MEGAN 7: Metagenome Analyzer

User Manual

Daniel H. Huson

June 26, 2025

Contents

Preface

This manual provides a comprehensive guide to MEGAN (MEtaGenome ANalyzer), a software application for analyzing metagenomic sequencing data. MEGAN enables users to explore the taxonomic and functional content of microbiomes through intuitive visualizations and powerful computational tools.

The manual is intended for:

- Researchers and students in microbiology, bioinformatics, and environmental genomics.
- Users new to metagenomics who wish to gain a practical understanding of how to analyze sequencing data.
- Experienced analysts looking to integrate MEGAN into automated workflows or large-scale projects.

This guide covers everything from installation and basic usage to advanced analyses, scripting, and command-line integration.

About this edition: This edition of the manual corresponds to MEGAN version 7, and reflects updates in features, interface, and supported formats. Older versions of MEGAN may differ in appearance and functionality.

Acknowledgments: MEGAN is developed and maintained by Daniel H. Huson at the University of Tübingen. We gratefully acknowledge the contributions of the broader bioinformatics community, and the developers of tools and databases such as DIAMOND, NCBI taxonomy, SEED, KEGG, and eggNOG.

We hope that this manual helps you make the most of MEGAN's capabilities in your research.

Daniel H. Huson
Tübingen, June 26, 2025

1 Getting Started

This is a brief user manual for MEGAN 7, covering both the free *Community Edition* (CE) and the licensed *Ultimate Edition* (UE). Features only available in the UE are shown in this color.

1.1 Background

In microbiome analysis, samples are subjected to metagenomic sequencing and three main computational questions are:

- Who is out there? What is the taxonomic content of a sample?
- What are they doing, or what can they do? What is the functional content of a sample?
- How do they compare? Do changes in the taxonomic or functional content of samples reflect changes in the microbiome?

The key idea of the DIAMOND+MEGAN approach is to align sequencing reads or assembled contigs against a protein reference database using DIAMOND [Buchfink et al., 2015], and then to analyze the alignments to perform taxonomic and functional binning of sequences, interactively using MEGAN and/or on the commandline using the MEGAN daa-meganizer tool [Huson et al., 2007, 2011, 2016].

Why use protein alignments? DNA alignments can certainly be used to identify known genomes in a sample, in the context of known pathogen detection, or in the analysis of well-studied environments (such as the human gut, for well-studied populations), say. However, for the analysis of unknown organisms from less well studied environmental sources, protein alignment is more suitable due to the higher level of sequence conversation.

The plot (see Figure 1.1) shows that only a small part of the phylogenetic diversity estimated to exist in the environment is represented by full DNA sequences in genomic databases [Wu et al., 2009].

In the DIAMOND+MEGAN approach, DNA sequencing reads, or assembled contigs, are first translated into protein sequences and then aligned to protein reference sequences in a "translated alignment":

- A short sequencing read:

  ```
  >HISEQ:457:C5366ACXX:2:1101:11231:33443
  CTTGCAAGGCAATTTTTGTTGGAGAACTTTGTGAGCCGTGGCATGGTGGTTGACTTCGCCGTACACCAGCCAGACCGGGA
  GGACGGCGGCATACCAAACCC
  ```

- The first of the six-frame translations:

  ```
  LARQFLLENFVSRGMVVDFAVHQPDREDGGIPN
  ```

- Alignment against a reference protein:

