



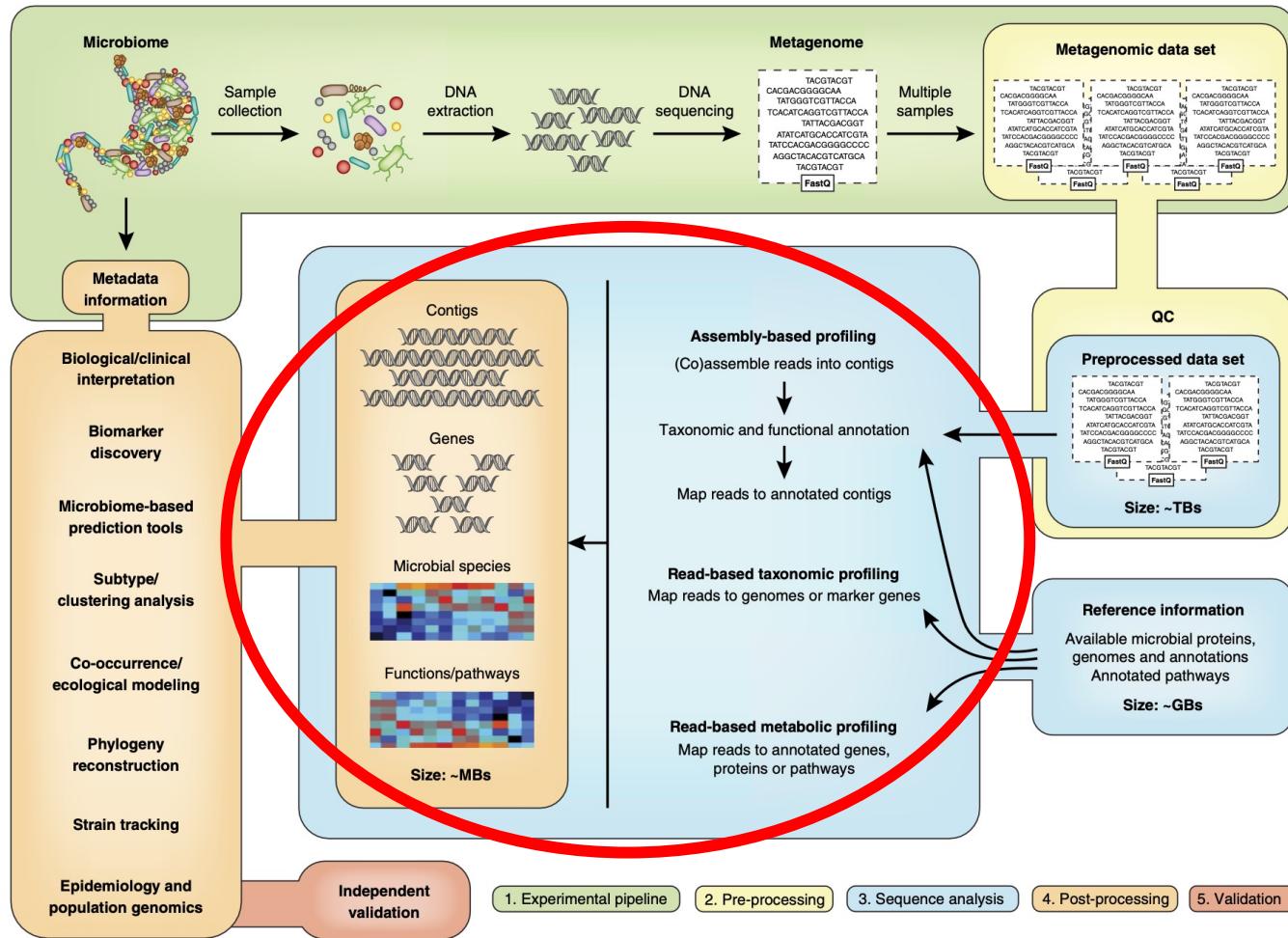
# Shotgun metagenomics

Dr. Natalia Zajac

Metagenomics, 12.2023

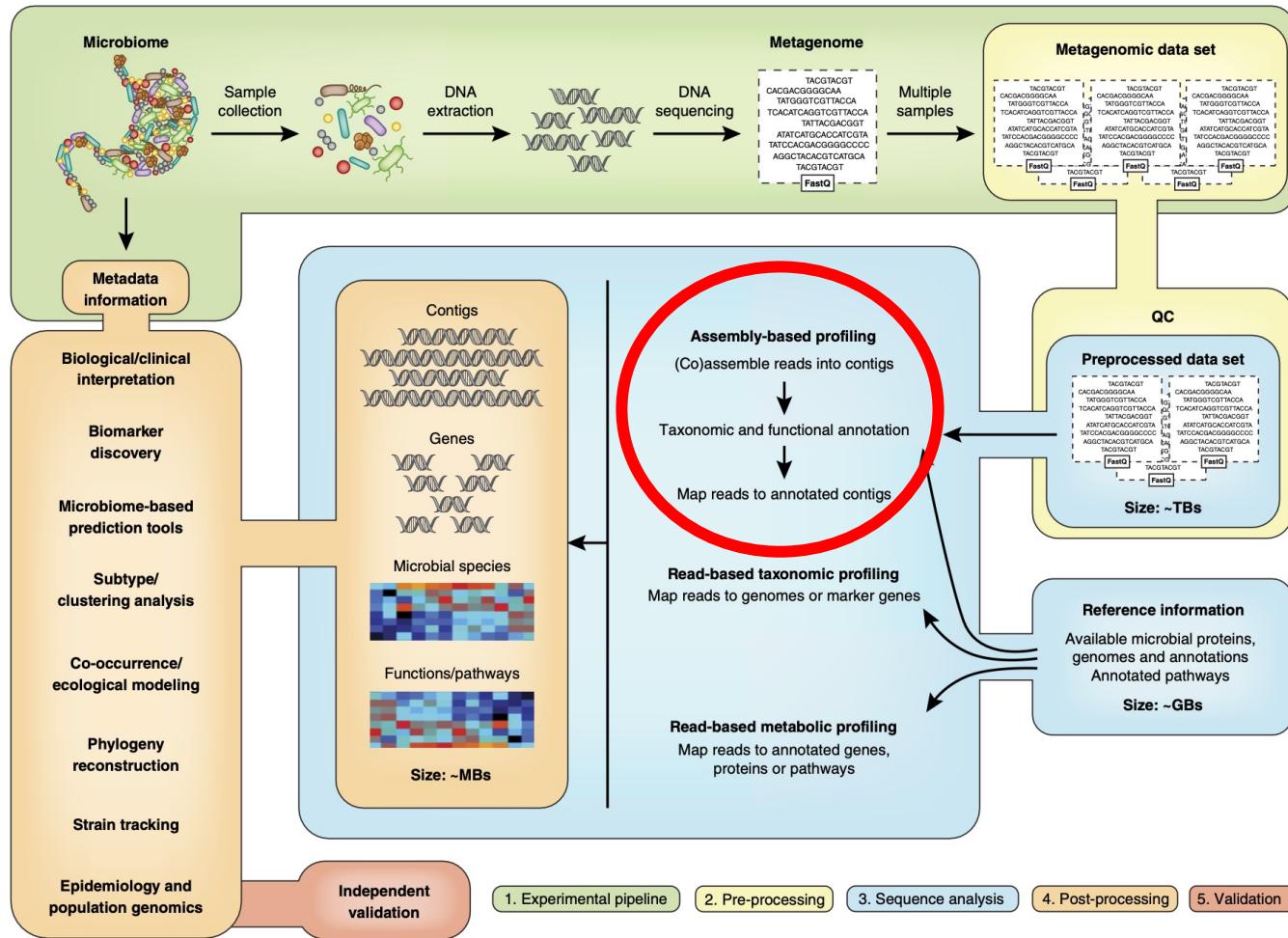


# Overview



## 3. Metagenome assembly and annotation

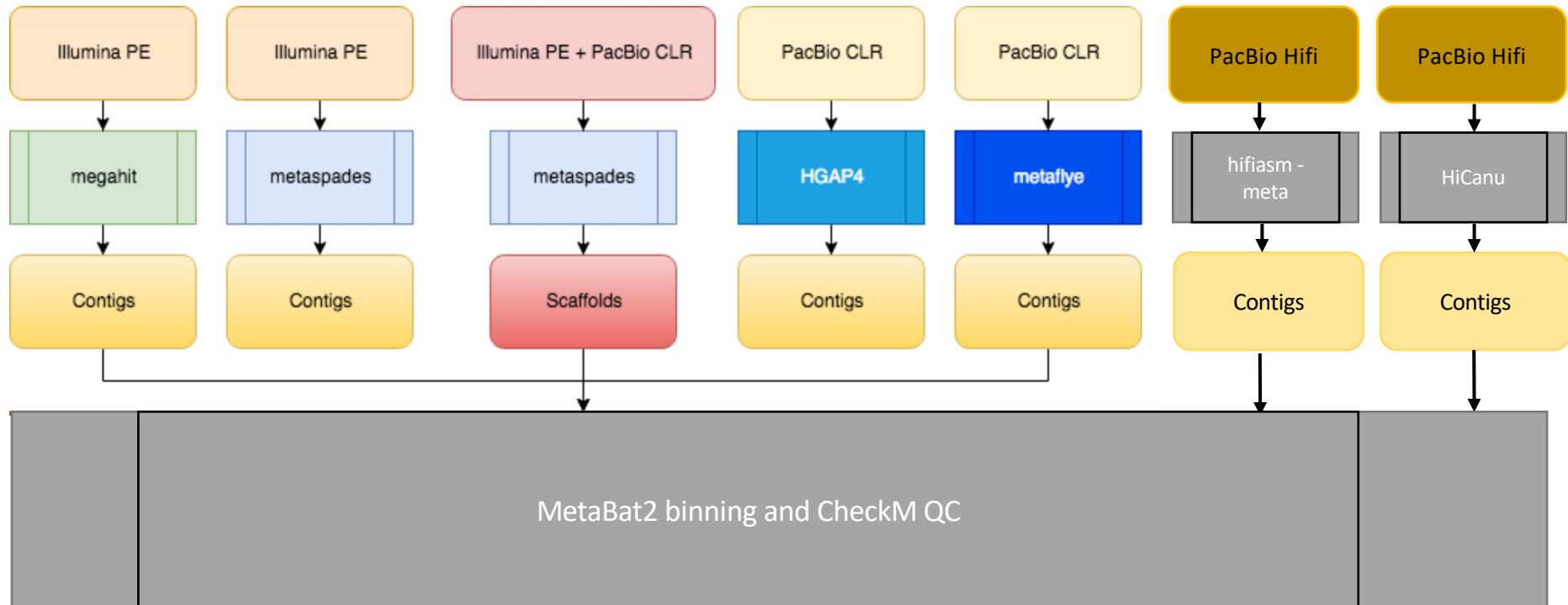
# Overview



## 3. Metagenome assembly and annotation

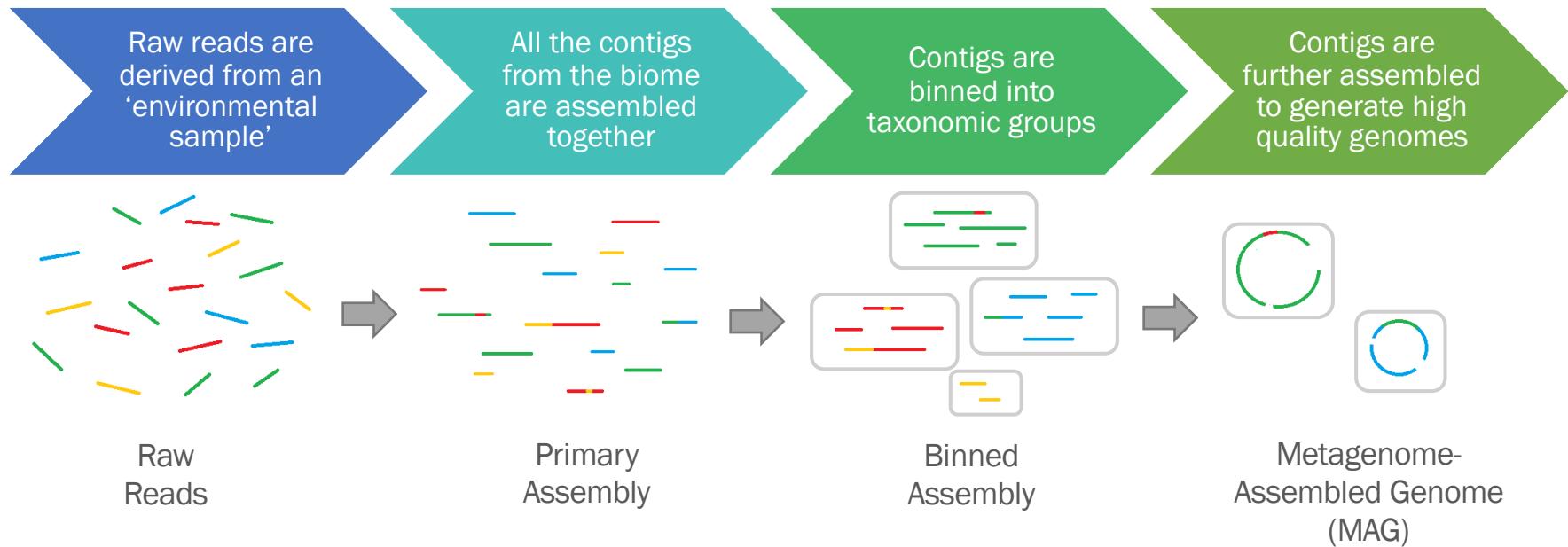


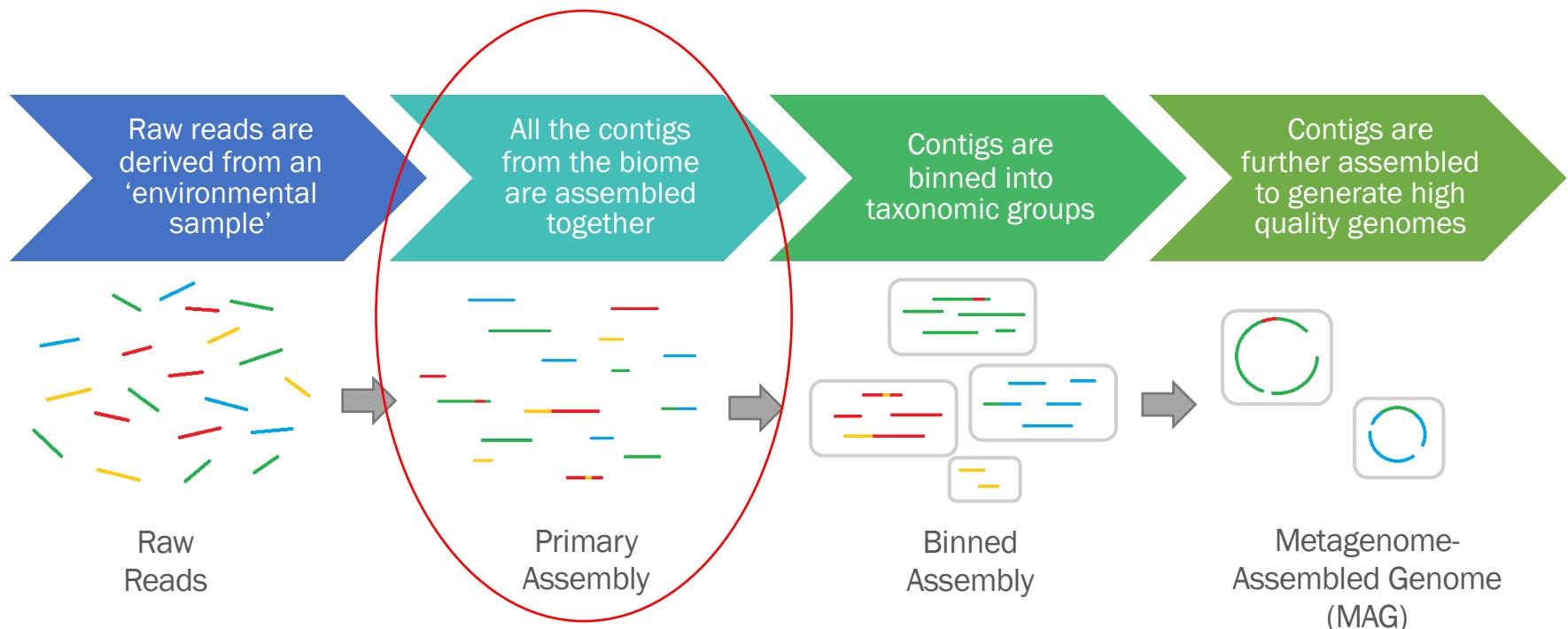
# Metagenome assembly and binning workflow



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

## Structuring Metagenomic Studies: Types Of Metagenomic Assembly





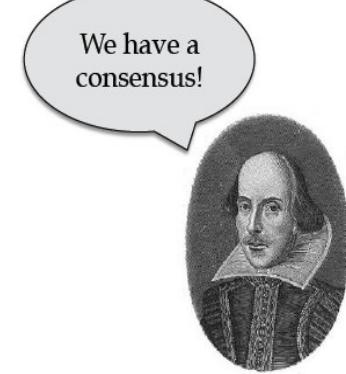
10  
01  
101101 1  
010 0  
0101 10...  
| ++ +++ 010 0101 01  
101 10  
010 01  
10 01  
0 1  
1

