KAIJU

Contents

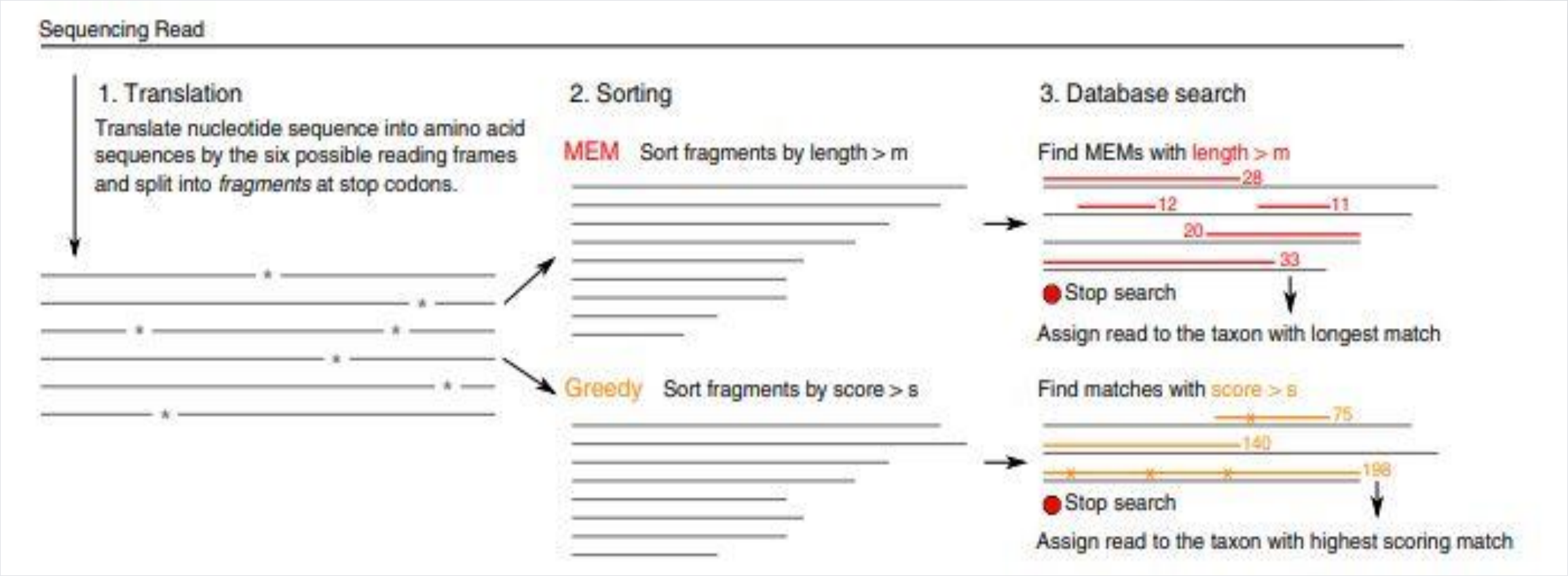
Intro to Kaiju

- Kaiju is a novel metagenome classifier designed for fast and sensitive taxonomic classification of metagenomic sequences.
- Utilizes protein-level sequence comparison using the Burrows-Wheeler Transform for maximum exact matches.
- Addresses the issue of evolutionary divergence, which affects the sensitivity of k-mer-based classification methods.
- Demonstrates improved classification performance, especially for genera underrepresented in reference databases, leading to classification of a larger fraction of metagenomic reads.
- Designed to be efficient, capable of processing millions of reads per minute on standard PC hardware.

Advantages

- Higher sensitivity compared to current k-mer-based classifiers, with similar or improved precision.
- Superior at classifying reads from genera with few or no genomes in reference databases, addressing a common limitation of existing methods.
- Offers a web server and freely available source code for accessibility.
- Requires significantly less memory and computational resources, making it feasible for use on standard computers.
- Demonstrates a substantial increase in the number of classified reads in real metagenomic datasets compared to competitors like Kraken and Clark.

Kaiju Algorithm



Source: https://www.nature.com/articles/ncomms11257

Kaiju Algorithm

- Step 1: Metagenomic Read Processing

  Kaiju starts by translating each metagenomic read into the six possible reading frames, and these amino acid sequences are then split into fragments at stop codons.

- Step 2: Fragment Sorting and Querying

MEM Mode: Fragments are sorted by length and queried against the reference database using backward search in the BWT. Only exact matches (MEMs) longer than a minimum length (m) are considered.

Greedy Mode: Sorts fragments based on BLOSUM62 score, allowing for amino acid substitutions during the backward search to extend matches at their ends.

Cont'd..

- ## Step 3: Database Search

Performs backward search in the BWT to find maximal exact matches (MEMs) between query fragments and the database sequences.

The search algorithm quickly identifies sequences in the database that share the longest subsequence with the query.

- ## Step 4: Taxonomic Classification

Once a match or matches are found, the taxonomic identifier from the corresponding database sequence is retrieved and printed to the output.

If equally long matches are found in multiple taxa, Kaiju determines their lowest common ancestor (LCA) from the taxonomic tree and outputs its taxon identifier.