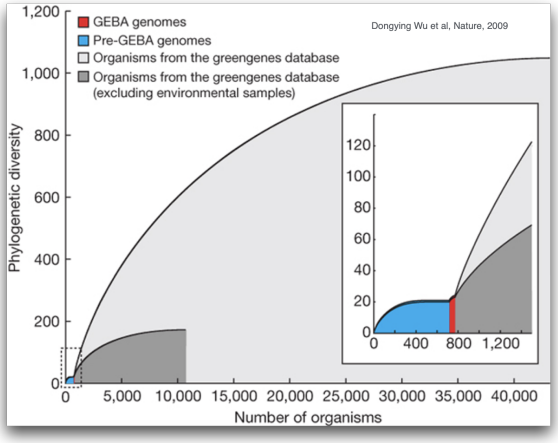

Figure 1.1: This plot shows that only a tiny proportion of the phylogenetic diversity of archaea and bacteria is represented by full genome sequences in public databases, source: [Wu et al., 2009].

```
>RCH48316
 Length = 421

 Score = 72 bits (175), Expect = 9e-11
 Identities = 33/33 (100%), Positives = 33/33 (100%), Gaps = 0/33 (0%)
 Frame = +1

Query:         1  LARQFLLENFVSRGMVVDFAVHQPDREDGGIPN  99
                  LARQFLLENFVSRGMVVDFAVHQPDREDGGIPN
Sbjct:       104  LARQFLLENFVSRGMVVDFAVHQPDREDGGIPN  136
```

Metagenomic sequencing projects can involve hundreds of samples each containing tens of millions of sequences. The NCBI-nr protein reference database contains over 800 million reference sequences. Thus, alignment-based metagenomic analysis is computationally demanding and the first steps are usually performed on a server or cluster.

The number of reference proteins is continuing to increase, see Figure 1.2, and thus alternative, smaller databases such as AnnoTree [Gautam et al., 2021], UniRef100, UniRef90, UniRef50 [Suzek et al., 2014], or NCBI-nr clustered at 90% or 50% identity, may be more suitable in the future.

To reduce computational load and the amount of required disk space, the DIAMOND+MEGAN pipeline is very stream-lined and produces only one output file for each input file (see Figure 1.3).

The two computationally demanding steps, alignment of sequences against a reference database (DIAMOND alignment), and then analysis of the resulting alignments (MEGANization), are usually run on a server, whereas the third step, interactive exploration and analysis of the results, is performed on a personal computer.

1.2 Introduction

MEGAN (MEtaGenome ANalyzer) is an interactive and versatile tool for analyzing metagenomic data. It allows users to explore the taxonomic and functional content of microbiome datasets derived from high-throughput sequencing experiments.

Originally developed to support the interpretation of BLAST comparisons of environmental sequence data, MEGAN has evolved into a comprehensive platform for metagenomic analysis.

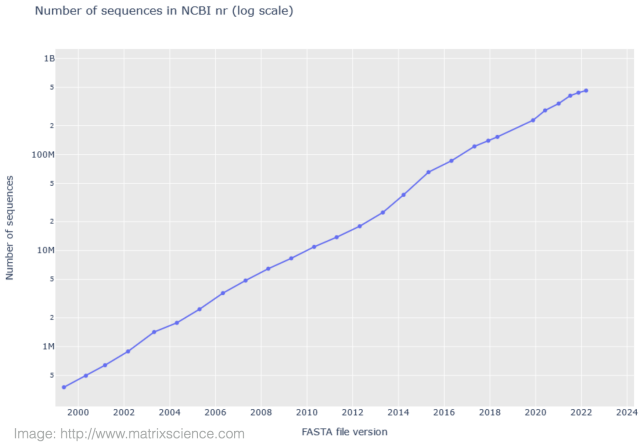

Figure 1.2: The number of non-redundant protein sequences represented in the NCBI-nr database is growing at an exponential rate, source: `http://www.matrixscience.com`.