## Assembly – The Problem

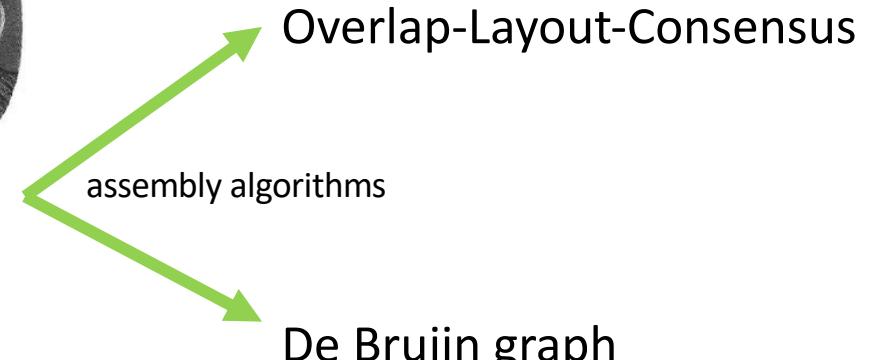
- **Reads**  
ds, Romans, count  
ns, countrymen, le  
Friends, Rom  
**send me your ears;**  
**crymen,** lend me

- **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;



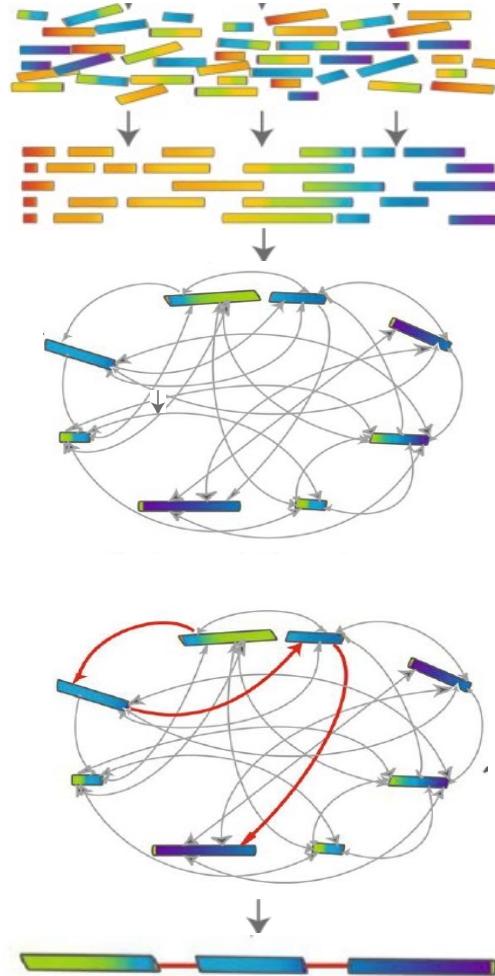
ds, Romans, count  
ns, countrymen, le  
crymen, lend me  
send me your ears;





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

## Overlap Layout Consensus - Overlap

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

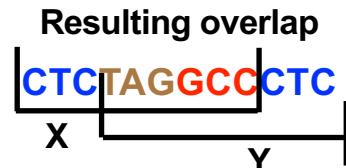
X: CTCTAGG**GCC**  
Y: TAGGCC**CTC**

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

Found it

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

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



- 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

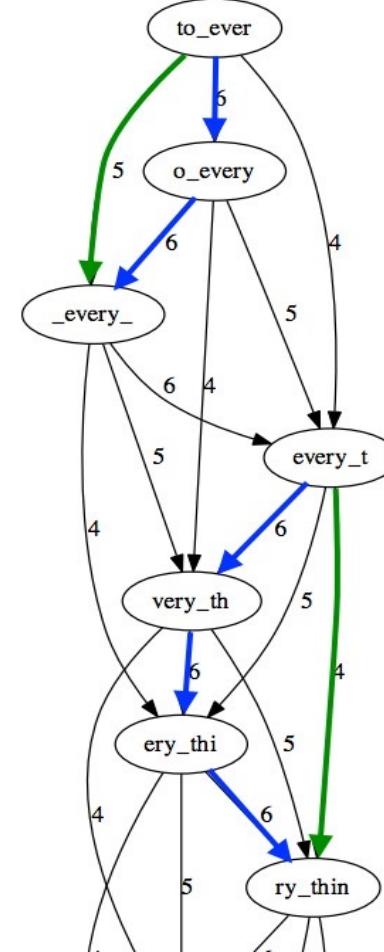
## Overlap Layout Consensus - Layout

### 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)

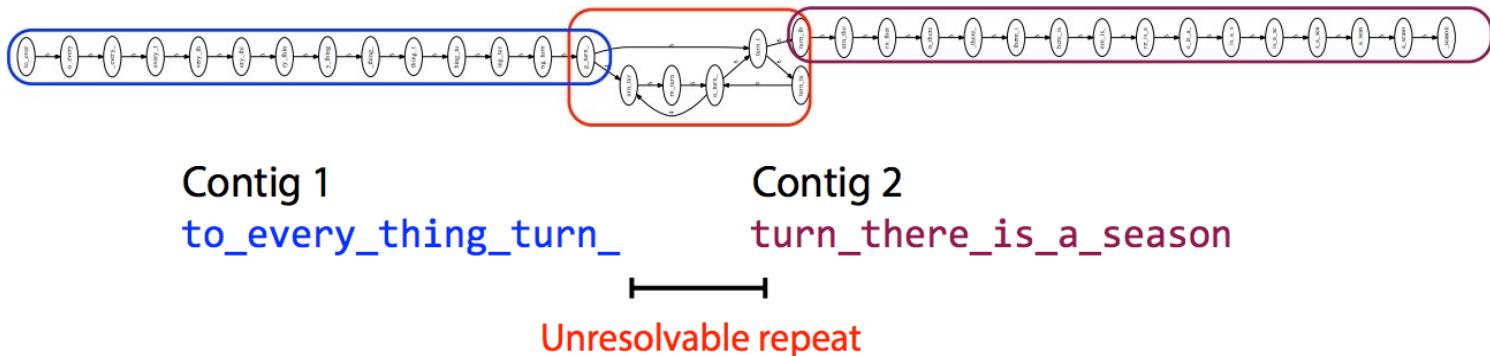




## Overlap Layout Consensus - Layout

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

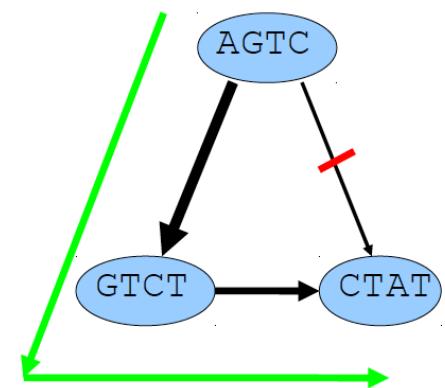
a path that visits each node exactly once

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  $(|V| - 1)!/2$ .

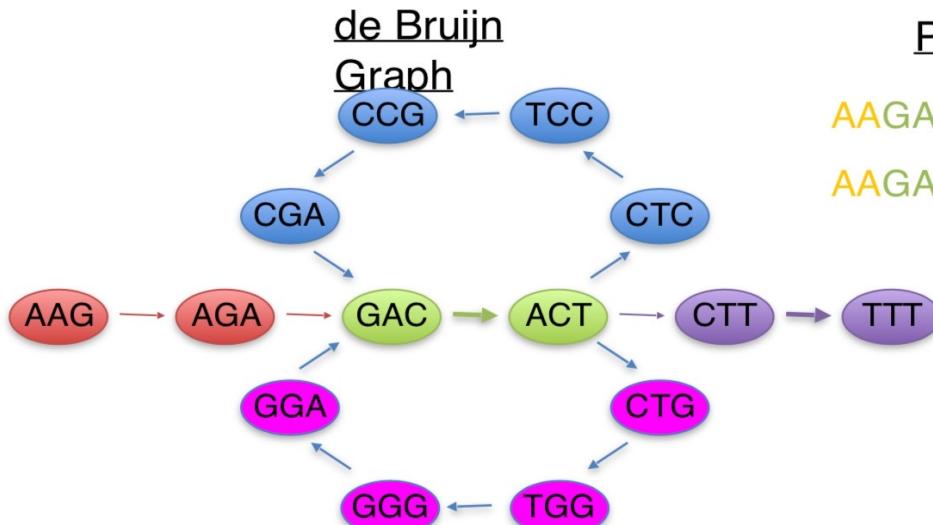
Heuristic algorithms generate solutions for large instances of reasonable quality

## De Bruijn 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  
...



### Potential Genomes

AAGACTCCGACTGGGACTTT  
AAGACTGGGACTCCGACTTT  
...

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

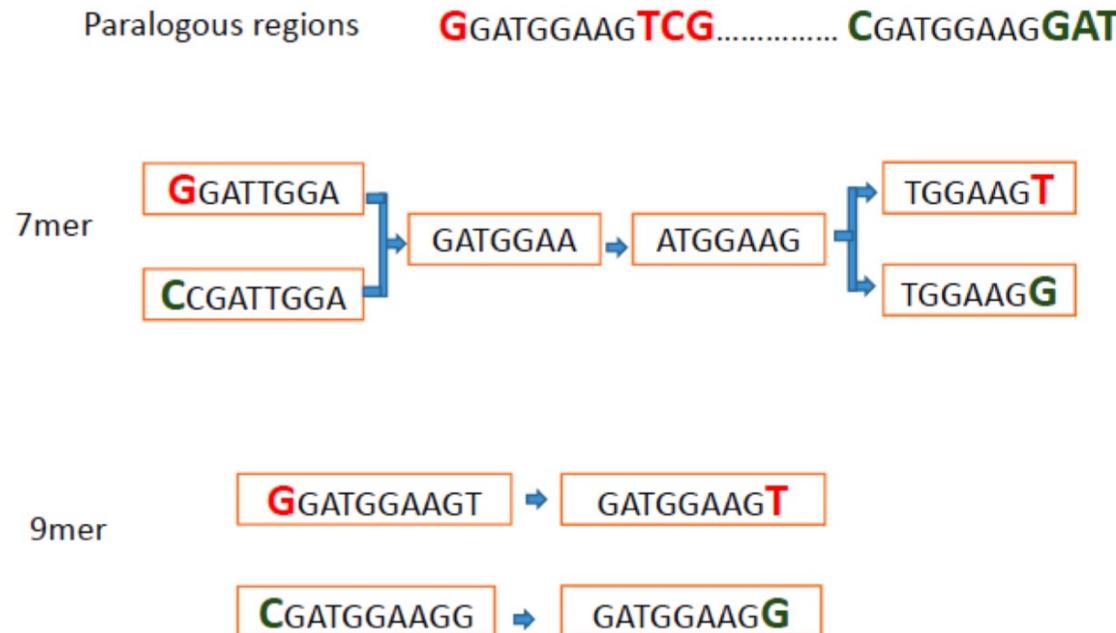
**A directed, connected graph is Eulerian if and only if it has at most 2 semi-balanced nodes and all other nodes are balanced**



## De Bruijn graph - DBG

### Impact of kmer length

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



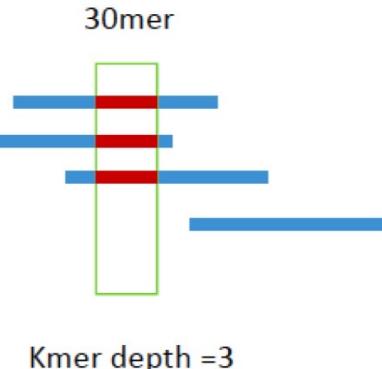
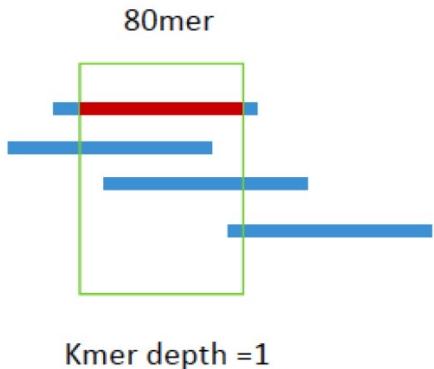