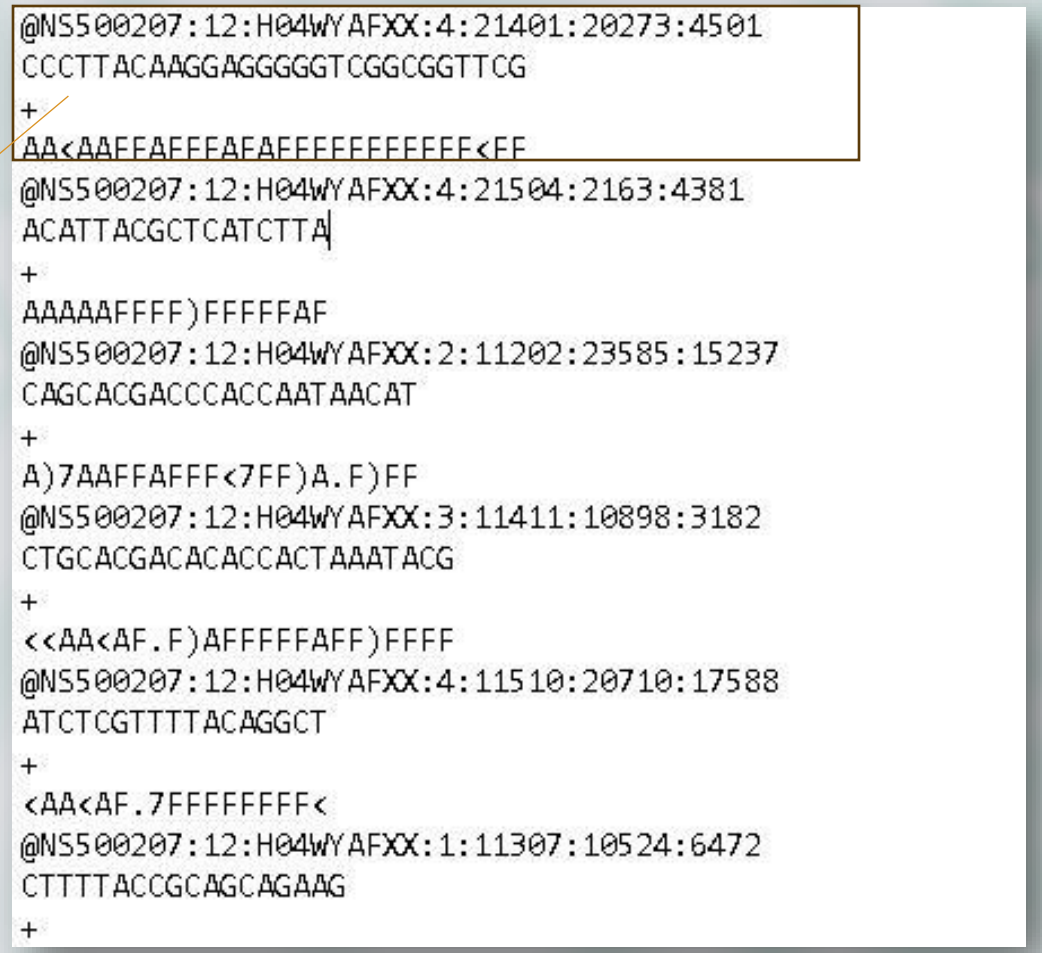
Understanding the Dataset

- Dataset: evol1.sorted.unmapped.R1.fastq

R1: Specifies that the file contains the first read of paired-end sequencing data.

unmapped: Indicates that the reads in this file did not align or map to a reference genome

- First Line: Starts with '@' followed by a sequence identifier and an optional description.
  Here: @NS500207:12:H04WYAFXX:4:21401:20273:4501 is an identifier indicating the instrument, run number, flow cell ID, and coordinates among other details.
- Second Line: Contains the nucleotide sequence (DNA or RNA). Here - CCCTTACAAGGAGGGGGTCGGCGGTTCG.
- Third Line: Begins with a '+' and may be followed by the same sequence identifier (optional) and a description.
- Fourth Line: Represents the quality scores for each nucleotide in the sequence, encoded as ASCII characters (AA<AAFFAFFFAFAFFFFFFFFFF<FF).



Snippet of the Dataset as viewed on Notepad

Total Reads: 18,753
Average Read Length: 24 nt
Minimum Read Length: 15 nt
Maximum Read Length: 150 nt

Cont'd..

- "@NS500207:12:H04WYAFXX:4:21401:20273:4501". This identifier contains specific components that are characteristic of **Illumina sequencing output**:
- Instrument ID (e.g., NS500207): Indicates the specific Illumina sequencer that generated the reads.
- Run Number (e.g., 12): A unique identifier for the sequencing run.
- Flow Cell ID (e.g., H04WYAFXX): Identifies the flow cell used for the sequencing run.
- Lane (e.g., 4): Specifies the lane of the flow cell where the sample was sequenced.
- Tile Number (e.g., 21401): Indicates the tile within the lane where the reads were generated.
- X and Y Coordinates (e.g., 20273:4501): Provide the position of the read within the tile.

Running Kaiju on Linux

Download Kaiju locally or create a Kaiju container

Download a database you want to use

- $ wget <database URL>
- $ tar xzf <database.tar>

Or create a database index locally

- $ singularity run /ifs/groups/eces450650Grp/containers/kraken-kaiju.sif kaiju-makedb -s <database>

Running Kaiju on Linux

Once done, the directory containing your database should have 3 files

- kaiju_db_<database>.fmi
- nodes.dmp
- names.dmp

Run the following command:

- $ singularity run /ifs/groups/eces450650Grp/containers/kraken-kaiju.sif kaiju –t nodes.dmp –f kaiju_db_<database>.fmi –i evol1.sorted.unmapped.R1.fastq

