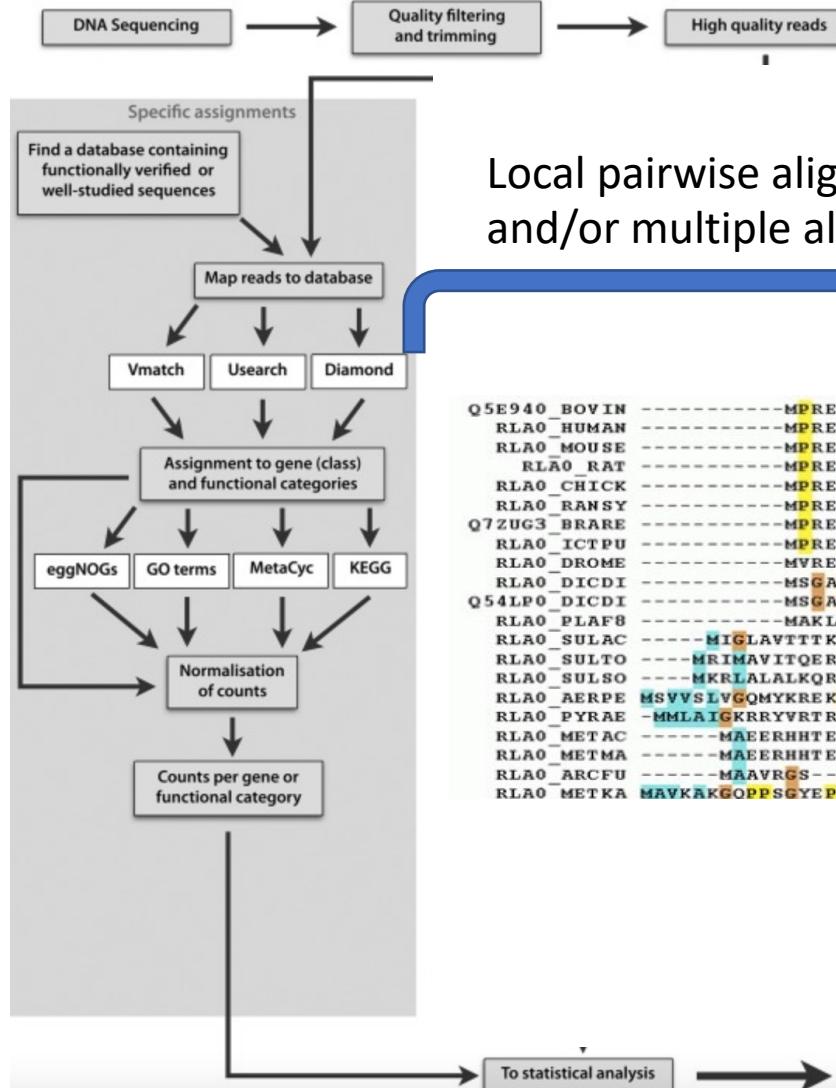
Services

Publications

KmerFinder 3.2

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

# Assembly-free approach: function



Sequence alignment results (e.g., BLAST output) showing alignments between various species (e.g., BOV, HUMAN, MOUSE, RAT, CHICK, RANSY, BRARE, ICTPU, DROME, DICDI, PLAF8, SULAC, SULTO, SULSO, AERPE, PYRAE, METAC, METMA, ARCFU, METKA) and a query sequence. The alignments are color-coded by residue conservation.

Query Position	Residue	Species 1	Species 2	Species 3	Species 4	Species 5	Species 6	Species 7	Species 8
Q5E940_BOV	I	-	-	-	-	-	-	-	-
RLAO_HUMAN	-	-	-	-	-	-	-	-	-
RLAO_MOUSE	-	-	-	-	-	-	-	-	-
RLAO_RAT	-	-	-	-	-	-	-	-	-
RLAO_CHICK	-	-	-	-	-	-	-	-	-
RLAO_RANSY	-	-	-	-	-	-	-	-	-
Q7ZUG3_BRARE	-	-	-	-	-	-	-	-	-
RLAOICTPU	-	-	-	-	-	-	-	-	-
RLAO_DROME	-	-	-	-	-	-	-	-	-
RLAO_DICDI	-	-	-	-	-	-	-	-	-
Q54LP0_DICDI	-	-	-	-	-	-	-	-	-
RLAO_PLAF8	-	-	-	-	-	-	-	-	-
RLAO_SULAC	-	-	-	-	-	-	-	-	-
RLAO_SULTO	-	-	-	-	-	-	-	-	-
RLAO_SULSO	-	-	-	-	-	-	-	-	-
RLAO_AERPE	-	-	-	-	-	-	-	-	-
RLAO_PYRAE	-	-	-	-	-	-	-	-	-
RLAO_METAC	-	-	-	-	-	-	-	-	-
RLAO_METMA	-	-	-	-	-	-	-	-	-
RLAO_ARCFU	-	-	-	-	-	-	-	-	-
RLAO_METKA	-	-	-	-	-	-	-	-	-