## De Bruijn graph - DBG

### 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

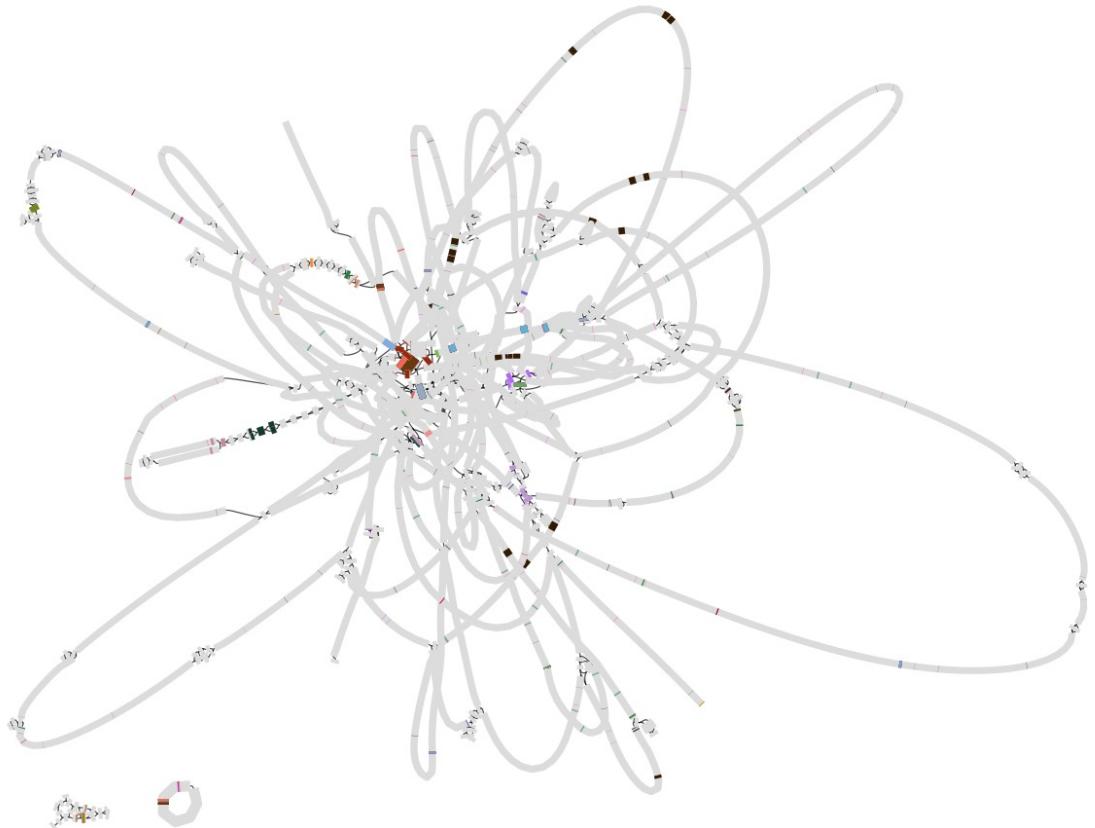


Read depth: 3



## 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





ETHzürich

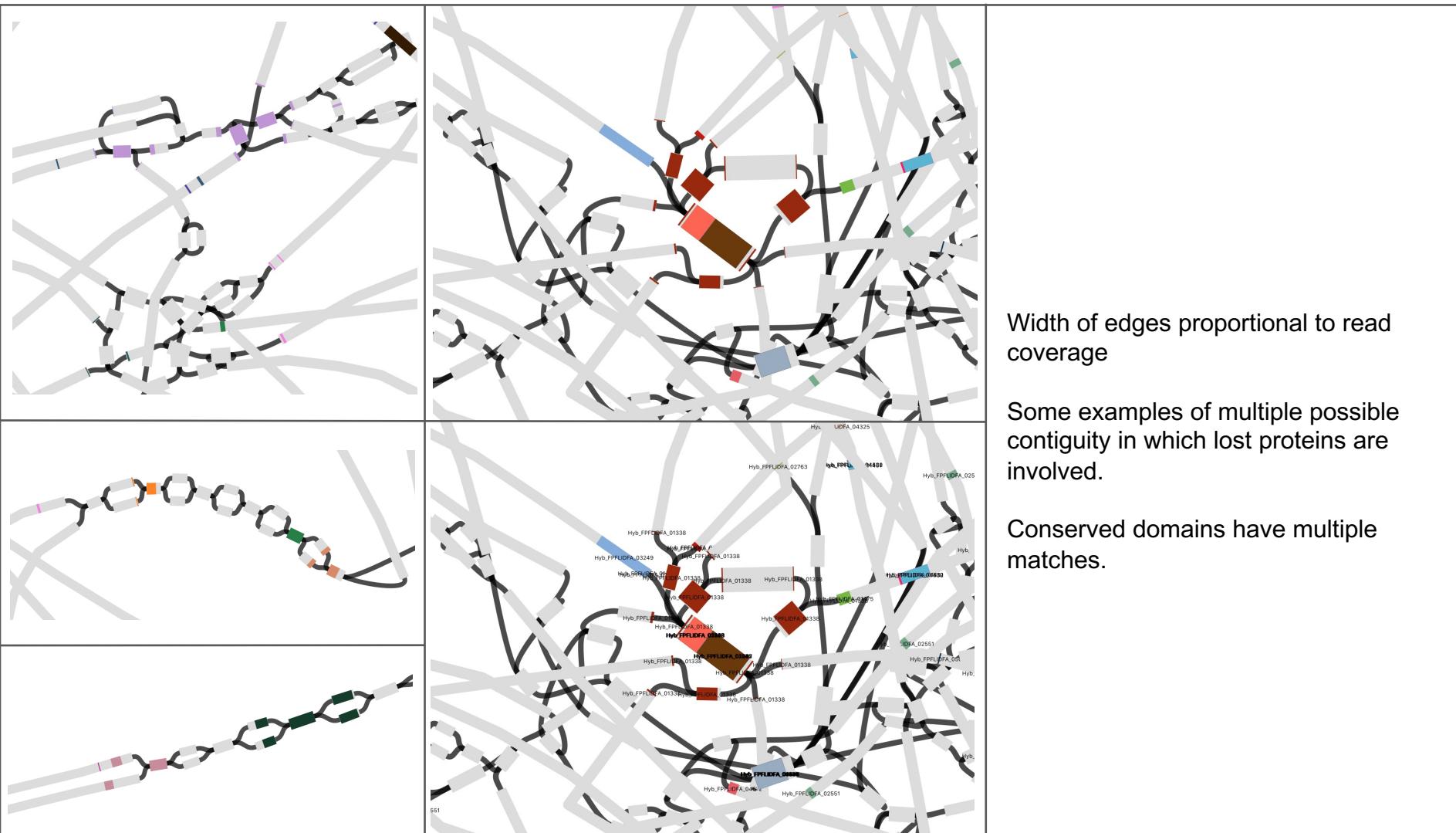
University of  
Zurich<sup>UZH</sup>

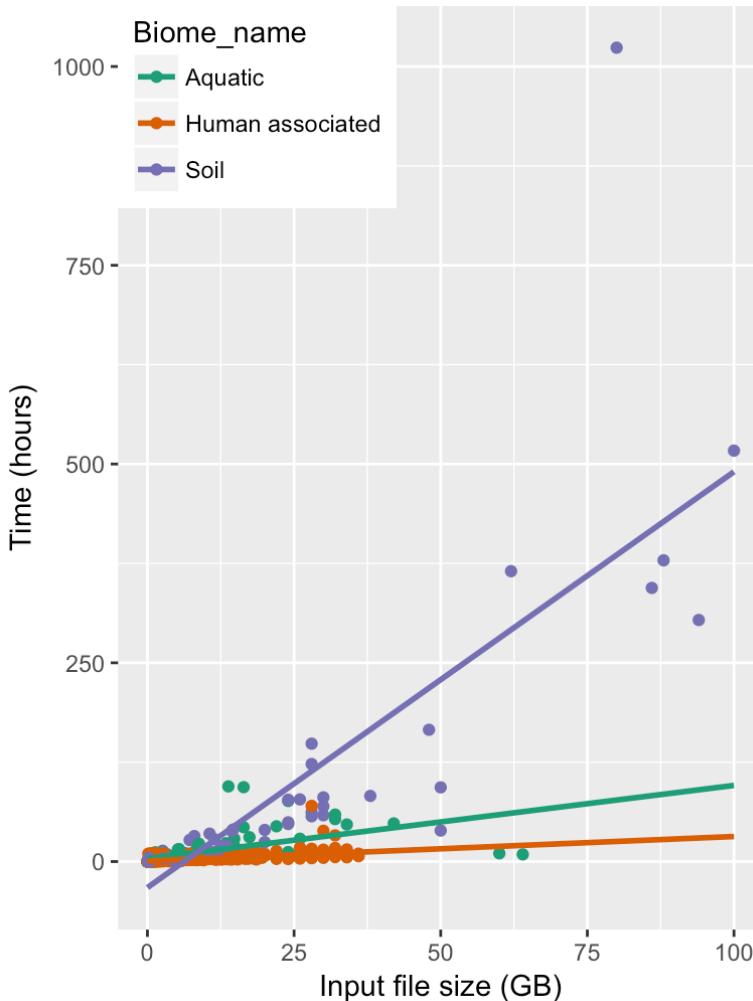
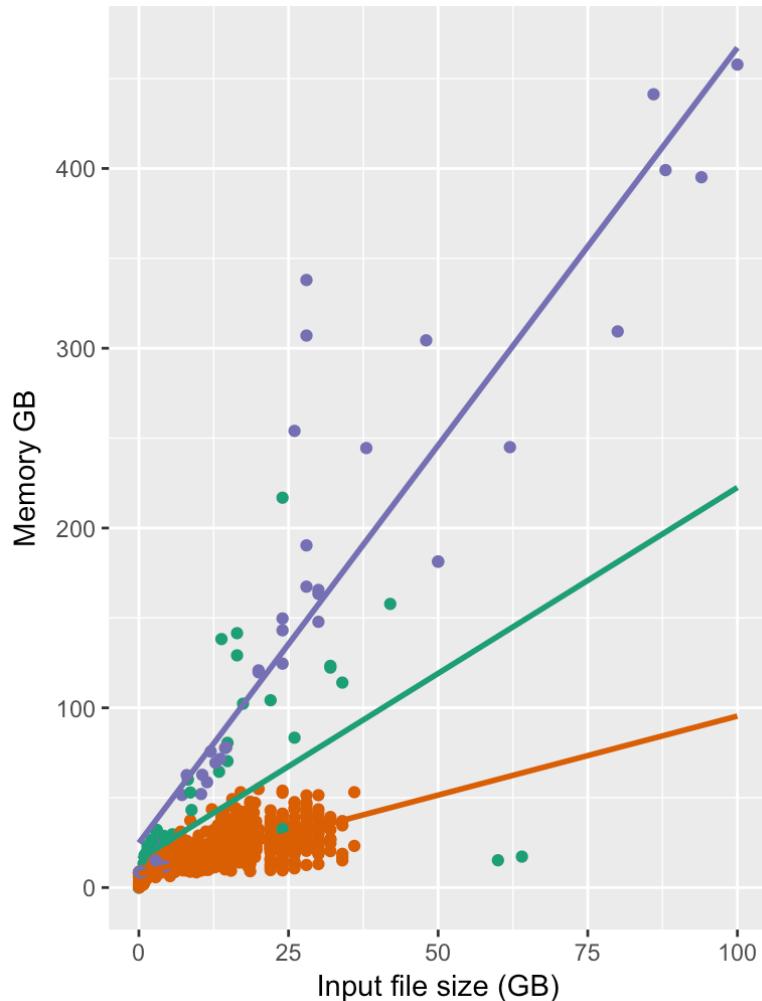
10  
01  
101



functional genomics center zurich

01 1  
10 0  
101 10  
010 01  
010 10  
f g c z  
+ 01 1

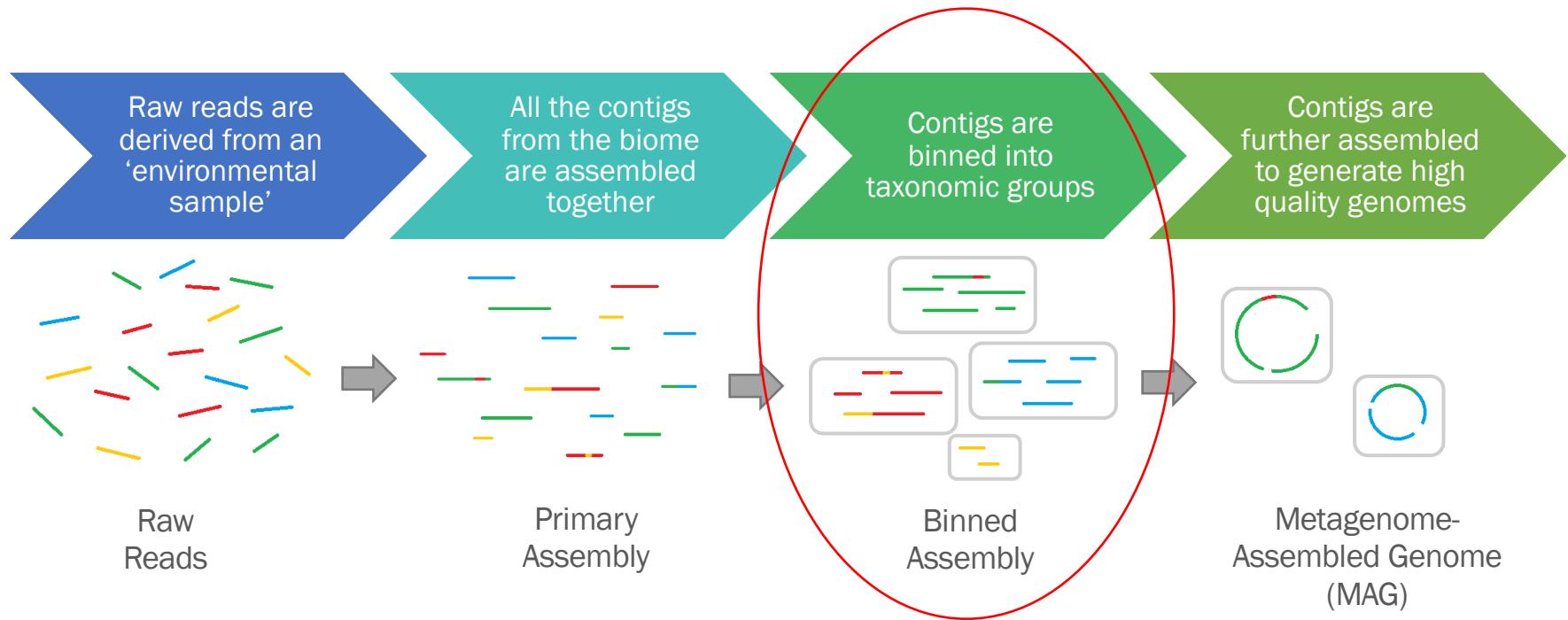




10  
01  
101

## 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





10  
01  
101

01 1  
01 01  
101 10  
10  
01 01  
01 01

## Binning contigs: Two ways

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

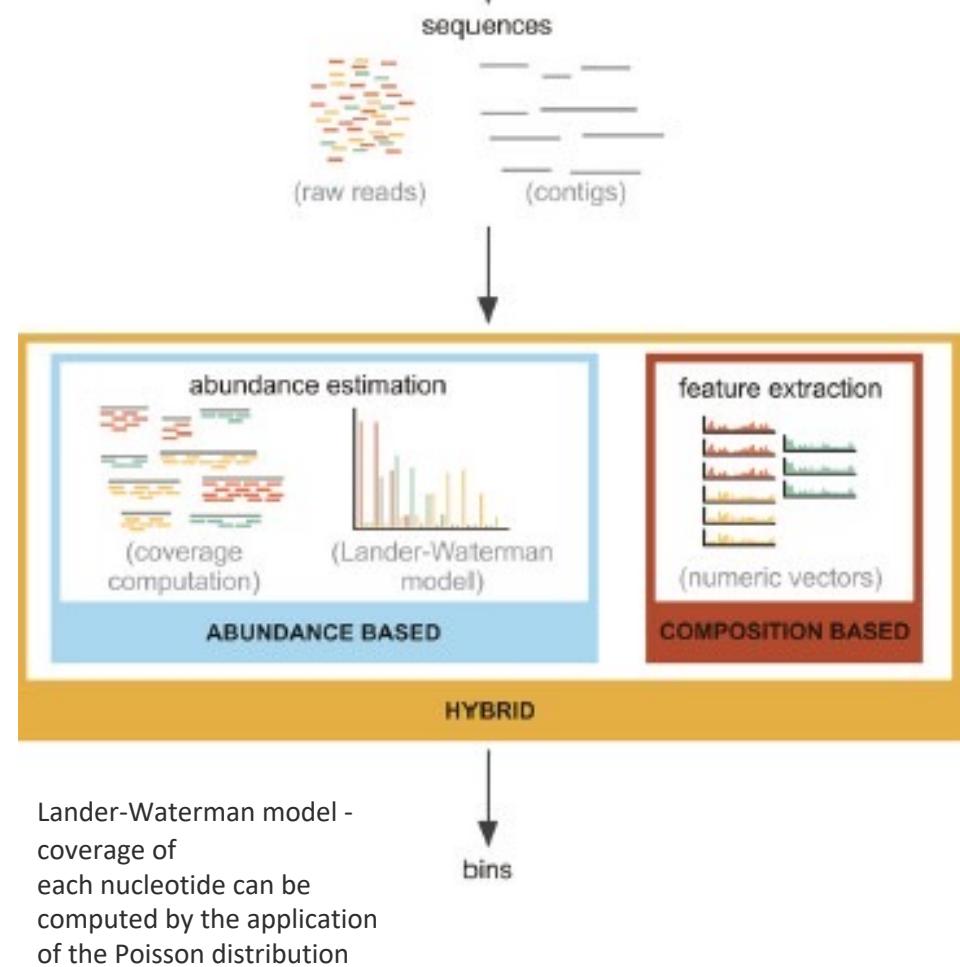
## Binning contigs: Taxonomy-independent binning

### 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

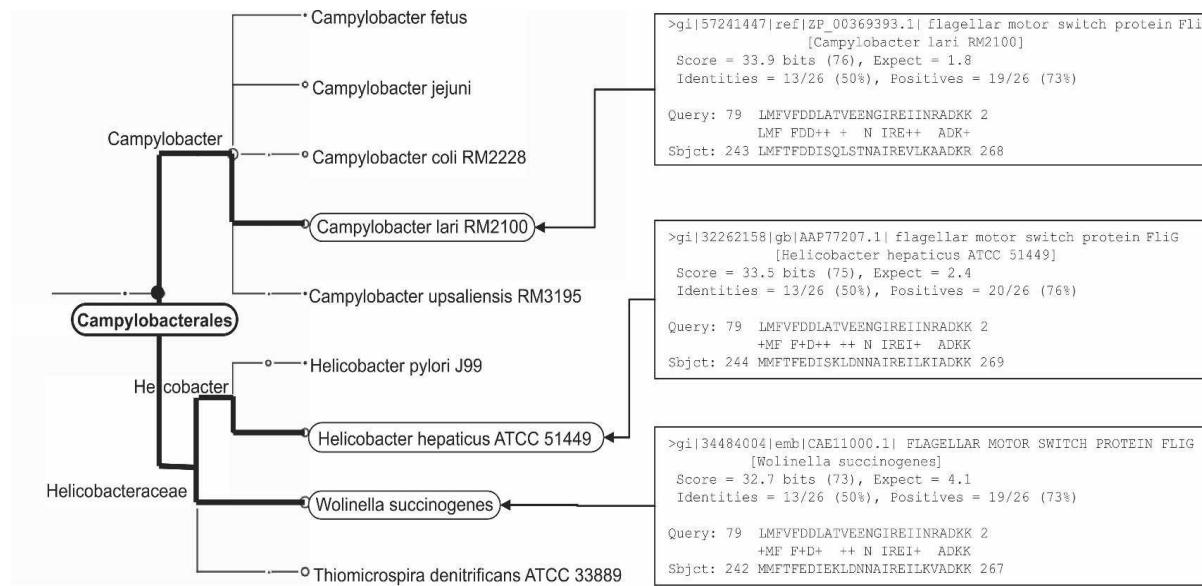
- Assumes sequence abundances as closest proxy for species similarity
- *De novo* assembly for initial contigs
- Abundance of contigs via read mapping
- Binning of contigs
- Concerned with k-mer abundance rather than content



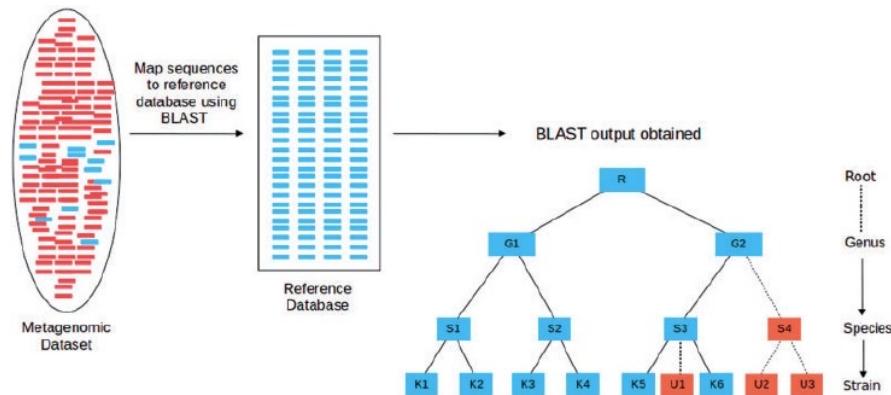
# Taxonomy dependent binning (LCA)