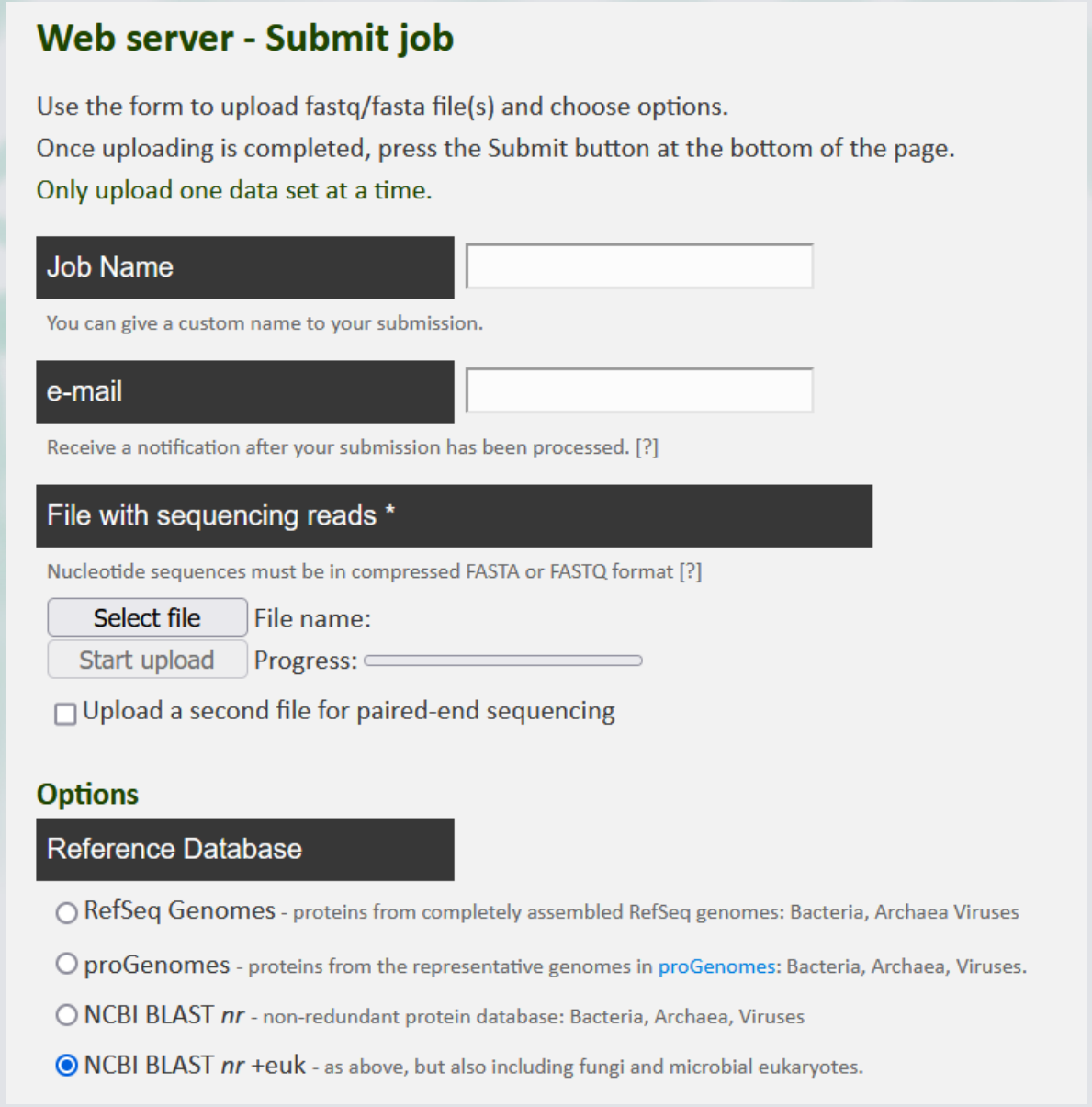
Running Kaiju on Linux

Pros

- Very customizable
- Allegedly much faster than the webserver
- Much larger selection of databases

Cons

- Needs very large storage space and RAM (204 GB for nr+euk)
- Expect there to be many errors