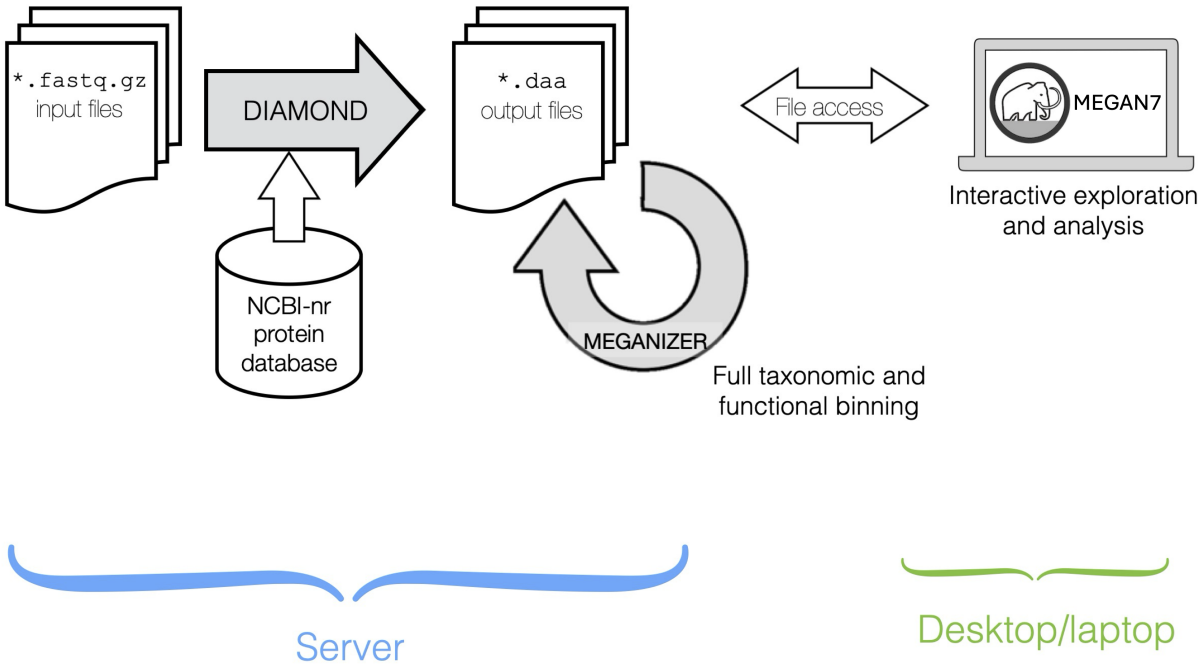

Figure 1.3: DIAMOND+MEGAN analysis is performed in two steps. In the first step, on a server, all sequences are aligned against a protein reference database using DIAMOND, and then the taxonomic and functional content is computed in a step called "meganization". In the second step, on a laptop or desktop, meganized datasets are interactively explored, analyzed and compared using MEGAN. .

It supports modern alignment formats such as DIAMOND DAA files and can handle very large datasets efficiently.

MEGAN provides a graphical user interface that enables users to:

- Classify reads taxonomically using the NCBI taxonomy or other taxonomies.
- Perform functional analysis using a variety of classification systems, including SEED and eggNOG, and KEGG (Ultimate Edition only).
- Visualize and explore data interactively using taxonomic trees, bar charts, heatmaps, and comparative plots.
- Conduct statistical and comparative analyses of multiple samples.
- Export publication-ready images and tables for downstream use.

MEGAN is designed to be accessible to users with a wide range of experience in bioinformatics. It offers powerful defaults for beginners, as well as extensive customization options for advanced users.

This manual is intended to guide users through the installation, usage, and interpretation of results in MEGAN. Whether you are exploring your first microbiome dataset or performing complex multi-sample comparisons, this manual will help you make the most of MEGAN's capabilities.

For further information, updates, and support, please visit the MEGAN project website at:

https://software-ab.cs.uni-tuebingen.de/download/megan7

1.3 Installation

1.3.1 System Requirements

MEGAN is a Java-based application and runs on Windows, macOS, and Linux. To run MEGAN, the following system requirements should be met:

- Operating System: Windows 10 or later, macOS 10.15 or later, or a recent Linux distribution.
- Memory: At least 8 GB of RAM (16 GB or more recommended for large datasets).
- Disk Space: Minimum 1 GB free disk space; additional space required for data files.
- Display: 1024×768 resolution or higher recommended.