## 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)

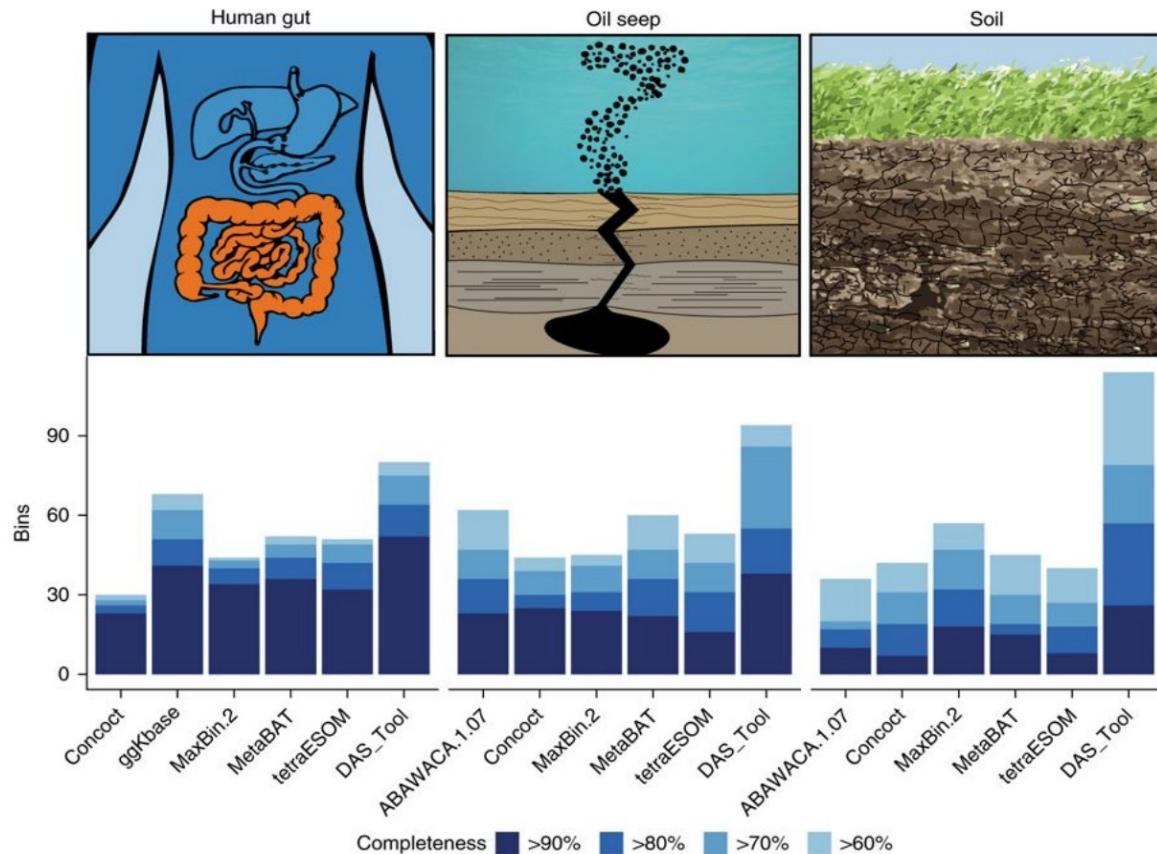
Matching bit-scores sometimes inadequate metric

Problematic with sample from mostly unexplored environments

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

## Binning contigs: Selection of methods

- Use multiple binning methods when possible



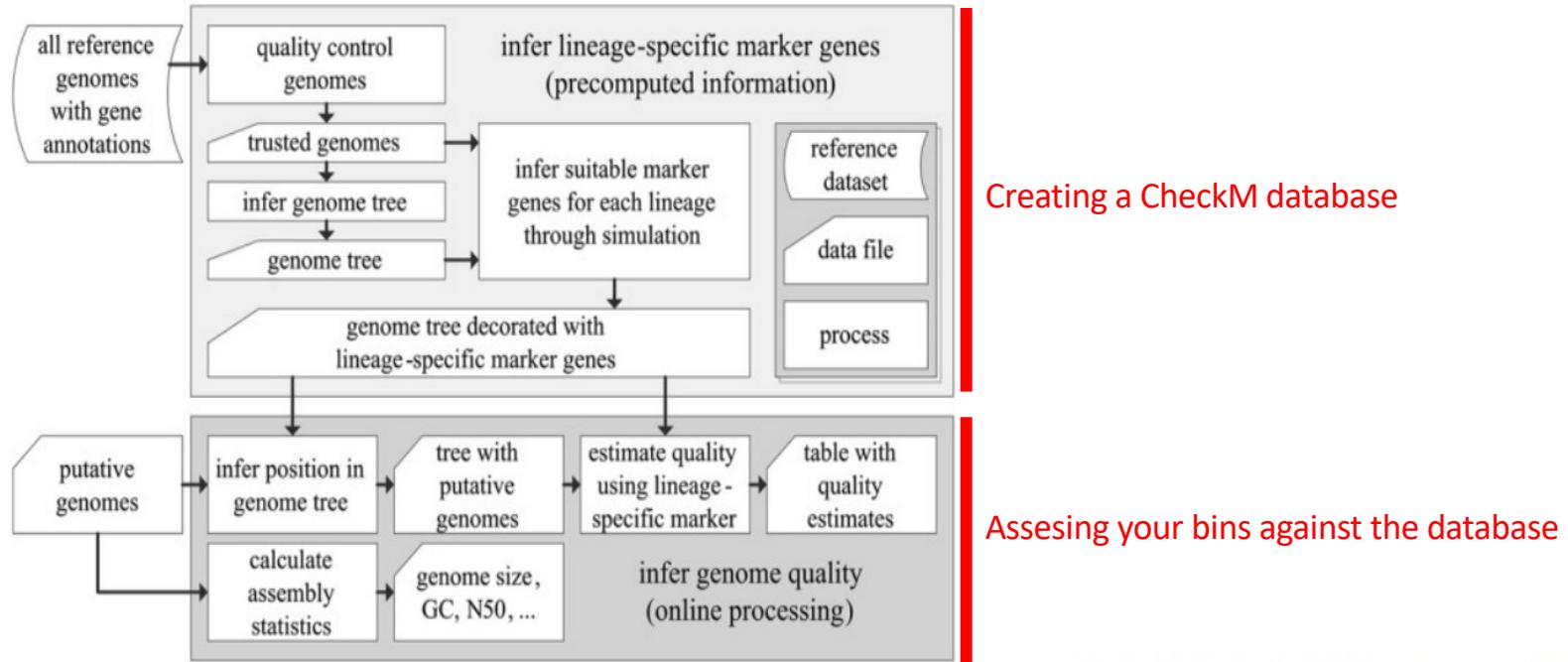


## Binning contigs

### 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

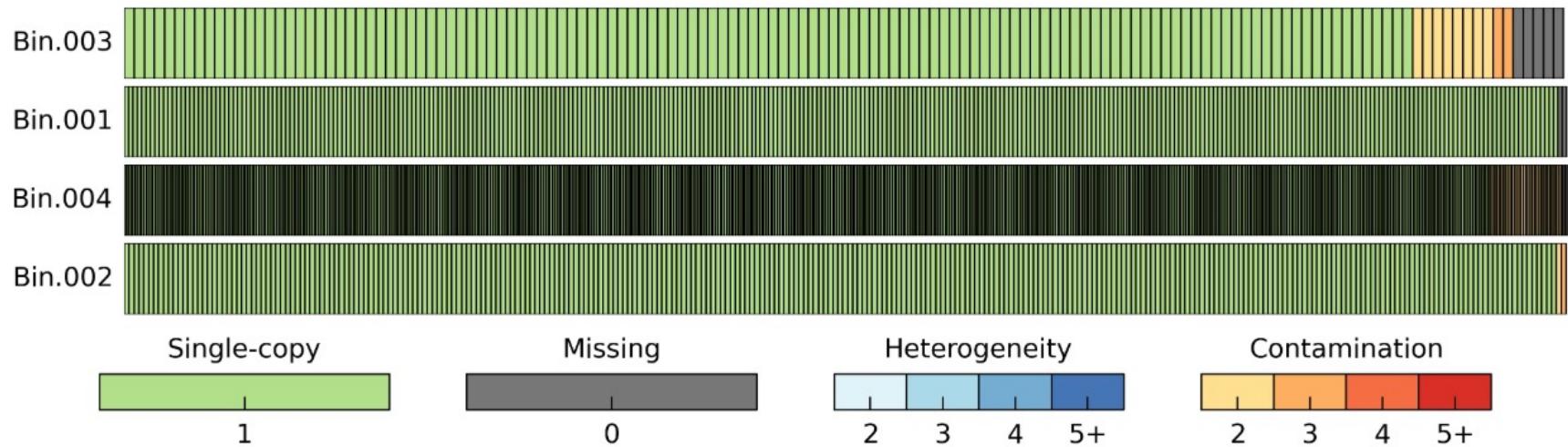


Parks DH et al. 2015. Genome Research



## Binning contigs: Assessing quality with CheckM

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





10  
01  
101

01 1  
01 01  
101 10  
10 01  
01 1

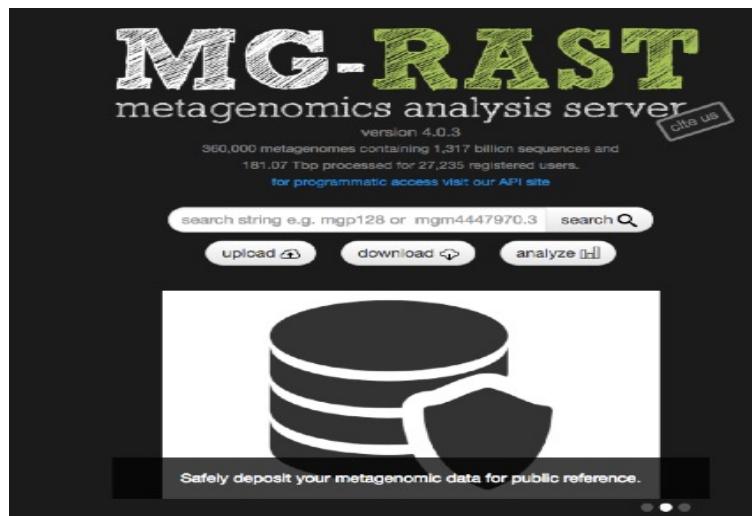
## What makes a good binned assembly

- 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

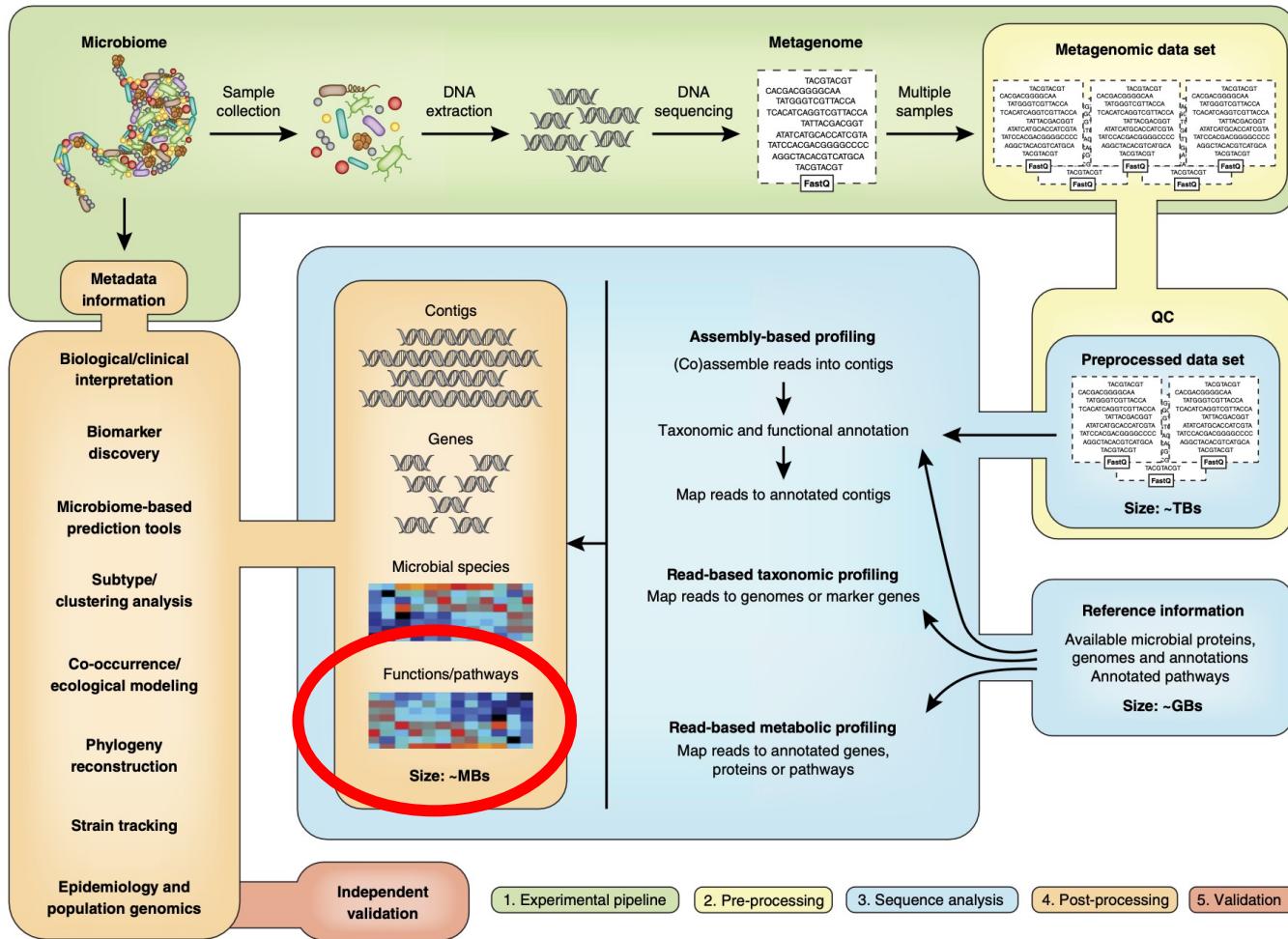
MG-RAST (GenBank annotation)  
(<https://www.mg-rast.org/>)



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.



# 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’.

## 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 logo features the text "MG-RAST" in large, bold, green, block letters with a textured, hand-drawn appearance. Below it, "metagenomics analysis server" is written in a smaller, black, sans-serif font. A small blue "cite us" button is in the top right corner. Below the main title, "version 4.0.3" is displayed. The text "360,000 metagenomes containing 1,317 billion sequences and 161.07 Tbp processed for 27,235 registered users." is followed by a link "for programmatic access visit our API site". A search bar contains the placeholder "search string e.g. mgfp128 or mgm4479/0.3" and a magnifying glass icon. Below the search bar are three buttons: "upload" with a cloud icon, "download" with a download icon, and "analyze" with a DNA helix icon. A large, stylized graphic of a database cylinder with a wedge removed from the side is centered below the buttons. At the bottom, the text "Safety deposit your metagenomic data for public reference." is displayed.

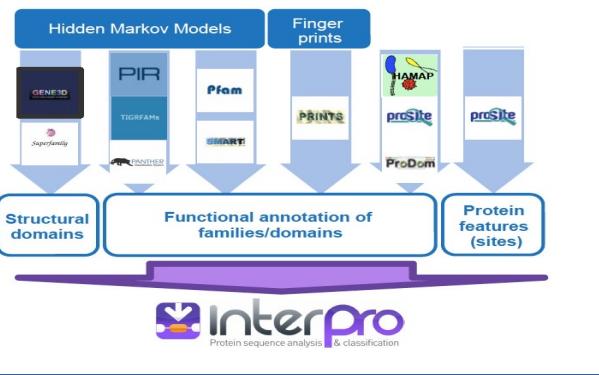
metagenome-atlas/  
**genecatalog\_atlas**

Extended genecatalog workflow for metagenome-  
atlas

1 Contributor    0 Issues    1 Star    0 Forks

## Annotation

## Annotation of protein domains (Interpro scan)



# Metabolic pathway prediction (KEGG)



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

The screenshot shows the EggNOG 4.5.1 web interface. On the left, a navigation sidebar includes links for Home, Sequence search, eggNOG-maps (with a red badge), Downloads, API, Methods, and Viral OGs. The main content area features a banner for 'eggNOG 5.0 now available' and a section titled 'Discover EggNOG 4.5.1' which describes it as a database of orthologous groups and functional annotation. It displays statistics: 2,031 Organisms, 352 Viruses, 190k Orthologous Groups, and 1.9M Trees & Algs. Below this is a search bar with a 'Search' button. On the right, there's an 'Enrichment analysis' tool with a text input for gene IDs, dropdown menus for biological process (set to 'biological process') and species ('Homo sapiens'), and a 'Submit' button. At the top right, there's a header for 'Gene Ontology Consortium' with links for Home, Documentation, Downloads, Tools, About, and Contact.

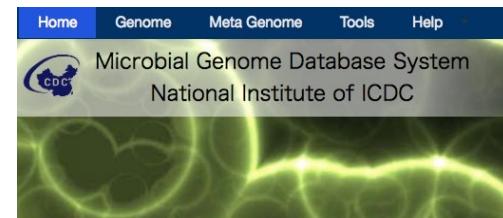
10  
01  
101010 01  
101 10  
010 01  
10  
01  
1

# Annotation

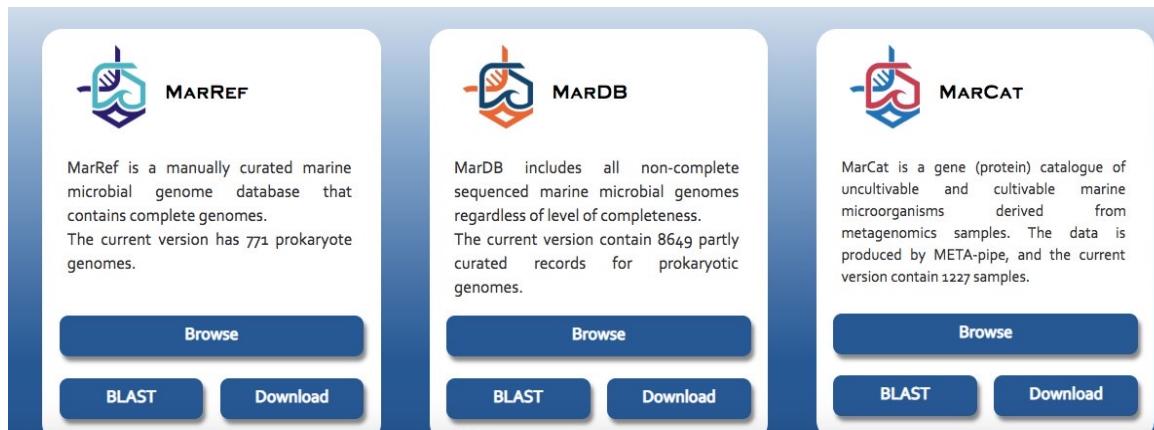
## Using ecosystem-specific databases for annotation and taxonomy



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, with "International Soil Metagenome Sequencing Consortium" below it. To the left of the text is a circular logo composed of a DNA helix and soil particles. Below the title is a photograph of a field with young plants.