Kaiju Webserver

Kaiju Webserver

Upload a G-zipped FASTA or FASTQ file

Choose a reference database

We used the RefSeq Genomes and NCBI BLAST nr + euk databases

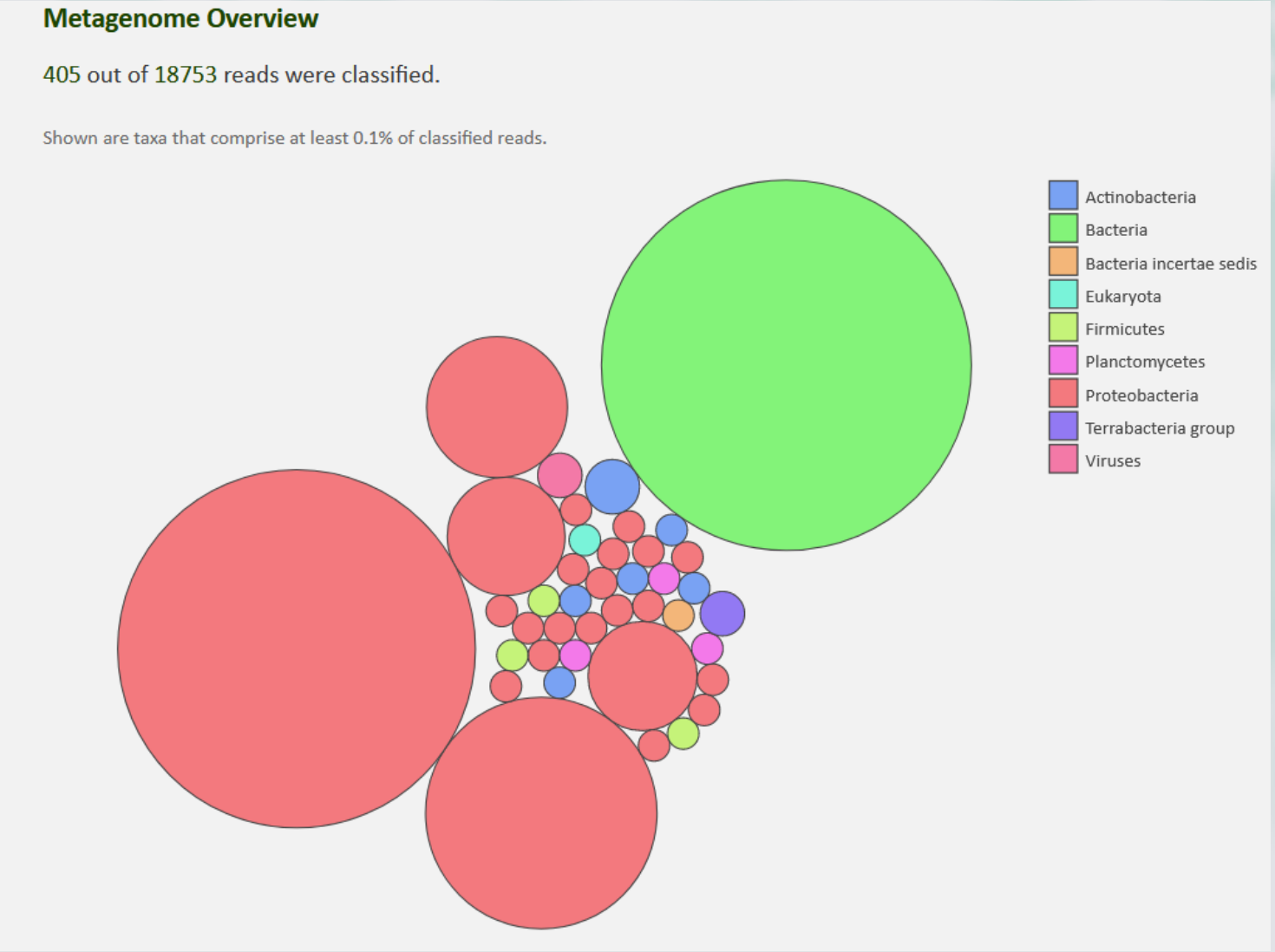
Choose a run mode

Mem or greedy

Adjust other settings

Kaiju Webserver

- Job Parameters
  - Job ID: 16185-2317974664
  - Job Name: tutorial11
  - Reference database: nr +euk
  - Database date: 2021-03
  - SEG low complexity filter: yes
  - Run mode: greedy
  - Minimum match length: 11
  - Minimum match score: 75
  - Allowed mismatches: 5
  - Max. E-value: 0.01
- https://kaiju.binf.ku.dk/output/16185-2317974664/krona.html



Kaiju Webserver

Pros

- User friendly
- Customizable
- Works well will little to no error
- Interesting visual results

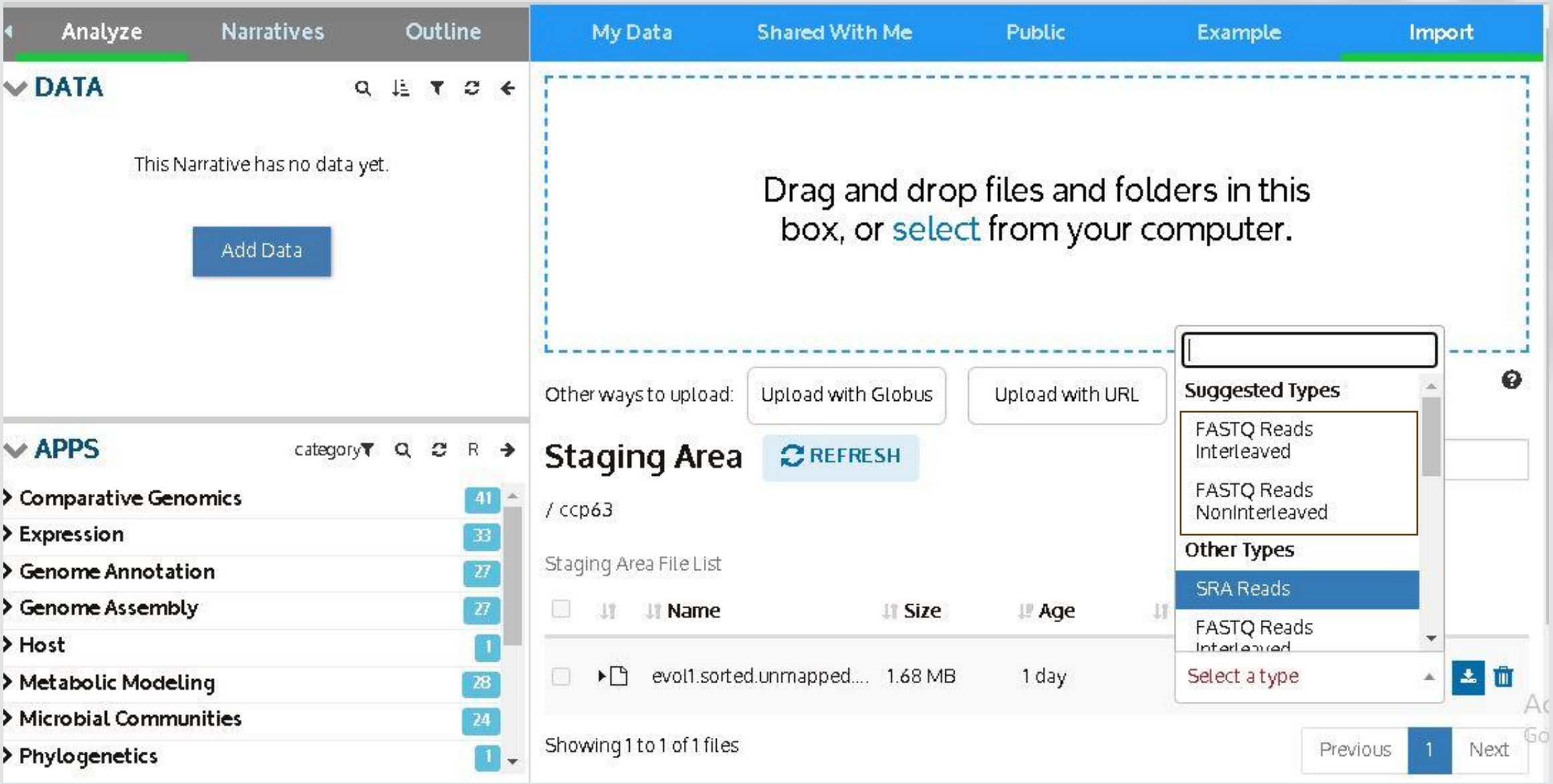
Cons

- Fewer available databases
- Runs very slowly (over 2 weeks per job)

Kaiju Webserver Vs KBase

Features	KBase	Webserver
Reference Database Options	Offer 8	Offers 4
Taxonomic Level	Available to select	Not Available
Low Abundance Filter	Range from 0 to 10	Not Available
Low Abundance Filter	1% to 100%	Not Available
Subsample Replicates	Available to set replicates	Not Available
Subsample Seed	Available to set a seed for subsampling	Not Available
Filter Low-Complexity	Available	SEG Filter Available
Min Match Length	Available with min. 9	Available with min. 7
Allow Imperfect Matches?	Selectable b/w Greedy or MEM	Selectable b/w Greedy or MEM
Greedy Parameters	Allows setting max mismatches, min bitscore, and max E-value	Allows setting minimum match score, allowed mismatches, and max E-value
Sorting Option	Available	Not Available
Time to process	10 mins to 1 hour	About 14 days