Sequence alignment results (e.g., BLAST output) showing alignments between various species (e.g., BOV, HUMAN, MOUSE, RAT, CHICK, RANSY, BRARE, ICTPU, DROME, DICDI, PLAF8, SULAC, SULTO, SULSO, AERPE, PYRAE, METAC, METMA, ARCFU, METKA) and a query sequence. The alignments are color-coded by residue conservation.

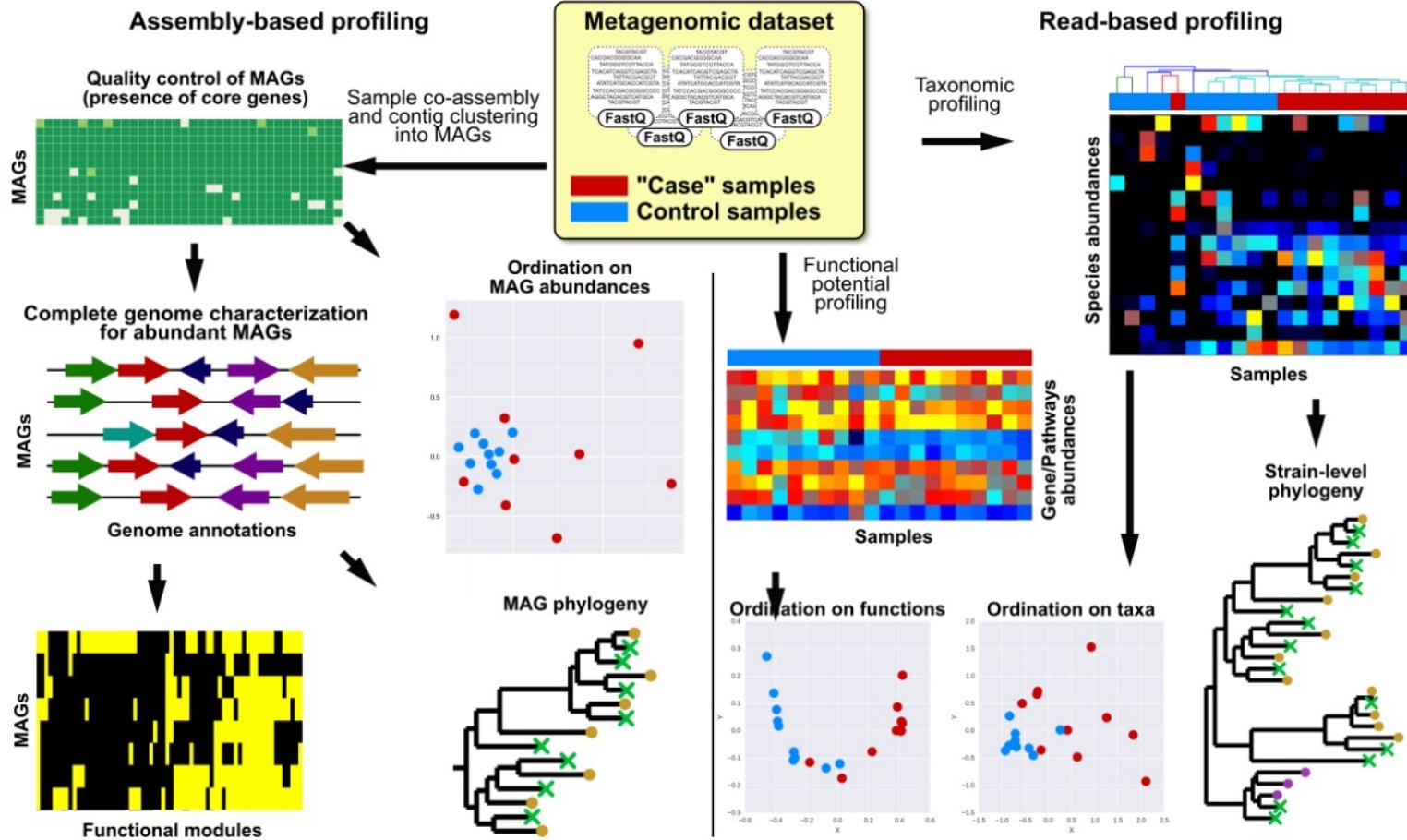
Query Position	Residue	Species 1	Species 2	Species 3	Species 4	Species 5	Species 6	Species 7	Species 8
Q5E940_BOV	I	-	-	-	-	-	-	-	-
RLAO_HUMAN	-	-	-	-	-	-	-	-	-
RLAO_MOUSE	-	-	-	-	-	-	-	-	-
RLAO_RAT	-	-	-	-	-	-	-	-	-
RLAO_CHICK	-	-	-	-	-	-	-	-	-
RLAO_RANSY	-	-	-	-	-	-	-	-	-
Q7ZUG3_BRARE	-	-	-	-	-	-	-	-	-
RLAOICTPU	-	-	-	-	-	-	-	-	-
RLAO_DROME	-	-	-	-	-	-	-	-	-
RLAO_DICDI	-	-	-	-	-	-	-	-	-
Q54LP0_DICDI	-	-	-	-	-	-	-	-	-
RLAO_PLAF8	-	-	-	-	-	-	-	-	-
RLAO_SULAC	-	-	-	-	-	-	-	-	-
RLAO_SULTO	-	-	-	-	-	-	-	-	-
RLAO_SULSO	-	-	-	-	-	-	-	-	-
RLAO_AERPE	-	-	-	-	-	-	-	-	-
RLAO_PYRAE	-	-	-	-	-	-	-	-	-
RLAO_METAC	-	-	-	-	-	-	-	-	-
RLAO_METMA	-	-	-	-	-	-	-	-	-
RLAO_ARCFU	-	-	-	-	-	-	-	-	-
RLAO_METKA	-	-	-	-	-	-	-	-	-