The screenshot shows the Microbial Genome Database System homepage. The header includes links for Home, Genome, Meta Genome, Tools, and Help. The main title "Microbial Genome Database System" is displayed in large white text, with "National Institute of ICDC" below it. To the left of the text is the logo for the Chinese Center for Disease Control and Prevention (CDC).



The screenshot shows the interface for three marine microbial databases: MarRef, MarDB, and MarCat. Each section has a logo and a title. Below each title are four buttons: "Browse", "BLAST", "Download", and another "Browse" button. The MarCat section also includes a circular logo for the "Microbial Genome Database".

- 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](#)

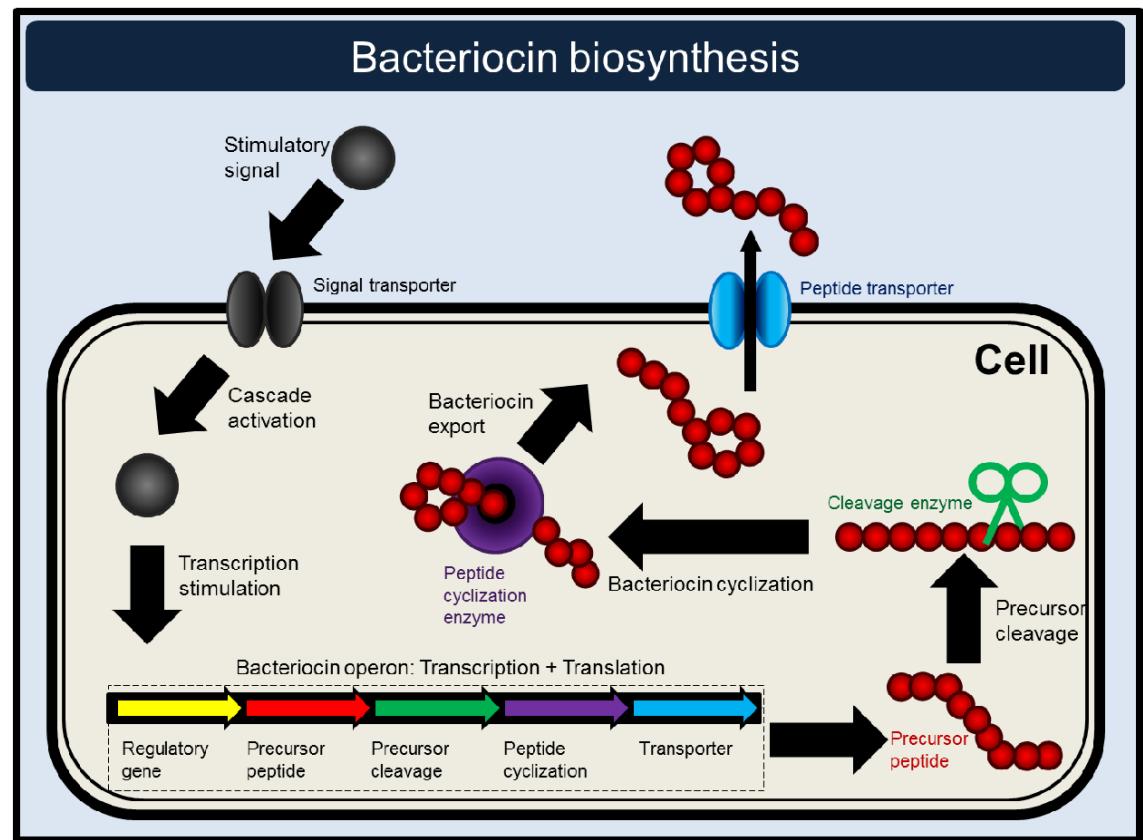
# Annotation

## Biosynthetic Gene Clusters

**Biosynthetic gene clusters (BGCs)** are a locally clustered group of two or more genes that together encode a biosynthetic pathway for the production of a secondary metabolite.

Classes of BGCs:

- peptide synthetases (NRPS)
- polyketide synthases (PKS)
- bacteriocins





# Annotation

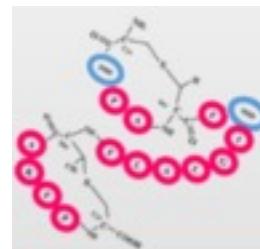
## Biosynthetic Gene Clusters

**Biosynthetic gene clusters (BGCs)** are a locally clustered group of two or more genes that together encode a biosynthetic pathway for the production of a secondary metabolite.



### Classes of BGCs:

- peptide synthetases (NRPS)
- polyketide synthases (PKS)
- bacteriocins



**RiPPMiner**  
A Bioinformatics Resource for Deciphering Chemical Structures of RiPPs



# Annotation

## Biosynthetic Gene Clusters

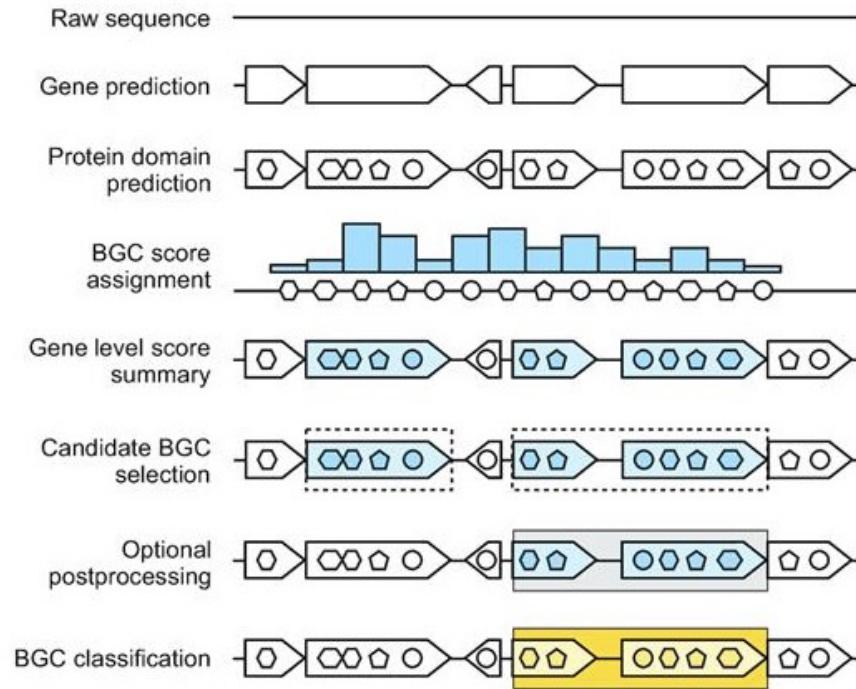
**Biosynthetic gene clusters (BGCs)** are a locally clustered group of two or more genes that together encode a biosynthetic pathway for the production of a secondary metabolite.

### Classes of BGCs:

- peptide synthetases (NRPS)
- polyketide synthases (PKS)
- bacteriocins



### deepbgc





# Annotation

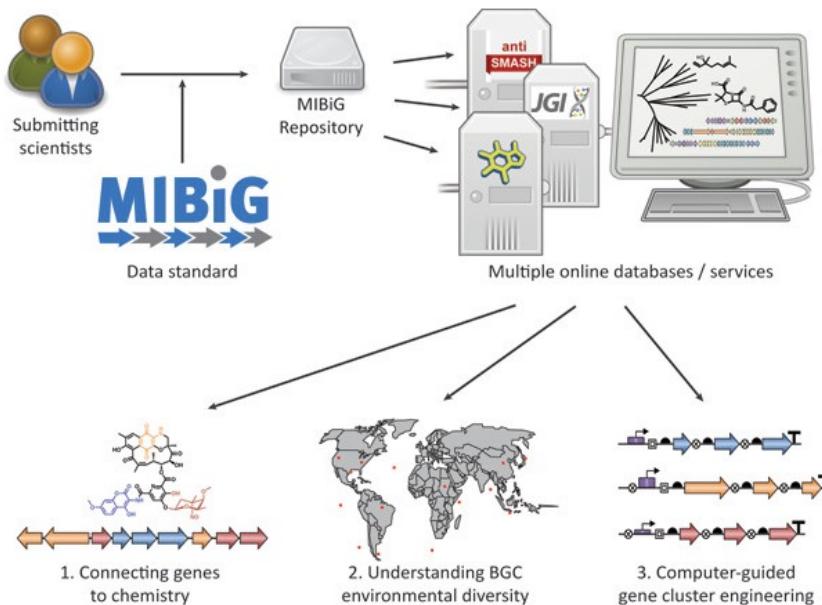
## Biosynthetic Gene Clusters



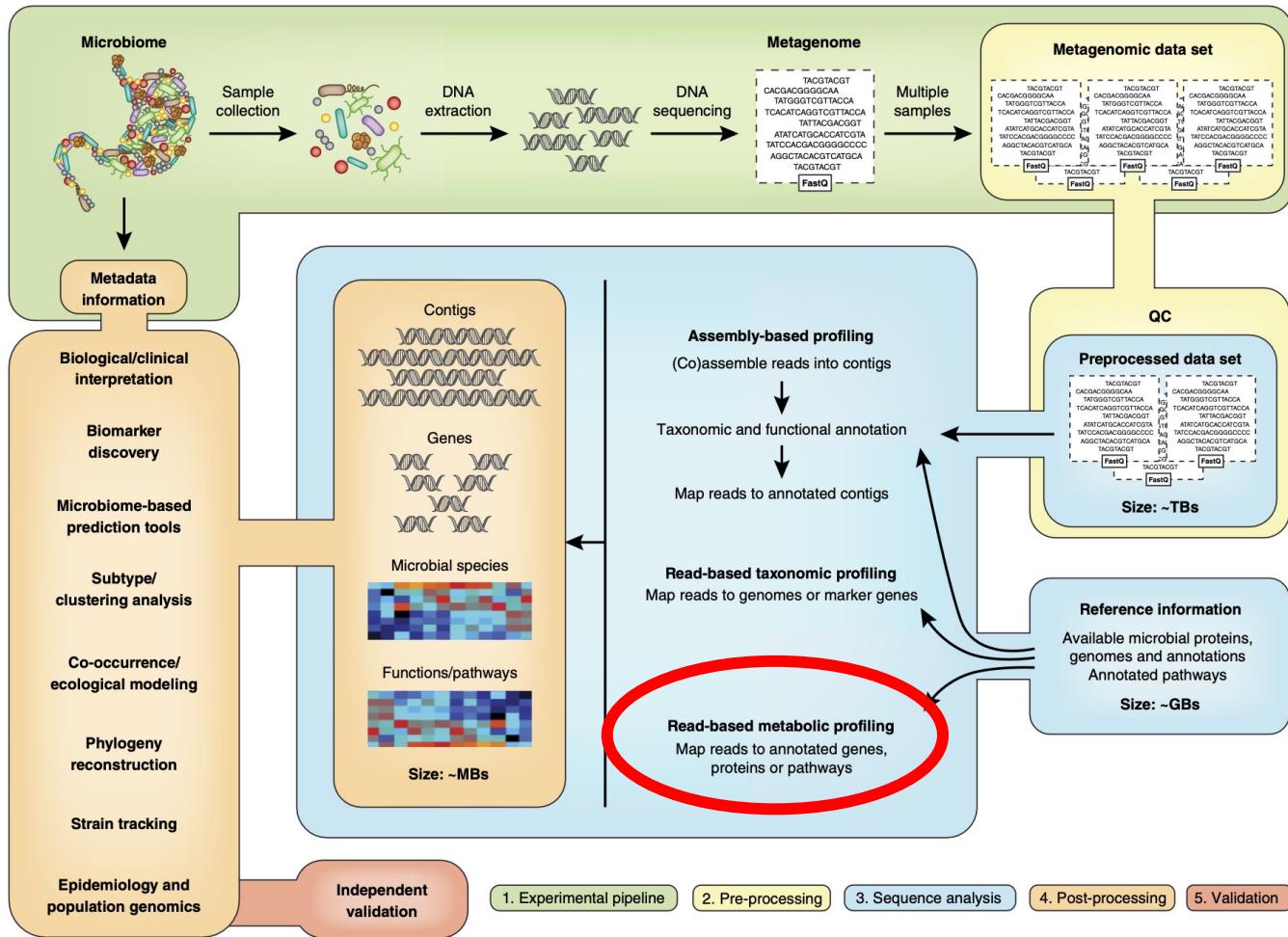
Minimum Information about a Biosynthetic Gene cluster

Minimum Information about a Biosynthetic Gene cluster (MIBiG):

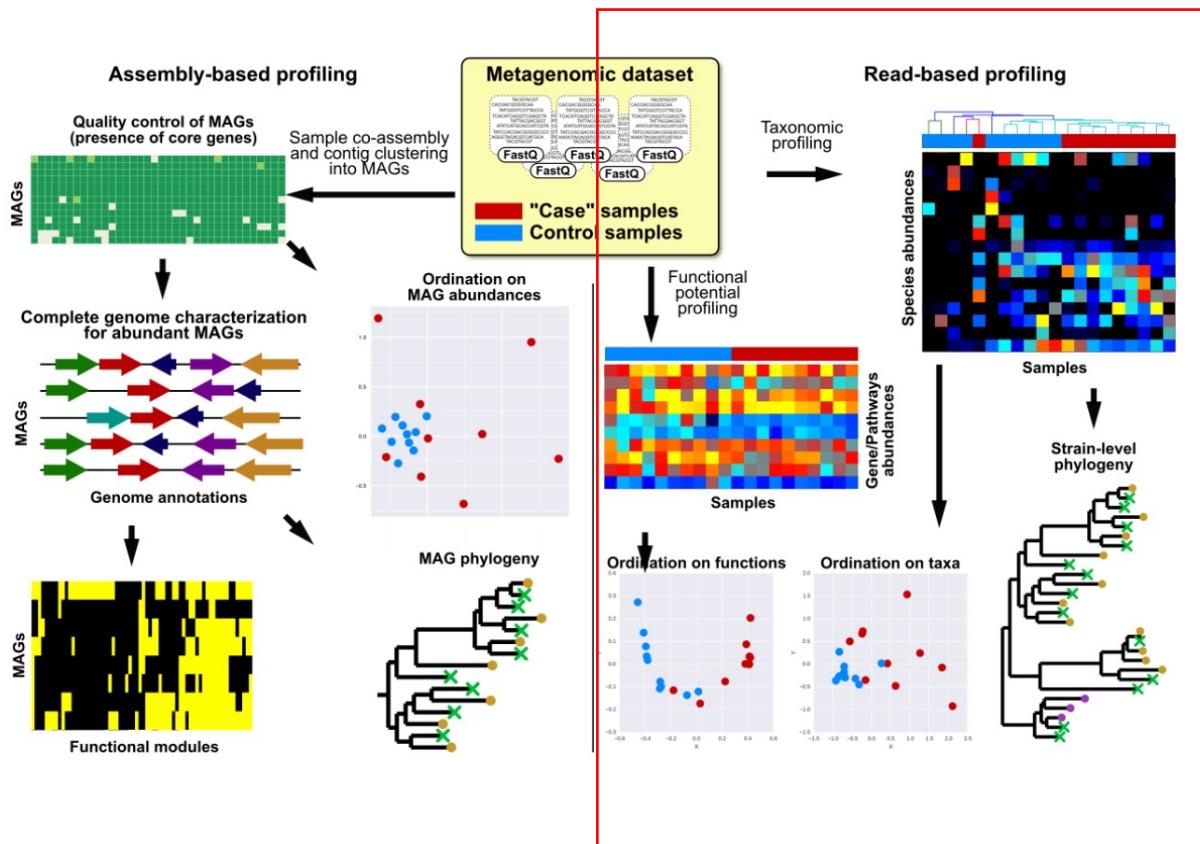
- facilitate consistent and systematic deposition and retrieval of data on biosynthetic gene clusters