Kaiju on KBase – Data Import



Comparisons

- Greedy + NCBI BLAST nr+euk
- Greedy + RefSeq
- MEM + NCBI BLAST nr+euk
- MEM + RefSeq

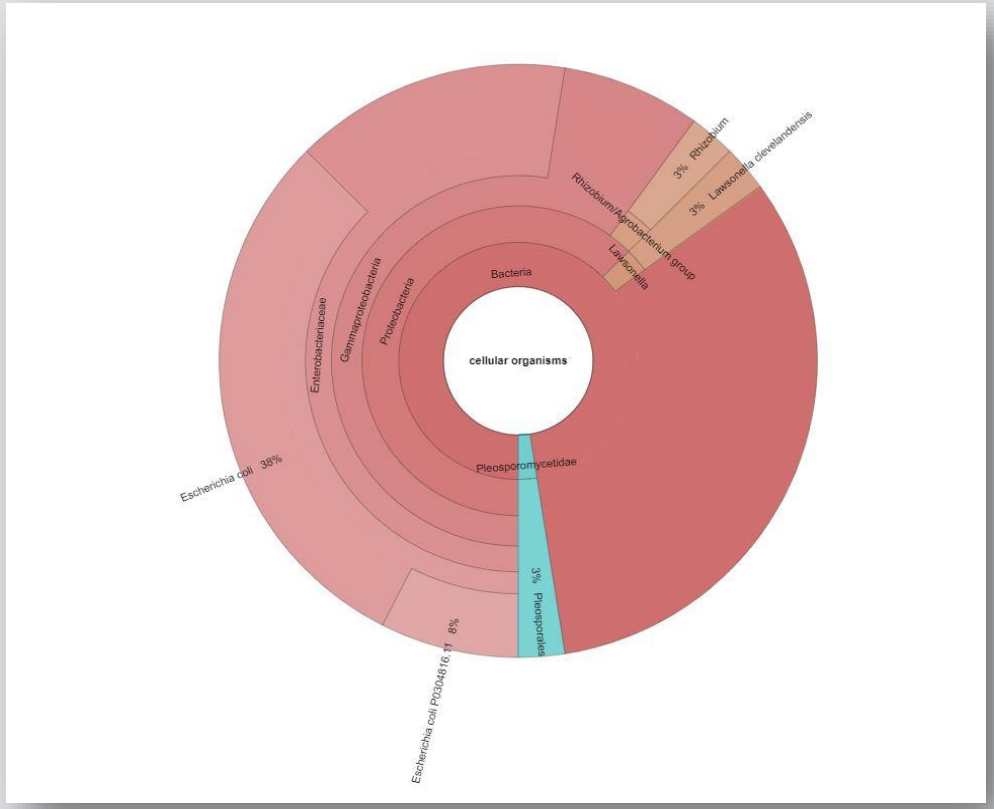
Parameters:

Features	Description
Taxonomic Level	Select one or more taxonomic levels to include in the summary plots.
Reference DB	Select the reference database to search against for classification.
Low Abundance Filter	Select to filter out taxa with low abundances, e.g. only show genera that comprise at least this percent of the total reads (default is 0.5%).
Subsample Percent	Subsample each data set to run faster.
Subsample Replicates	Add replicates to determine the robustness of measurement.
Subsample Seed	Set the seed for random subsampling.
Filter Low-Complexity	Use the SEG algorithm to remove low-complexity regions from input sequences (recommended).
Min Match Length	The shortest alignment match to use for classification, in base pairs (default is 11).
Allow Imperfect Matches?	Imperfect matches can be used for classification with below thresholds (recommended).
Greedy Max Mismatches	Greedy (imperfect) match maximum mismatches (default is 3 - used to be 5).
Greedy Min Bitscore	Greedy (imperfect) match minimum bitscore (default is 65 - used to be 75).
Greedy Max E-value	Greedy (imperfect) match maximum e-value (default is 0.01 - used to be 0.05).
Sort Plots by	Show abundance plots sorted either by alphabetical of taxa or by total abundance (def is total abundance).