1.3.2 Downloading MEGAN

You can download the latest version of MEGAN from the official website:

https://software-ab.cs.uni-tuebingen.de/download/megan7

The download page provides platform-specific installers for Windows, macOS, and Linux.

1.3.3 Installation Instructions

Windows

1. Download the `MEGAN_Community_windows-x64_7_x_x.exe` file.

2. Double-click the installer and follow the on-screen instructions.
3. Once installation is complete, you can launch MEGAN via the Start menu or desktop shortcut.

macOS

1. Download the `MEGAN_Community_macos_7_x_x.dmg` file.
2. Double-click on the DMG and follow the on-screen instructions.
3. On first launch, you may need to confirm that you want to run the application via `System Preferences > Security & Privacy`.

Linux

1. Download the `MEGAN_Community_unix_7_x_x.sh` file.
2. Location the file in a terminal and run it by typing `./MEGAN_Community_unix_7_x_x.sh`.
3. Navigate to the extracted directory and run `./MEGAN` from the terminal.

1.3.4 Installing the Ulimate Edition

[The following is *Ultimate Edition Only.*]

Windows

1. Download the `MEGAN_Ultimate_windows-x64_7_x_x.exe` file.
2. Double-click the installer and follow the on-screen instructions.
3. Once installation is complete, you can launch MEGAN via the Start menu or desktop shortcut.

macOS

1. Download the `MEGAN_Ultimate_macos_7_x_x.dmg` file.
2. Double-click on the DMG and follow the on-screen instructions.
3. On first launch, you may need to confirm that you want to run the application via `System Preferences > Security & Privacy`.

Linux

1. Download the `MEGAN_Ultimate_unix_7_x_x.sh` file.
2. Location the file in a terminal and run it by typing `./MEGAN_Ultimate_unix_7_x_x.sh`.
3. Navigate to the extracted directory and run `./MEGAN` from the terminal.

1.3.5 Troubleshooting Installation

- **macOS warning:** On newer versions of macOS, MEGAN may need explicit permission to run. Use `System Preferences > Security & Privacy` to allow the app to launch.
- **Permission issues on Linux:** Make sure the MEGAN binary is marked as executable using `chmod +x MEGAN`.

- **Startup problems:** Run MEGAN from the command line to view any error messages.
- [The following is *Ultimate Edition Only.*] **Licensing problems:** The Ultimate Edition requires a license key, please see the Appendix for more details.

For further support, consult the MEGAN user forum or contact the development team via the project website.

1.4 Mapping Databases

To perform taxonomic and functional classification of reads, MEGAN relies on a **mapping database** that connects reference sequences (such as those in the NCBI-nr protein database) to classification systems like the NCBI taxonomy, SEED, eggNOG and KEGG (UE only).

This mapping information is essential when:

- Meganizing a DIAMOND alignment file (`.daa`) using the `daa-meganizer` tool or using MEGAN.
- Converting a BLAST output file into a MEGAN-compatible `.rma6` file.

MEGAN requires a single, unified `.mdb` file in SQLite format that contains these mappings. Without it, MEGAN cannot assign reads to taxa or functional categories.

The mapping database can be downloaded from the official MEGAN7 website:

https://software-ab.cs.uni-tuebingen.de/download/megan7

The database file is typically named something like `megan-nr-r1.mdb`, with the number indicating the release number.

After downloading, place the file in your working directory or specify its location using the `-mdb` option when running command-line tools such as:

```
daa-meganizer -i input.daa -mdb megan-nr-r1.mdb
```

In the GUI, the path to the mapping database can be set via the import dialog or in the program preferences.

It is important to use a mapping database that corresponds to the version of the reference database used for alignment (e.g., the same release of NCBI-nr), to ensure consistent and accurate classification. There are different mapping files for different reference databases, such as NR, NR90, NR50, UniRef100, UniRef90 and UniRef50.