To statistical analysis

See Fig. 3



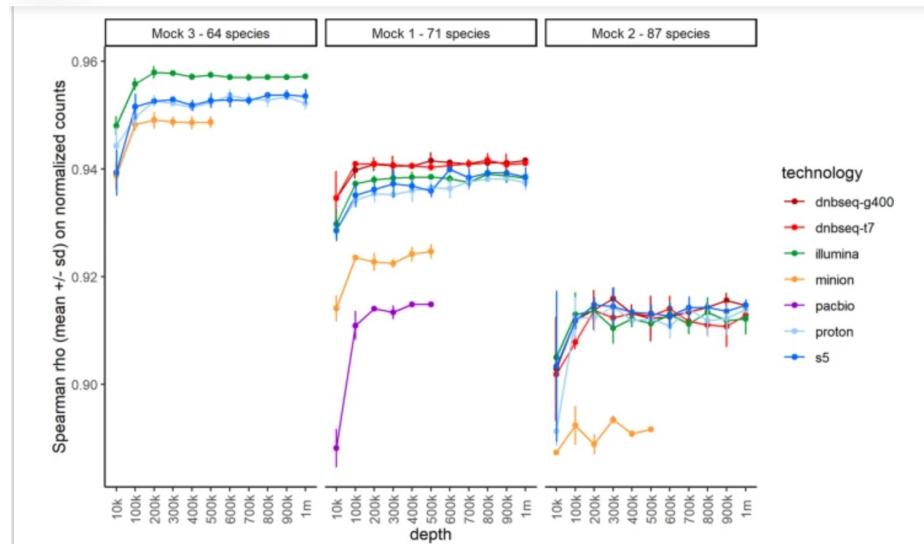
# Conclusion



# 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 (metatlyfe)	PacBio Sequel II (metatlyfe)
Nb Reads (M)	20	20	2 x 10	2 x 10	2 x 10	0.696	0.524
Nb Contigs	45,510	43,879	40,147	44,887	44,603	1,283	<b>437</b>
Largest Contig (bp)	384,996	794,907	1,599,668	1,063,396	1,002,925	4,324,150	<b>7,147,004</b>
N50 (bp)	7,847	9,089	13,707	8,519	8,184	759,940	<b>2,013,697</b>
Genome Fraction(%)	54.767	55.257	61.897	49.397	47.365	44.955	<b>68.197</b>
Mismatches per 100kbps	83.29	89.12	47.55	77.22	107.52	339.99	<b>18.3</b>
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	<b>6</b>
Fully Unaligned Length (bp)	900,150	821,545	620,805	426,856	711,992	6,279,694	<b>134,713</b>
NB full genome*	5	5	12	7	7	22	<b>36</b>

10  
01  
10101 01  
101 10  
010 01  
10 01  
01 1

## 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](https://github.com/zajacn/metagenomics_course_FGCZ)