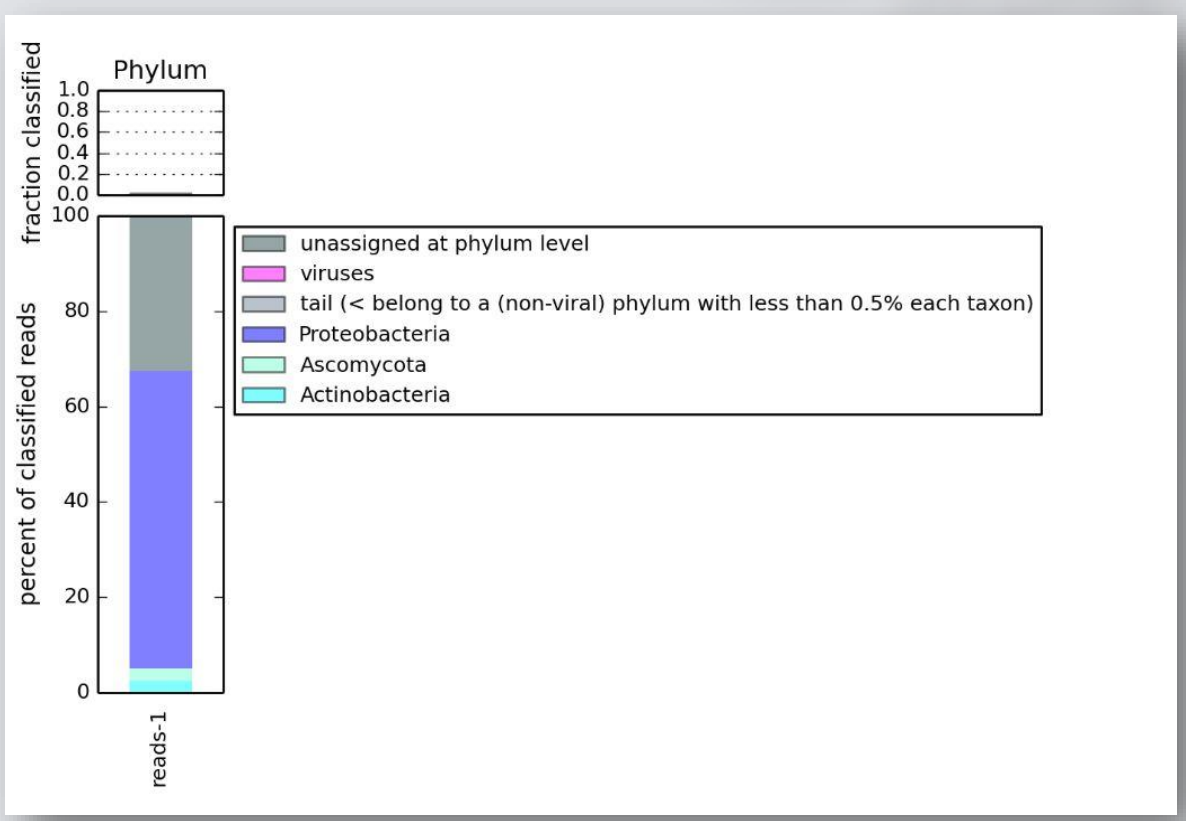
Parameters Used for all tests

- For the MEM mode, a minimum match length of 11 amino acids (a.a.) was chosen.
- In the Greedy modes, a minimum required match score of 65 was utilized.