For example, while `megan-nr-r1.mdb` is to be used with the NCBI-nr database, the mapping file `megan-nr50-r1.mdb` should only be used with the `nr50.gz` database.

[The following is *Ultimate Edition Only.*] If (and only if) you are using the Ultimate Edition of MEGAN, then please use mapping files that have `-ue` in there name as only these contain mapping information for KEGG.

1.5 Quick Start Guide

This section walks you through a basic MEGAN workflow using sample data. By the end, you'll know how to import data, explore the taxonomy, and perform simple analyses.

1.5.1 Obtaining Example Data

MEGAN provides example datasets that you can use to familiarize yourself with the interface. You can download these from:

https://software-ab.cs.uni-tuebingen.de/download/megan7/example-data/

Download and unzip the example archive to a convenient location on your system.

1.5.2 Launching MEGAN

1. Start MEGAN by double-clicking the application icon or executing `./MEGAN` from a terminal.
2. The MEGAN main window will appear, displaying the toolbar, menu, project panel, and data visualization areas.

1.5.3 Importing Alignment Files

1. Select `File > Import from BLAST/DIAMOND file` or use the toolbar icon.
2. Browse to a file such as `example.daa` (produced by DIAMOND).
3. Specify how reads should be assigned to taxa or functional categories.
4. Select the appropriate mapping db file.
5. Click `OK` to begin import. A progress bar will indicate when loading is complete.

1.5.4 Exploring the Taxonomy

Once data is loaded:

- The taxonomy tree appears in the main window.
- Click on a taxon node to view the number of assigned reads and some metadata.
- Use right-click to open context menus for additional actions such as inspecting or exporting.

1.5.5 Viewing Functional Annotations

1. Switch to a functional classification view using the corresponding toolbar item.
2. MEGAN supports SEED, eggNOG and KEGG (Ultimate Edition), and other optional classifications.
3. Explore the tree or chart view to examine functionally annotated reads.

1.5.6 Basic Analysis

- Use the `Chart` menu items to create bar charts, heatmaps, or other charts.
- Use the `Compare...` menu item to compute a comparison document containing and comparing multiple samples.
- Save figures with `File > Export Image`.

You are now ready to begin exploring your own metagenomic datasets with MEGAN!

2 Taxonomic binning

MEGAN performs taxonomic binning of all input sequences (reads or contigs) based on their alignments to a reference database. Reads are assigned to nodes of the NCBI taxonomy [Schoch et al., 2020] in the main viewer. In addition, MEGAN also assigns sequences to archaeal and bacterial nodes in the GTDB taxonomy [Parks et al., 2020], in a separate GTDB viewer.

For short reads (sequences usually overlapping one a single gene), by default, the program uses the *naïve LCA* algorithm to assign reads to taxonomic bins [Huson et al., 2007]. For long reads or contigs (sequences usually overlapping multiple genes), MEGAN uses the *interval-union LCA* algorithm to assign sequences to taxonomic bins [Huson et al., 2018].

2.1 The NCBI Taxonomy

The NCBI Taxonomy database provides unique names and IDs for approximately 2.3 million taxa [Schoch et al., 2020]. This comprehensive resource is used to catalog and classify all forms of life, including formal names and informal names used outside the standard codes of nomenclature.

There are approximately 1.4 million prokaryotes, including bacteria and archaea. The number of animals is about 200,000, covering a wide range of species from simple invertebrates to complex vertebrates. There are around 300,000 plants, spanning various categories of flora including flowering plants, ferns, and mosses. There are roughly 100,000 entries for viruses, reflecting the vast diversity of viral species cataloged in the database. These figures are constantly updated.

The entries are hierarchically grouped into clades at the levels of: Superkingdom, Kingdom, Phylum, Class, Order, Family, Genus, and Species (and some unofficial clades in between). At startup, MEGAN automatically loads a copy of the complete NCBI and then displays the taxonomy as a rooted tree. The taxonomy is stored in an NCBI tree file and an NCBI mapping file, which are supplied with the program.