# Overview



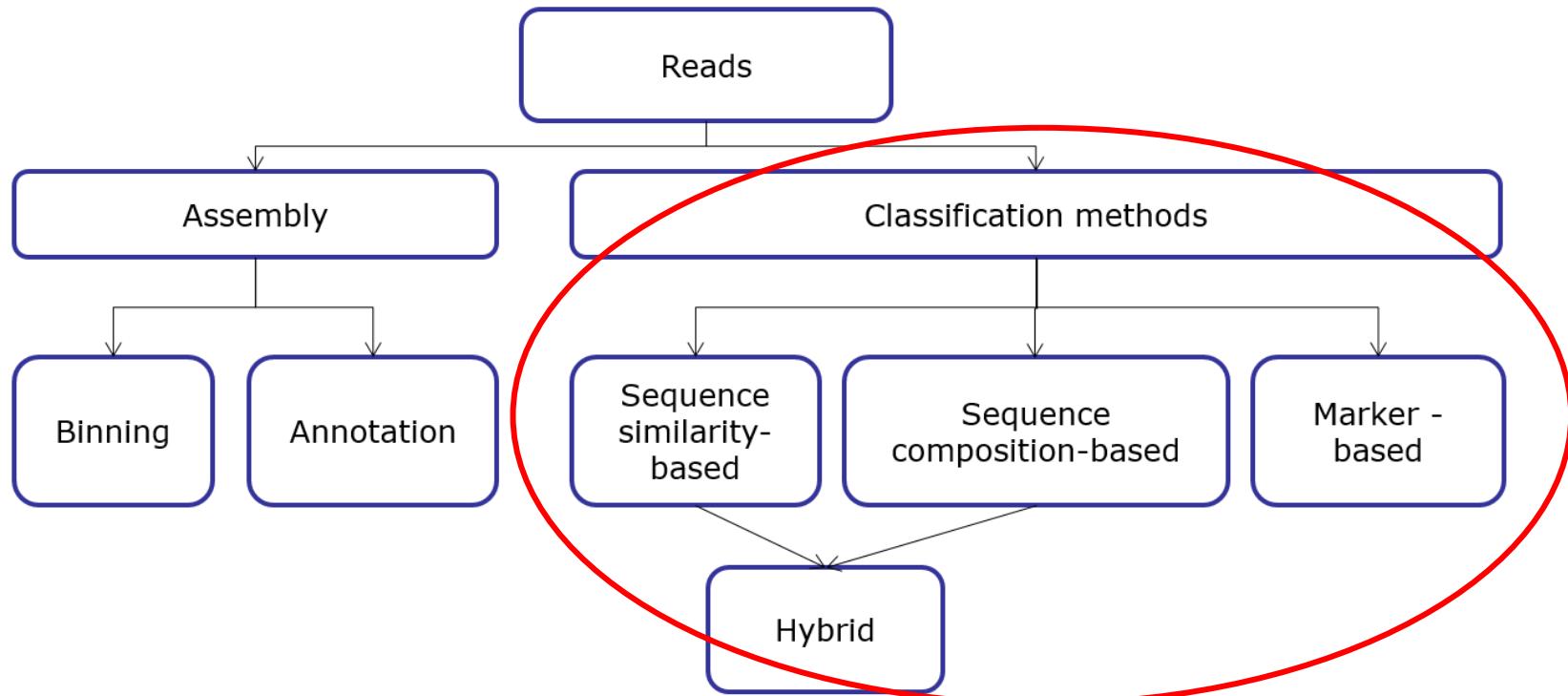
## Assembly-free approach



## 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

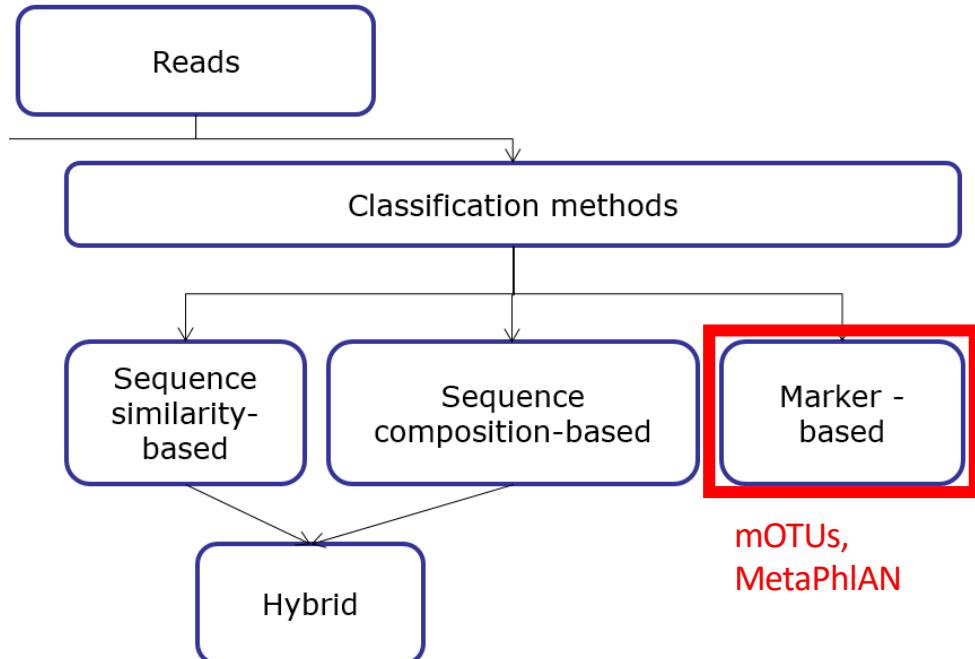


10  
01  
101101 1  
010 0  
0101 10010 01  
101 10  
010 01  
10 0  
01 1

## MetaPhlAn 3.0

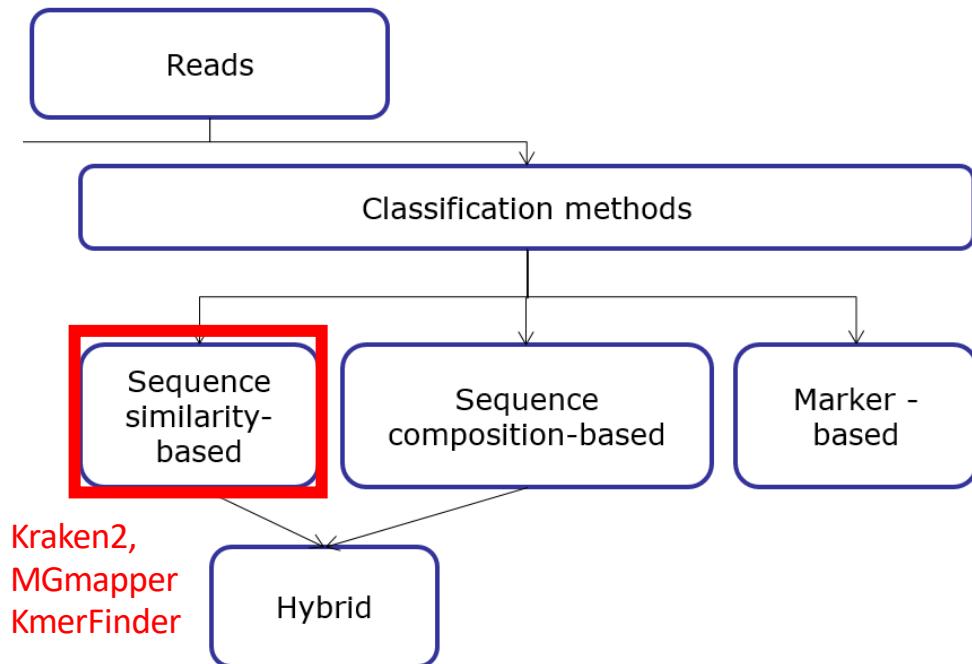
(<http://huttenhower.sph.harvard.edu/metaphlan/>)

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





# Assembly-free approach: taxonomy

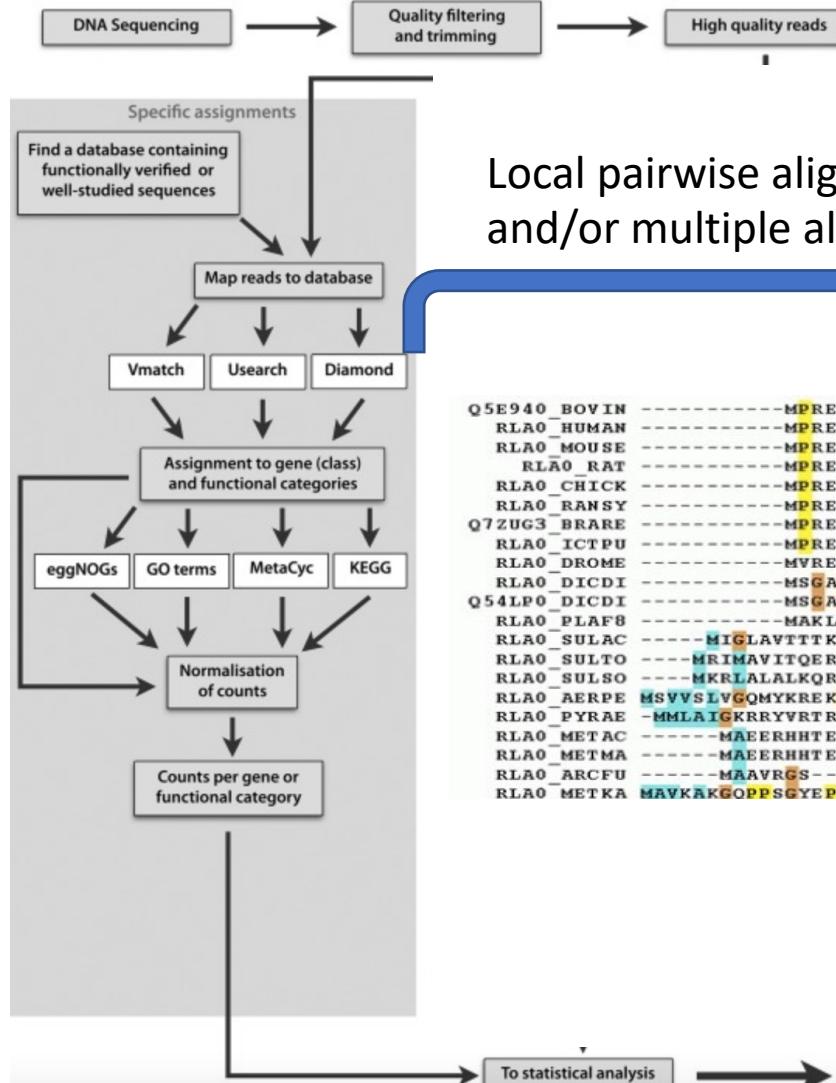


(<https://ccb.jhu.edu/software/kraken2/>)

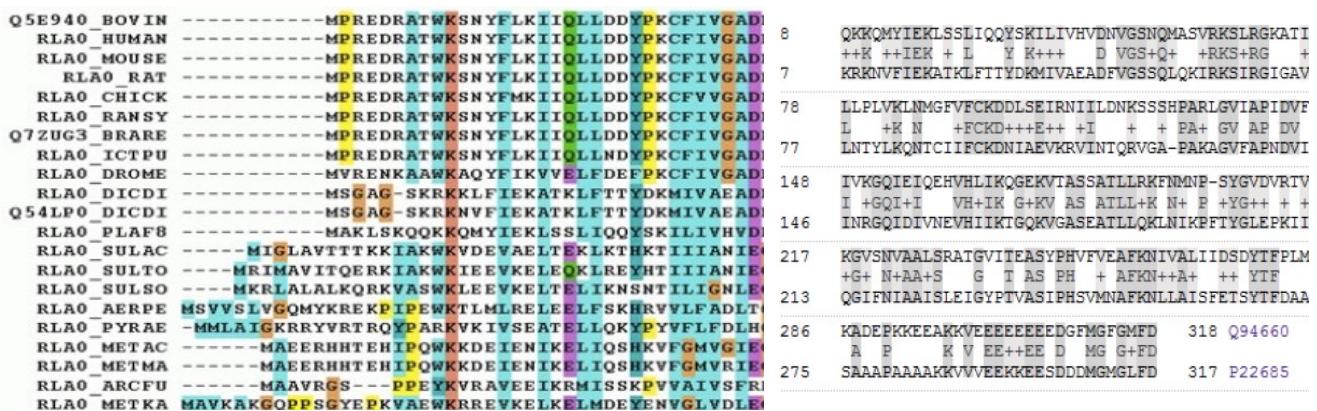


KmerFinder 3.2  
(<https://cge.cbs.dtu.dk/services/KmerFinder/>)

# Assembly-free approach: function



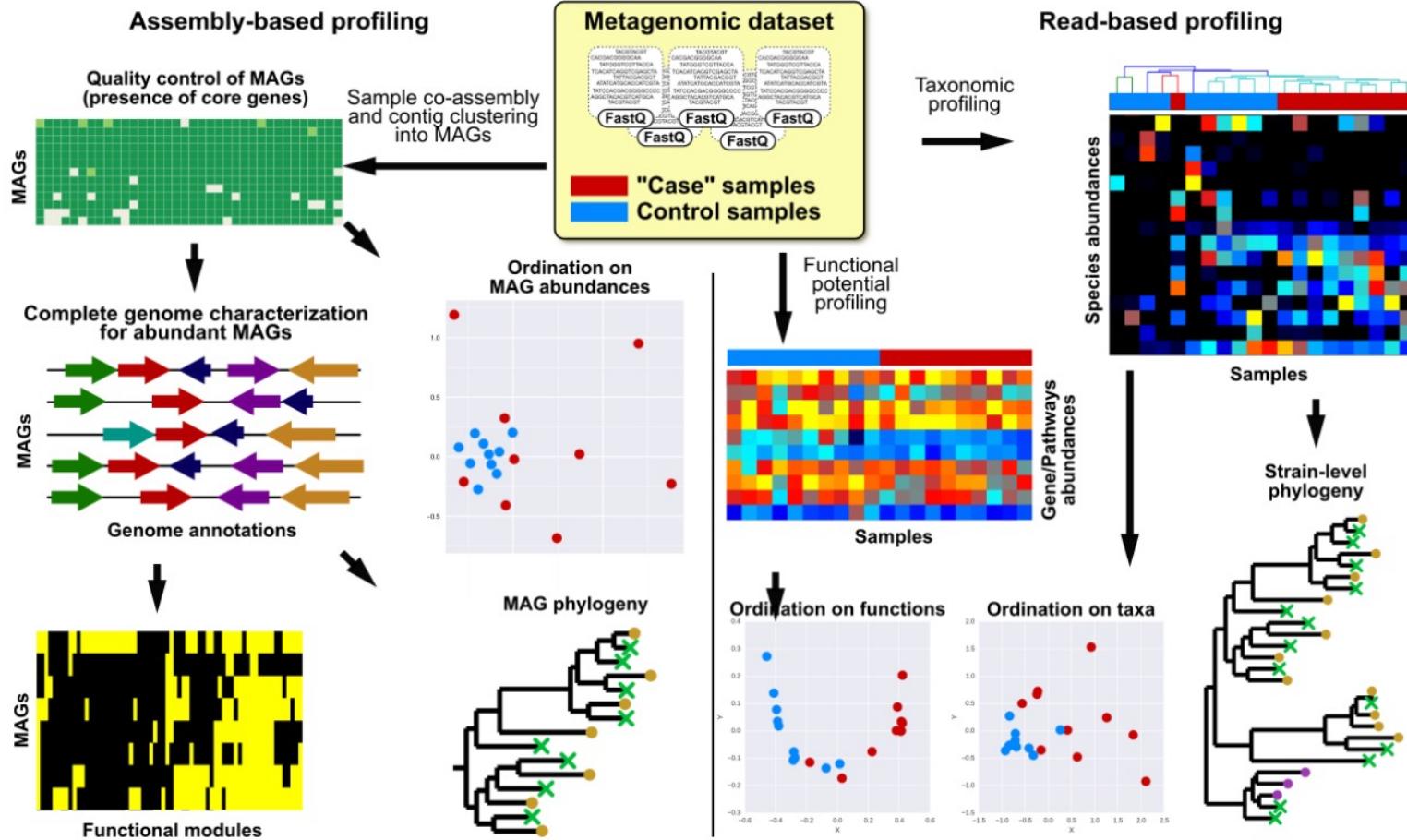
## Local pairwise alignment and/or multiple alignment



The figure displays a sequence alignment visualization. On the left, a multiple sequence alignment of various organisms (e.g., BOV, HUMAN, MOUSE, RAT, CHICK, RANSY, BRARE, ICTPU, DROME, DICDI, PLAF8, SULAC, SULTO, SULSO, AERPE, PYRAE, METAC, METMA, ARCFU, METKA) is shown. The sequences are color-coded by residue type. On the right, individual local pairwise alignments are shown for specific positions (e.g., 8, 7, 78, 77, 148, 146, 217, 213, 286, 275). Each alignment shows two sequences with matching residues highlighted in blue and green, and non-matching residues in black. Above each alignment, the position number is indicated.