Greedy Mode & NCBI BLAST nr + euk

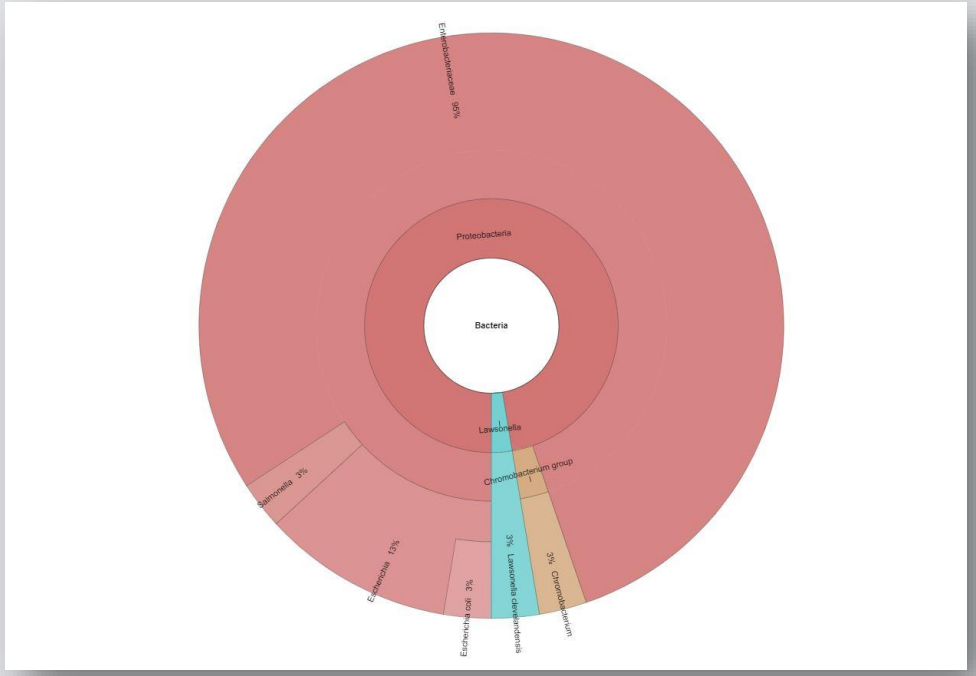


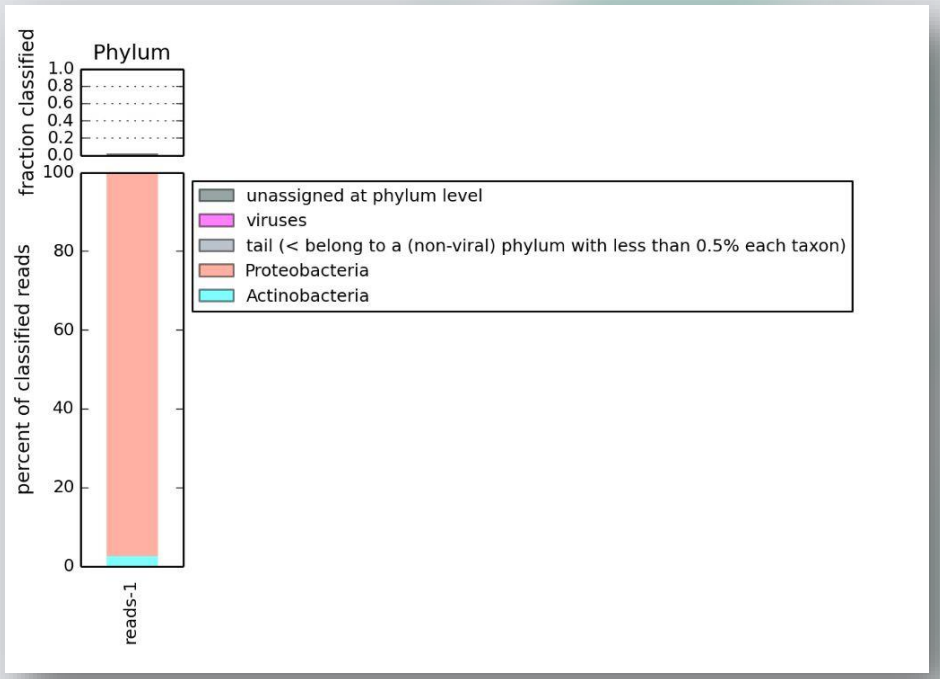
Krona Plot



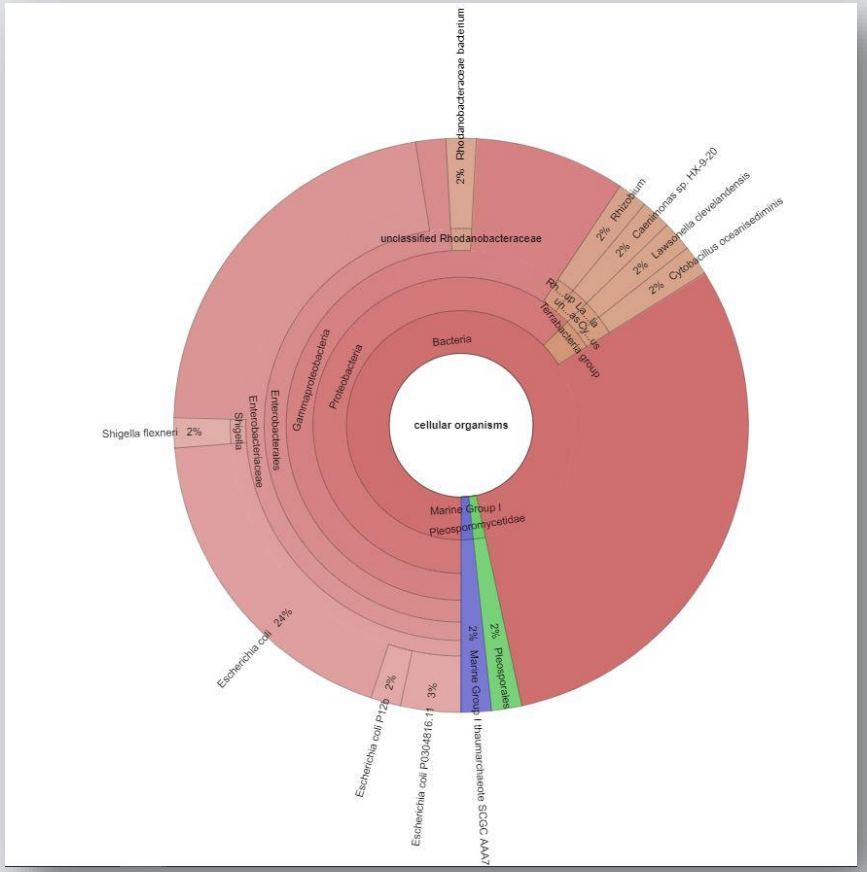
Stacked Bar Plot

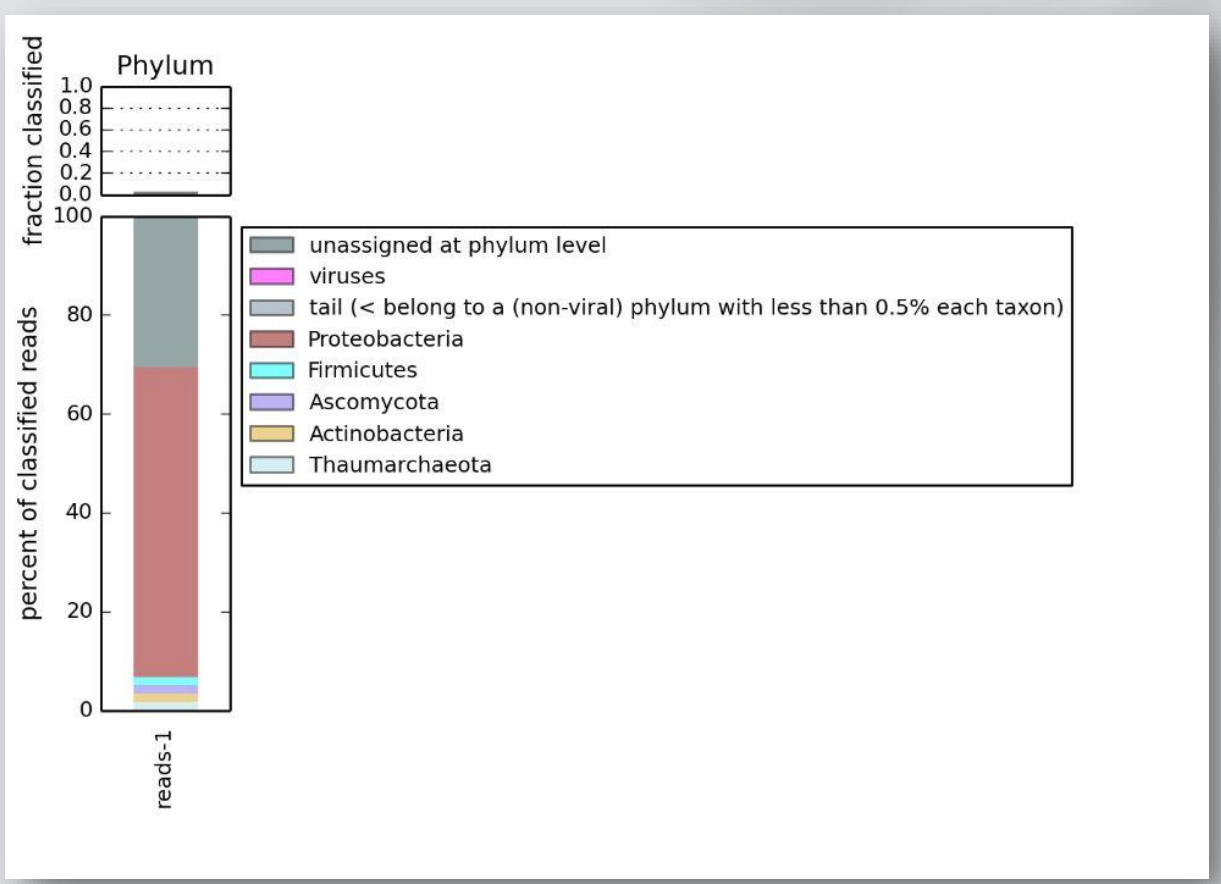
Greedy Mode & RefSeq



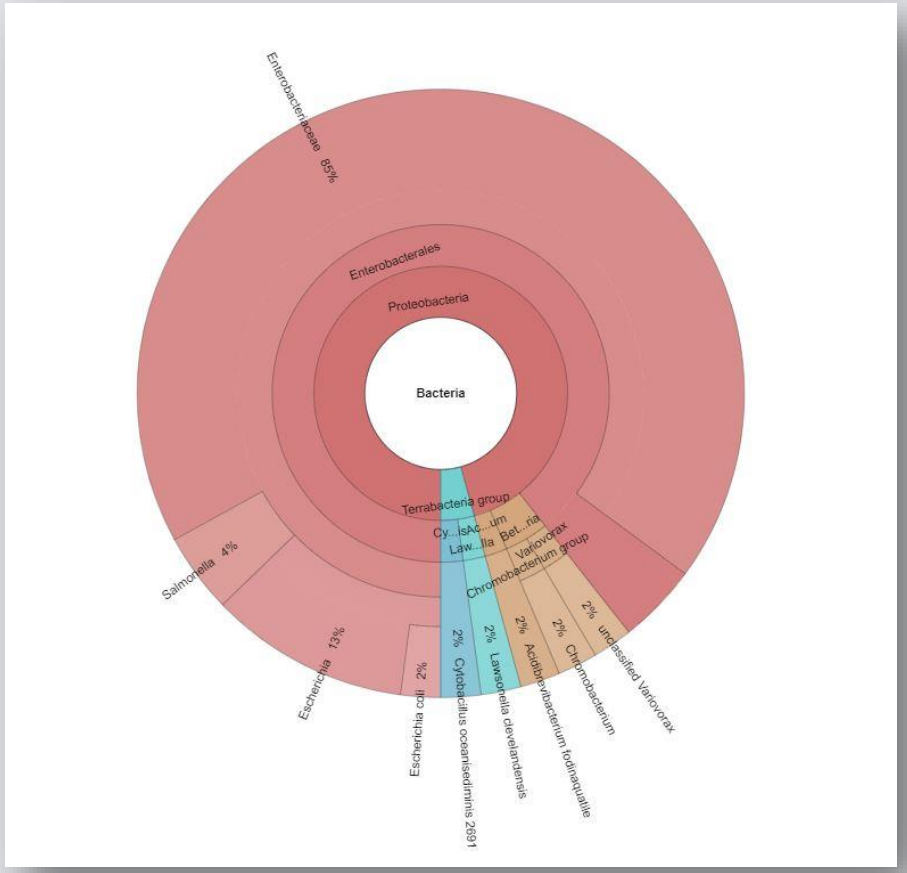


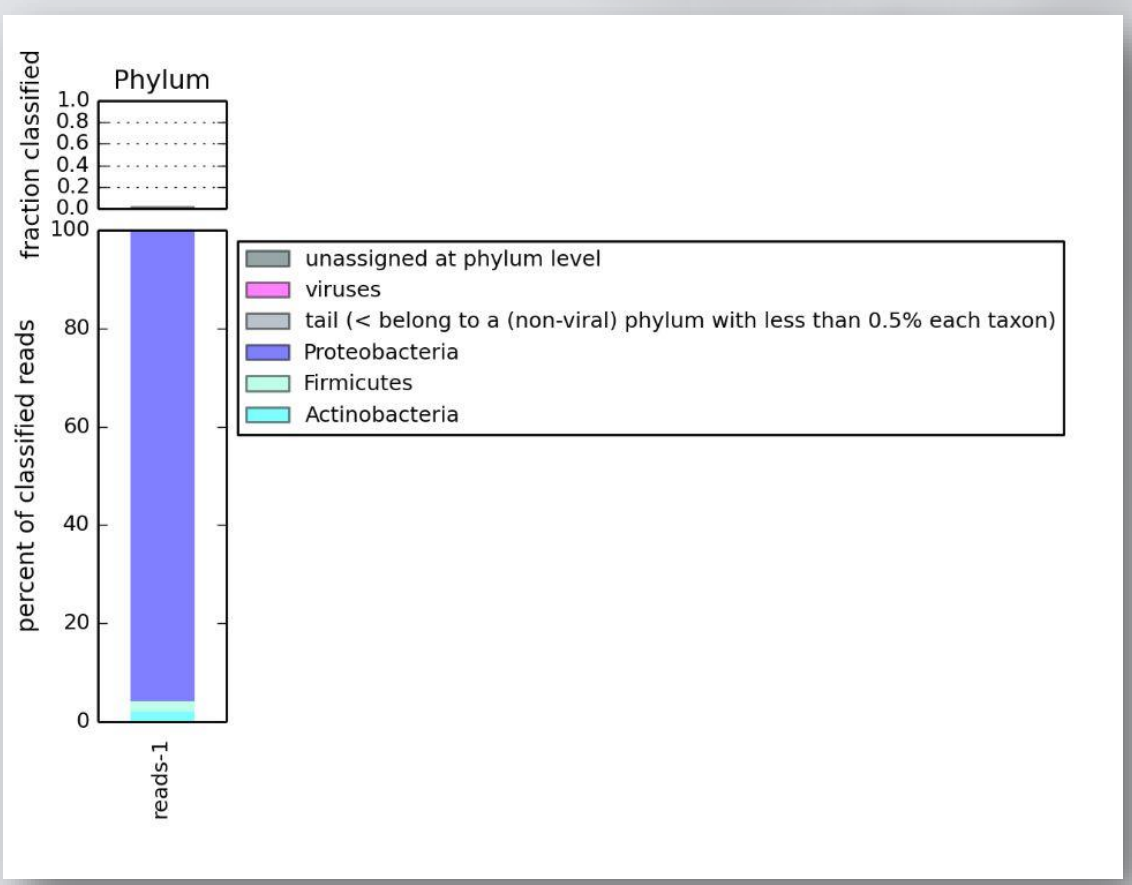
MEM & NCBI BLAST nr + euk





MEM & RefSeq


Enterobacteriaceae 85%
Enterobacterales
Proteobacteria
Bacteria
Terrabacteria group
Cy...lsAc...um
Law...lla
Bet...ria
Variovorax
Chromobacterium group
Salmonella 4%
Escherichia 13%
Escherichia coli 2%
2% Cytobacillus oceanisediminis 2691
2% Lawsonella clevelandensis
2% Acidipropionibacterium jodinamyliticum
2% Chromobacterium
2% unclassified Variovorax


Phylum
fraction classified
1.0
0.8
0.6
0.4
0.2
0.0
100
80
60
40
20
0
percent of classified reads
reads-1
unassigned at phylum level
viruses
tail (< belong to a (non-viral) phylum with less than 0.5% each taxon)
Proteobacteria
Firmicutes
Actinobacteria