Sequence assignment to the NCBI taxonomy items are shown in the main viewer (see Figure 2.1A).

2.2 The GTDB Taxonomy

The GTDB taxonomy (version 214.1) contains more than 395,000 bacterial and than 7,000 archaeal entries, in a taxonomy containing more than 500,000 items in total Parks et al. [2020]. The entries are hierarchically grouped into clades at the levels of: Domain, Phylum, Class, Order, Family, Genus, and Species.

Sequence assignment to the GTDB taxonomy items are shown in the GTDB viewer (see Figure 2.1B).

A B

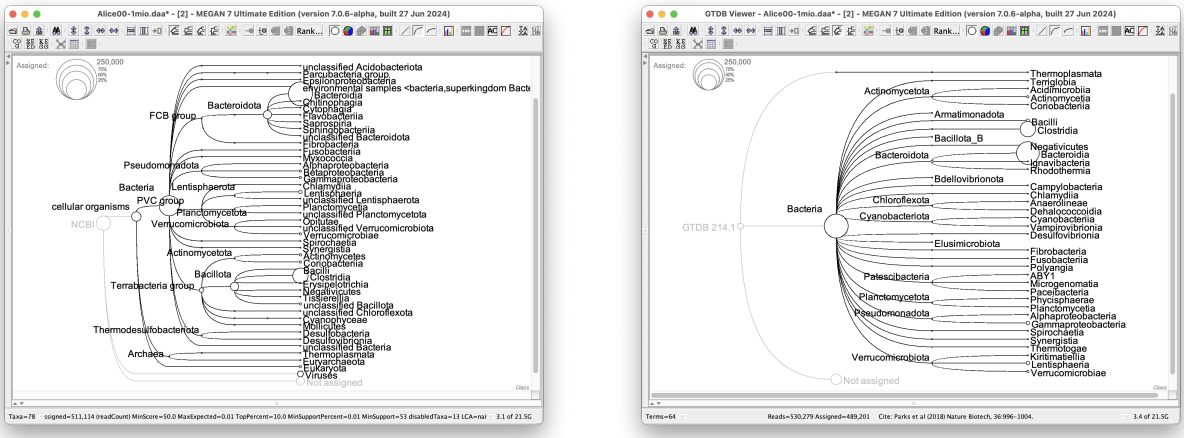

Figure 2.1: A: The NCBI taxonomy viewer summarizes the assignment of reads to different nodes in the NCBI taxonomy. The size of nodes represents the number of reads, or the number of aligned bases, assigned to each node, for short reads or long reads, respectively. Here, the taxonomy is shown collapsed at the rank of Class. B: The GTDB viewer shows a similar view, based on the GTDB taxonomy.

2.3 The NCBI-nr database

The NCBI-nr ("non-redundant") protein sequence database is available from the NCBI website. It contains entries from GenPept, Swissprot, PIR, PDF, PDB and RefSeq. It is non-redundant in the sense that identical sequences are merged into a single entry. The size of this database is growing exponentially, the database contained less than 2 million entries and in 2024 the number is approaching 1 billion entries. Due to this increase, NCBI plans to discontinue providing this database in the form a of a single download in the future.

2.3.1 NR90 and NR50

Due to its rapidly increasing size, the NCBI-nr database is becoming too unwieldy for metagenomic analysis. To address this, we provide two clustered versions of the database, *nr90*, clustered at 90% identity, and *nr50*, clustered at 50% identity [Buchfink et al., 2023]. We also provide the corresponding mapping files required by MEGAN to analyze alignments to these databases.

2.3.2 Uniref100, Uniref90 and Uniref50

We also provide support for aligning against the Uniref100, Uniref90 and Uniref50 databases [Suzek et al., 2014].