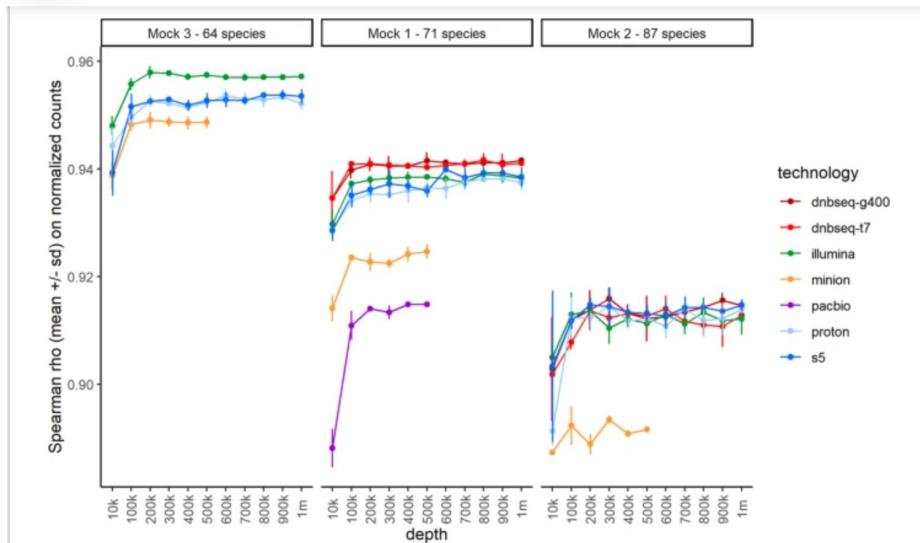
# Conclusion



# 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 (metafiley)	PacBio Sequel II (metafiley)
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	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



# Snakemake

## Create rules

```
rule plot:  
    input:  
        "raw/{dataset}.csv"  
    output:  
        "plots/{dataset}.pdf"  
    shell:  
        "somecommand {input} {output}"
```

# Install dependencies automatically

```
channels:-
  - bioconda|
  - r|
dependencies:-
  - python=2.7|
  - checkm-genome=1.0.7|
  - prodigal >=2.6.1|
```

# Atlas workflow

Sample1

Sample2

Sample

1. QC
2. Assembly
3. Gene prediction

Genes

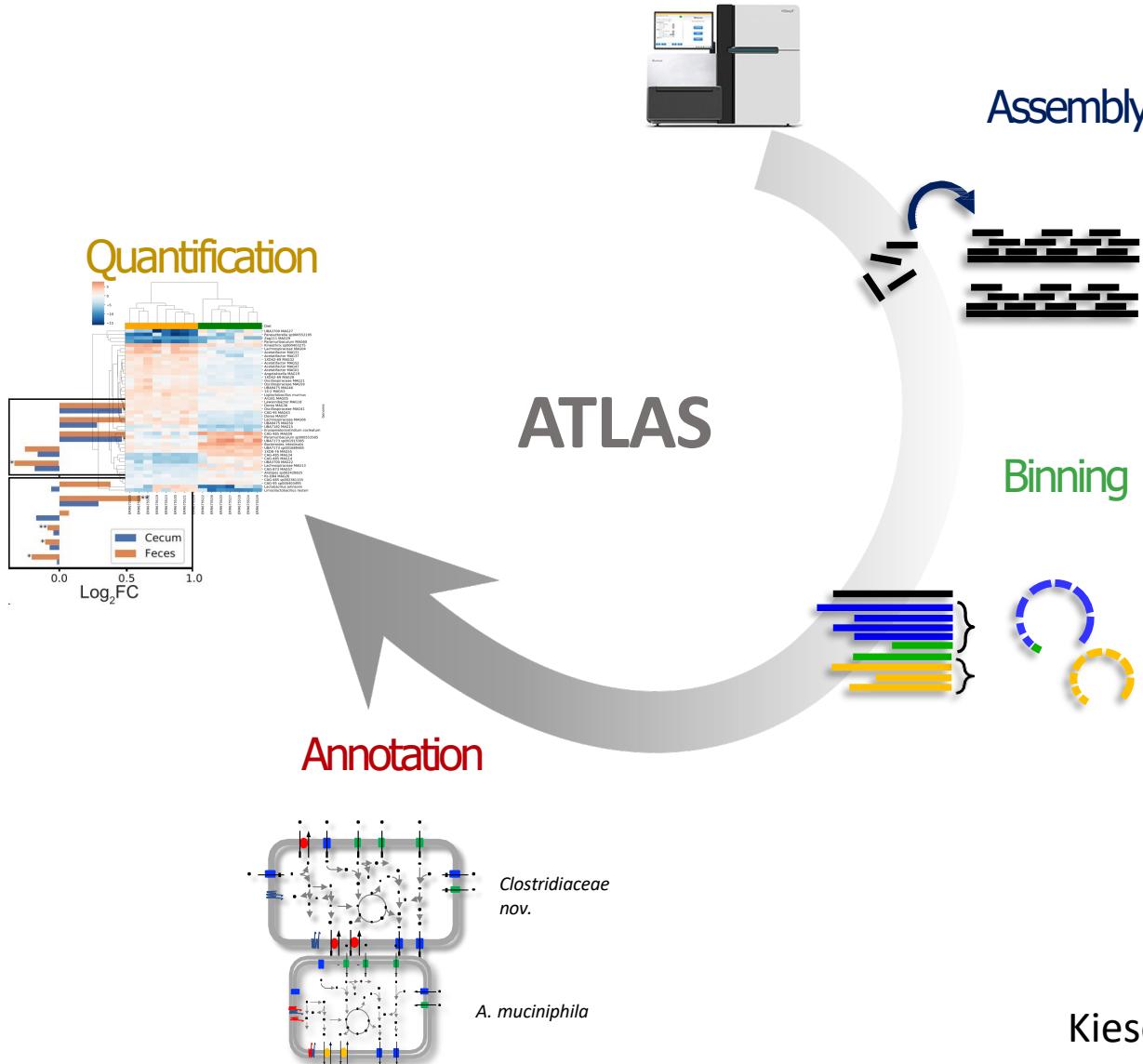
Genes

Genes

Unique Gene catalog

4. Annotation
5. Quantification

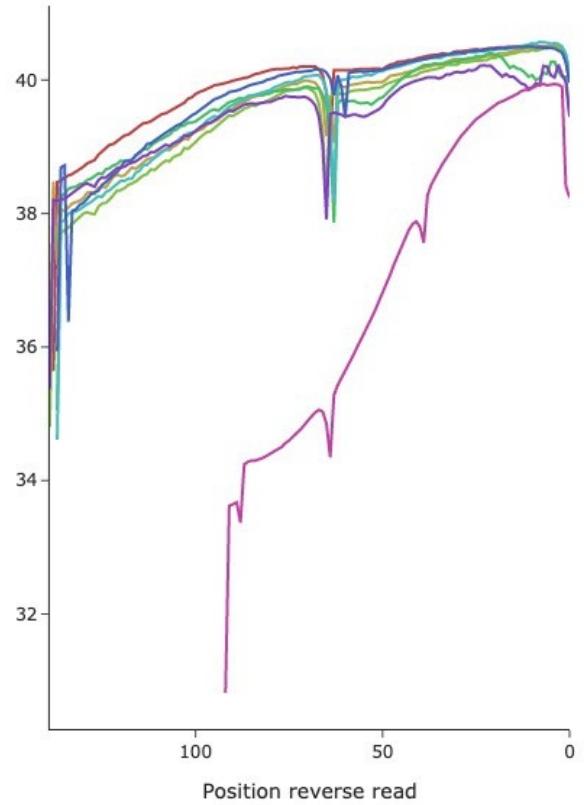
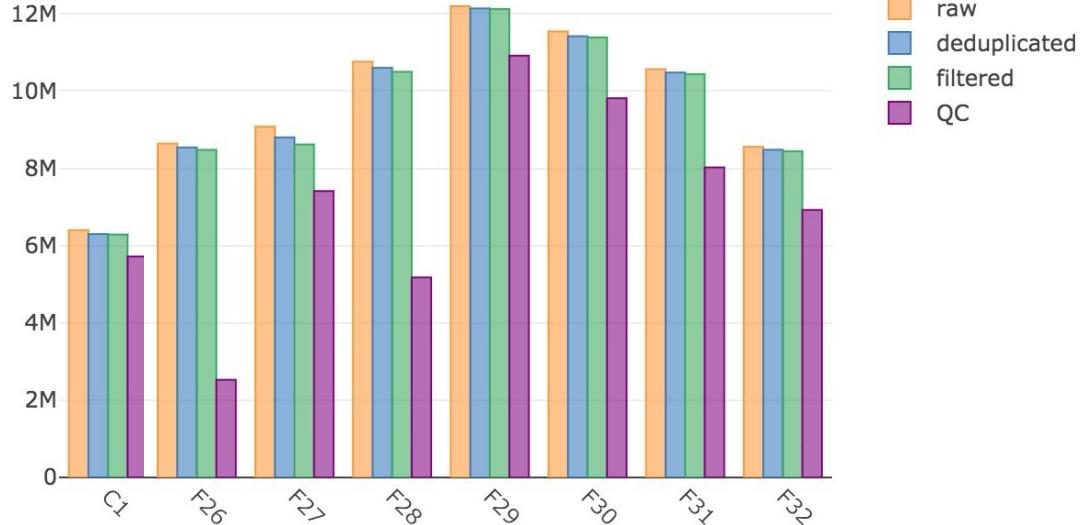
# Metagenome-Atlas in detail





# Quality report

## Total reads per sample



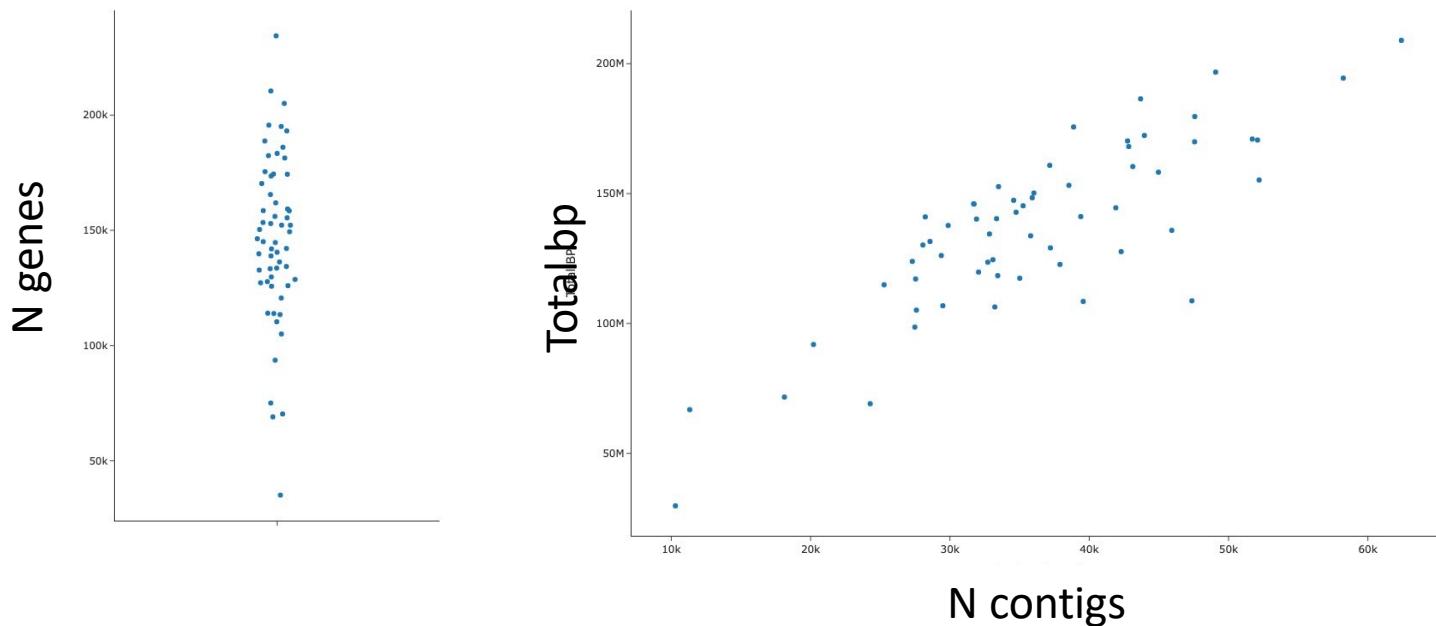
10  
01  
10101 1  
0 0  
101 10  
010 01  
101 10  
010 01  
f g c z  
10  
01 1

## 2. Assembly

- Uses metaSpades or megahit
- Pre-processing
  - Error correction
  - Paired-end merging (pre-assembly)
- Hybrid assembly supported



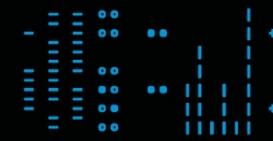
# Assembly report



10  
01  
010  
010101 1  
01 01  
101 10  
10 01  
01 1

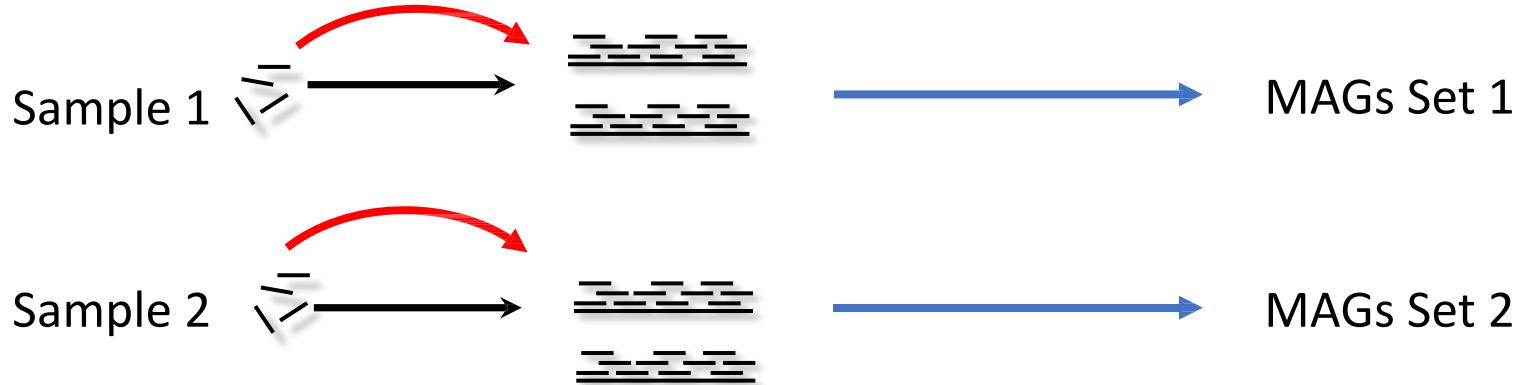
# 3 Binning

- Single-sample / Cross mapping:
  - Metabat2
  - Maxbin2
- Co-Binning
  - Vamb
  - SemiBin

10  
01  
101

# Co-abundance

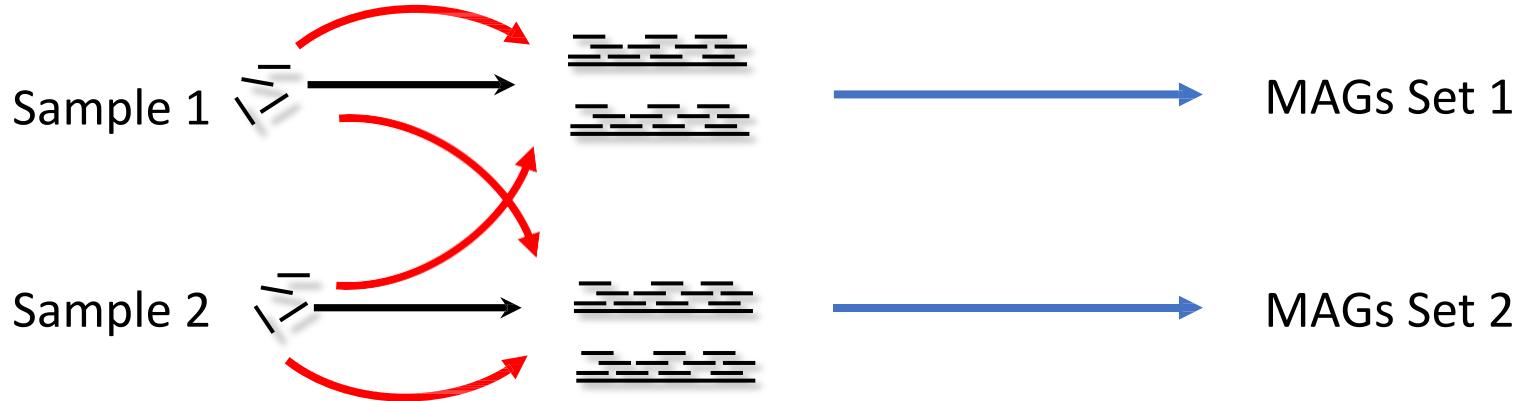
Option 1: Single-sample assembly/Binnig



10  
01  
10101 1  
01  
101 10  
010 01  
101 10  
010 01  
f g c z  
10  
01 1

# Co-abundance

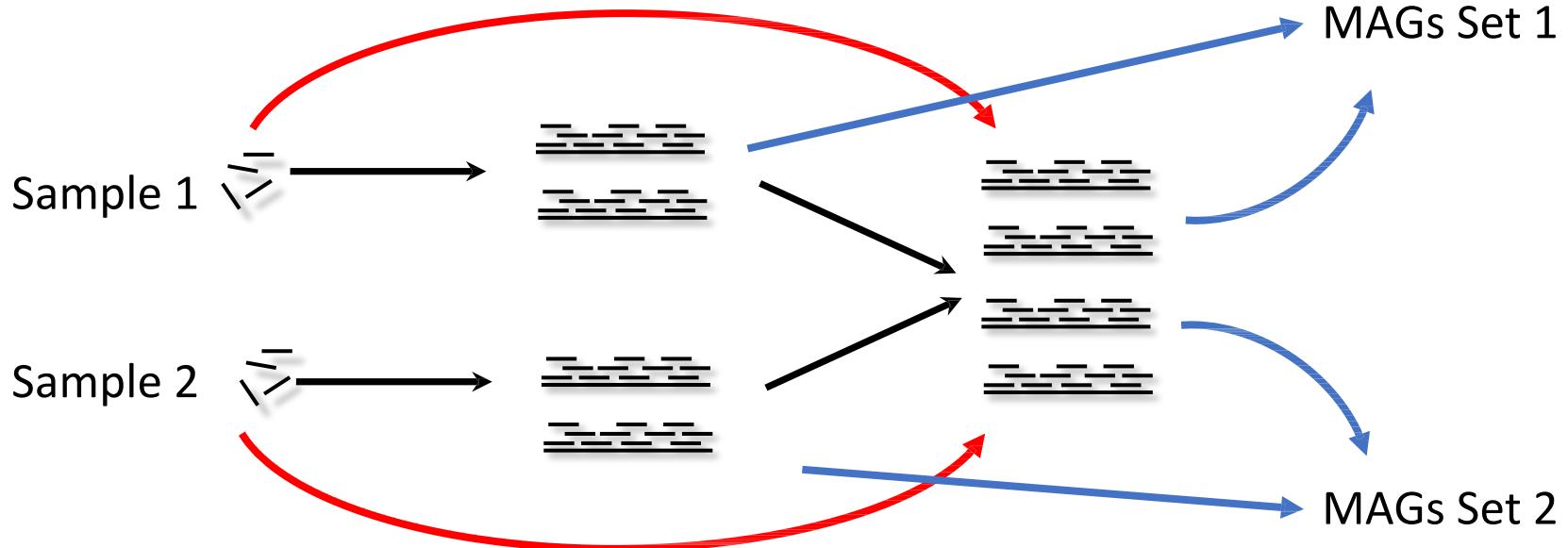
## Option 2: Cross mapping





# Co-abundance

## Option 3:Co-binning

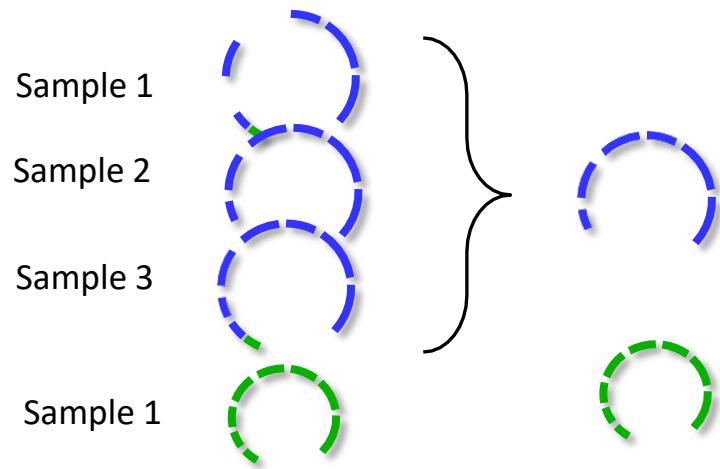


# scale studies on the Human microbiome

	CIBO	EBI	JGI	ATLAS
	Pasolli et al. 2019	Almeida et al. 2019	Nayfach et al. 2019	Kieser et al. 2020
Assembly	metaSpades Megahit			
Binning	Metabat	Metabat	Metabat Maxbin Concoct DASTool	Metabat Maxbin  DASTool VAMB SemiBin
Quality estimation	CheckM			