2.4 Assigning Reads to Taxa

One main problem addressed by MEGAN is to perform "taxonomic binning" by assigning the reads or contigs from a metagenomics sequencing experiment to appropriate taxa in the NCBI taxonomy.

The program implements the following naive approach to this problem:

1. Compare a given set of DNA reads to a database of known sequences, such as NCBI-nr

Benson et al. [2005], using a sequence comparison tool such as DIAMOND Buchfink et al. [2015].

2. Process this data to determine all hits of taxa by reads.
3. For each read r, let H be the set of all taxa that r hits.
4. Find the lowest node v in the NCBI taxonomy that encompasses the set of hit taxa H and assign the read r to the taxon represented by v.

This is called the *naïve LCA* algorithm (LCA = "lowest common ancestor") . In this approach, every read is assigned to some taxon. If the read aligns very specifically only to a single taxon, then it is assigned to that taxon. The less specifically a read aligns to taxa, the higher up in the taxonomy it is placed. Reads that hit ubiquitously may even be assigned to the root node of the NCBI taxonomy.

If a read has significant matches to two different taxa a and b, where a is an ancestor of b in the NCBI taxonomy, then the match to the ancestor a is discarded and only the more specific match to b is used.

The program provides a threshold for the bit score of hits ("min score"). Any hit that falls below the threshold is discarded. Secondly, a threshold can be set to discard any hit whose score falls below a given percentage of the best hit ("top percent"). Finally, a third threshold is used to report only taxa that are hit by a minimal number of reads or minimal percent of all assigned reads ("min support"). By default, the program requires at least 0.1% of all assigned reads to hit a taxon, before that taxon is deemed present. All reads that are initially assigned to a taxon that is not deemed present are pushed up the taxonomy until a node is reached that has enough reads. This is set using the `Min Support Percent` or `Min Support` item.

This algorithm is also used to assign sequences to the GTDB taxonomy.

2.4.1 Weighted LCA Algorithm

The *weighted LCA algorithm* operates as follows: In a first round of analysis, each reference sequence is given a weight. This is the number of reads that align to the given reference and that have the property that all the significant alignments for the read are to the same species as the reference sequence (but can also be to a strain or sub-species below the species node). In a second round of analysis, each read is placed on the node that is above 75% (default value) of the total weight of all references for which the read has a significant alignment.

The weighted LCA algorithm will assign reads more specifically than the naive LCA algorithm. Because it performs two rounds of read and match analysis, it takes twice as long as the naive algorithm. Also, because it must maintain a table of all references seen, it uses much more memory.

This algorithm is also used to assign sequences to the GTDB taxonomy.

2.4.2 Interval-Union LCA Algorithm for Long Reads

The naïve LCA and weighted LCA algorithms are based on the assumption that the input reads are short enough to usually overlap with a single gene. When using long-read sequencing or when analyzing assembled contigs, then this assumption is not fulfilled and a more sophisticated "long-read" LCA algorithm is required for taxonomic binning.

The *interval-union LCA algorithm* is such a "long-read LCA algorithm".

As described in [Huson et al., 2018], this algorithm processes each read or contig r in turn, in two steps. First, the read is partitioned into a set of intervals $v_1, \ldots, v_m$ that have the property