# De-replication

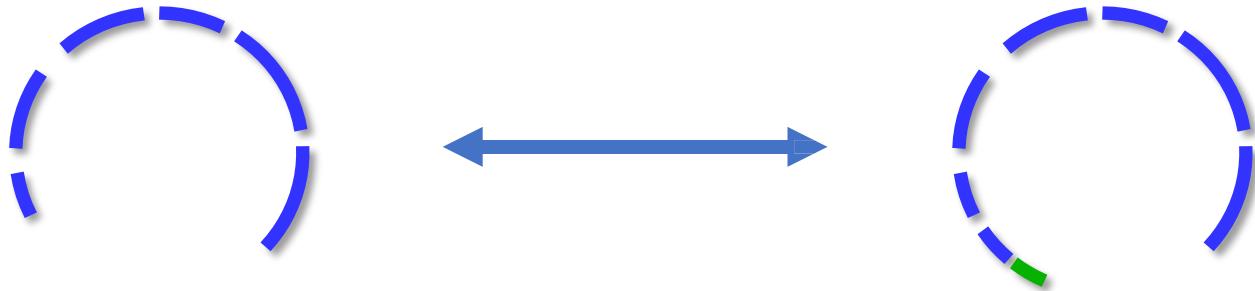


10  
01  
101101 1  
..  
010 0  
0101 10

functional genomics center zurich

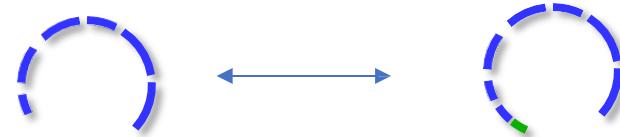
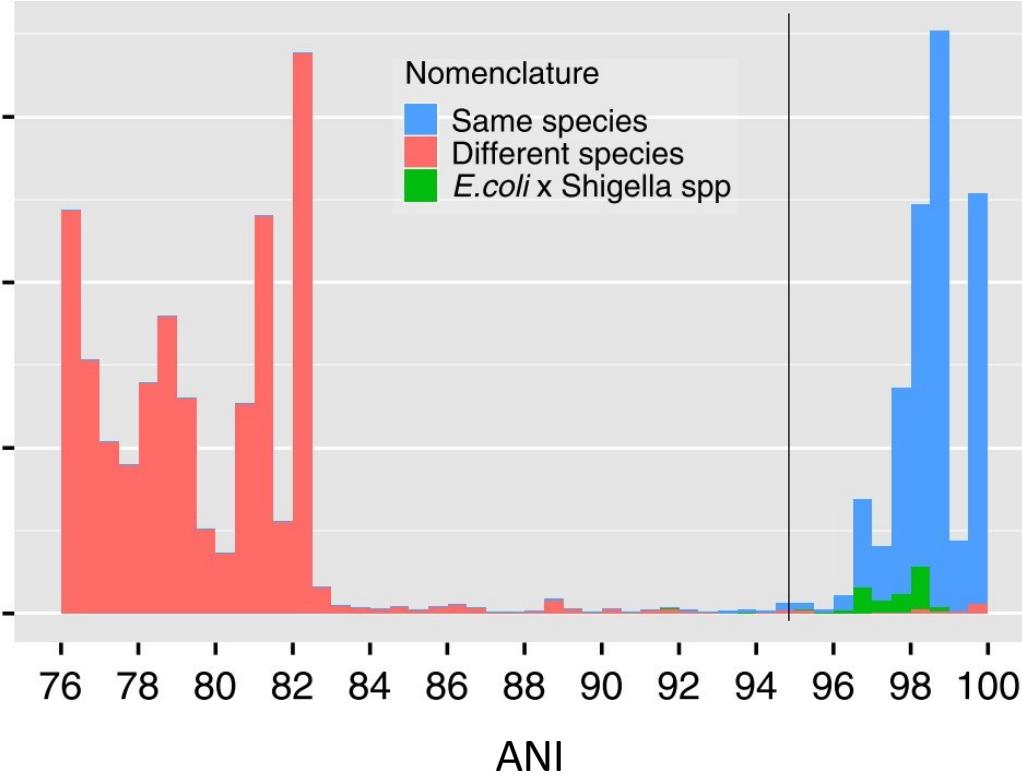
01 1  
01 01  
101 10  
010 01  
010 01  
10 01  
01 1

# Average nucleotide Identity (ANI)





# 95% ANI used as species threshold



Jain et al. 2018

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

# Taxonomic annotation

# Genome Taxonomy database (GTDB)

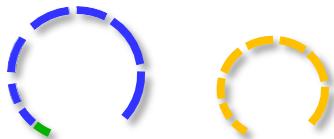
## GTDB-Tk

[pypi v2.3.2](#) [downloads 103k](#) [bioconda v2.3.2](#) [downloads 93k](#) [docker v2.3.2](#) [pulls 4.7k](#)

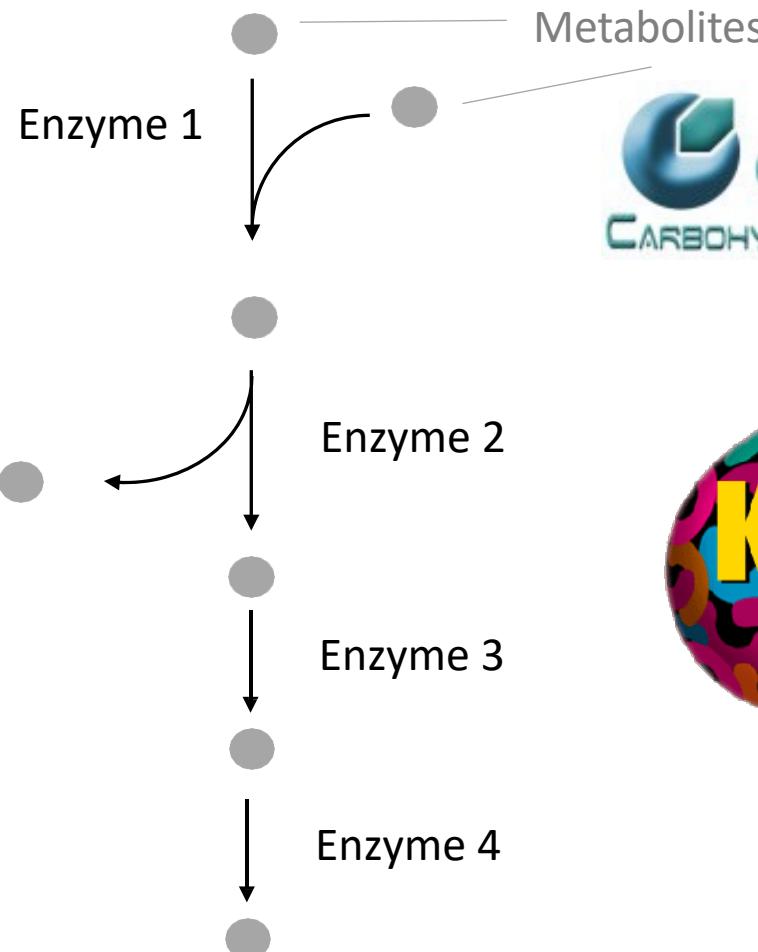
GTDB-Tk is a software toolkit for assigning objective taxonomic classifications to bacterial and archaeal genomes based on the Genome Database Taxonomy ([GTDB](#)). It is designed to work with recent advances that allow hundreds or thousands of metagenome-assembled genomes (MAGs) to be obtained directly from environmental samples. It can also be applied to isolate and single-cell genomes. The GTDB-Tk is open source and released under the [GNU General Public License \(Version 3\)](#).

# Pathway inference

MAG 001 MAG002



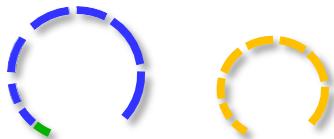
Enzyme 1	Enzyme 1
Enzyme 2	Enzyme 2
	Enzyme 4
X	✓



DRAM

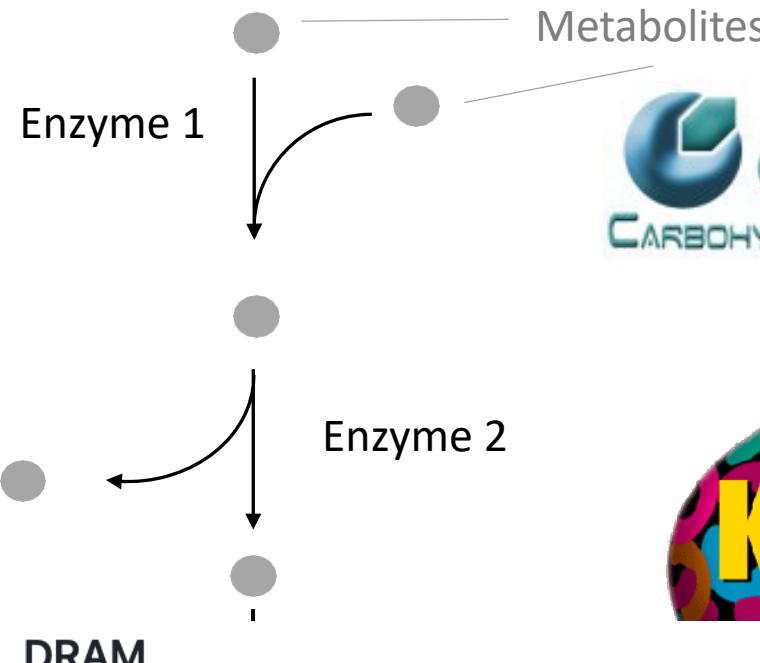
# Pathway inference

MAG 001 MAG002



Enzyme 1	Enzyme 1
Enzyme 2	Enzyme 2
	Enzyme 4

X ✓

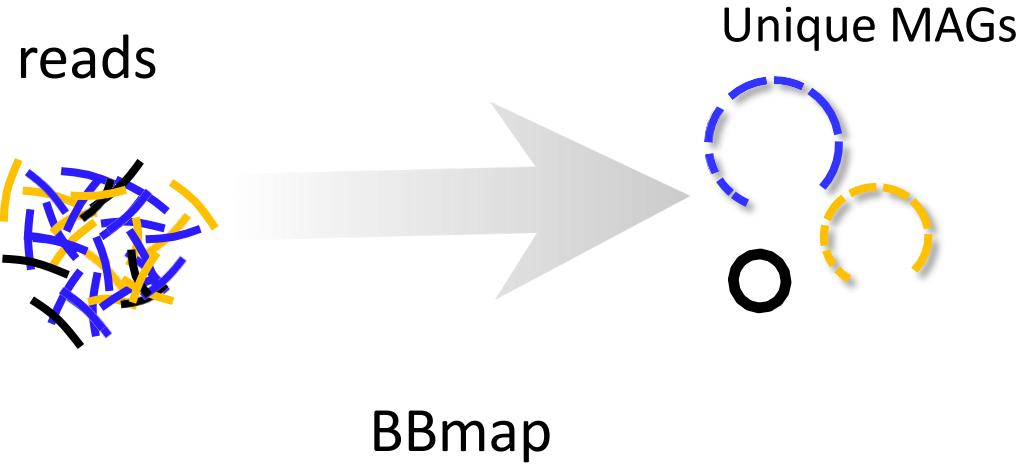


FAILED Bioconda 26k

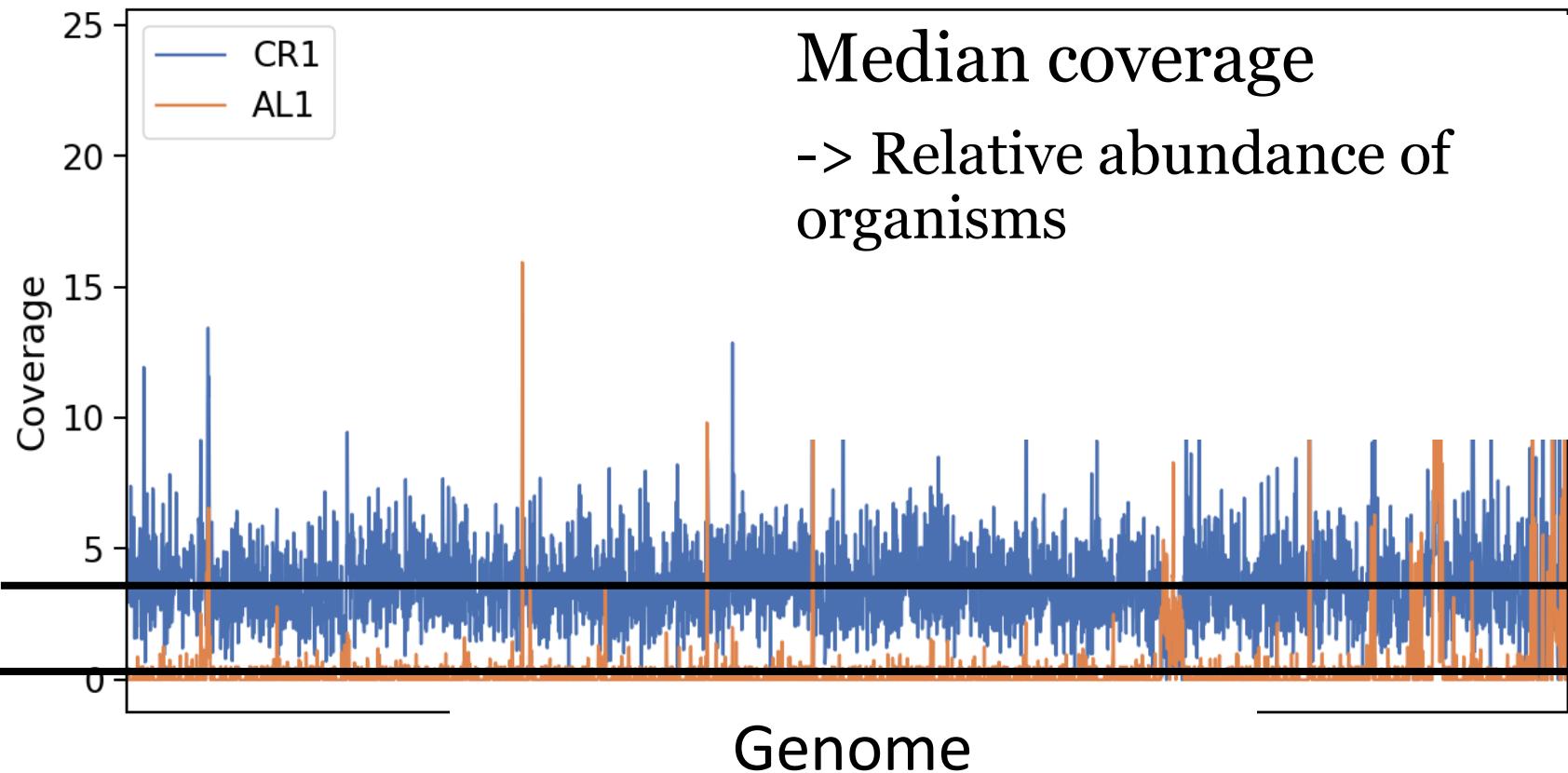
DRAM (Distilled and Refined Annotation of Metabolism) is a tool for annotating metagenomic assembled genomes and [VirSorter](#) identified viral contigs. DRAM annotates MAGs and viral contigs using [KEGG](#) (if provided by the user), [UniRef90](#), [PFAM](#), [dbCAN](#), [RefSeq viral](#), [VOGDB](#) and the [MEROPS](#) peptidase database as well as custom user databases. DRAM is run in two stages. First an annotation step to assign database identifiers to gene, and then a distill step to curate these annotations into useful functional categories. Additionally, viral contigs are further analyzed during to identify potential AMGs. This is done via assigning an auxiliary score and flags representing the confidence that a gene is both metabolic and viral.

10  
01  
101101 1  
010 0  
0101 1001 1  
010 01  
101 10  
10 01  
01 0

# Quantification



# What is the abundance of a genome?



10  
01  
10101 1  
01 01  
101 10  
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)