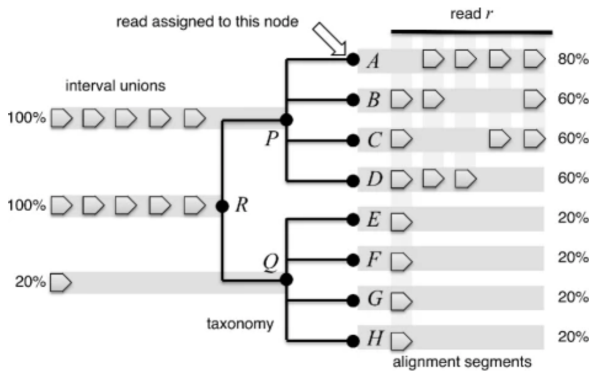

Figure 2.2: To illustrate the interval-union LCA algorithm, here we show eight hypothetical species $A, B, \ldots, H$ separated into two genera, P and Q, belonging to the same family R. Alignments from the read r to proteins associated with the species are indicated by arrows on the right and cover between 80% (for A) and 20% (for H) of the aligned read. Using arrows, on the left we depict the sets of intervals computed for nodes P, Q, R as the union of the sets of intervals of the children of each node. Nodes R and P each cover 100% of the aligned read. The read r is placed on A as it is the lowest taxonomic node with $\leq 80\%$ coverage. Note that, if A only covered 60% of the aligned read, then the read would be assigned to the higher taxon P (and this would remain the case even if one of the taxa below Q had 60% coverage). Source: [Huson et al., 2018].

that every alignment associated with r starts and ends at the beginning or end of some interval, respectively. In other words, a new interval starts wherever some alignment begins or ends. We say that an alignment a_i is significant on an interval v_j , if its bit score lies within 10% (by default) of the best bit score seen for any alignment that covers v_j. This threshold is referred to as the `topPercent` parameter.

In the second step, for each taxon t that is associated with any of the alignments, let $I(t)$ denote the union of all intervals for which there exists some significant alignment a i associated with taxon t. In a post-order traversal, for each higher-rank taxonomic node s we compute $I(s)$ as the union of the intervals covered by the children of s. In result, every node of the taxonomy is labeled by a set of intervals. Note that, during the computation of the union of interval sets, we merge any overlapping intervals into a single interval. The read r is then placed on the taxon s that has the property that its set of intervals $I(s)$ covers 50% (by default) of the total aligned or covered portion of the read, while none of its children does. This threshold is referred to as the `percentToCover` parameter . Note that it is possible that there are multiple nodes that have this property, in which case the read is assigned to the LCA of all such nodes.

This algorithm is also used to assign sequences to the GTDB taxonomy.

3 Functional binning

When applied to short reads, MEGAN7 performs taxonomic binning by assigning reads to functional classes such as orthogonal groups (eggNOG), functional roles and, in the case of the Ultimate Edition, KEGG orthology groups (KEGG). The bins can be inspected using the inspector window.

When applied to long reads or contigs, MEGAN7 determines the presence of genes belonging to different functional along the whole sequence. These annotations can be inspected using the long-read inspector window.

MEGAN7 performs functional binning using eggNOGv6 [Powell et al., 2012] and SEED as represented in PATRIC [Overbeek et al., 2013, Gillespie et al., 2011].

[The following is *Ultimate Edition Only.*] In addition, MEGAN7 Ultimate Edition also performs functional binning using KEGG [Kanehisa et al., 2018].

3.1 eggNOG (v6)

The eggNOG (evolutionary genealogy of genes: Non-supervised Orthologous Groups) database, version 6 [Powell et al., 2012], provides a comprehensive classification of genes into orthologous groups, enabling the prediction of gene functions across a wide range of species. It uses a robust phylogenomic framework to cluster genes based on evolutionary relationships, facilitating the identification of shared ancestry and functional characteristics. Version 6 of eggNOG expands on previous iterations by incorporating more genomes and improving algorithms for more accurate orthologous group predictions. This classification system is pivotal for comparative genomics, functional annotation of novel genes, and understanding the evolutionary dynamics of gene families.

During meganization of a DIAMOND DAA file that represents the alignment of metagenomic reads or contigs against the NCBI-nr or similar database, the alignments of sequences to reference proteins that have a eggNOG (v6) annotation are analyzed and reads are assigned to eggNOG orthogonal groups.

The MEGAN representation of eggNOG contains around 23,000 orthologonal groups that form the leaves of a hierachical classification that contains 34 additional nodes. The result of such a classification is shown in the Figure 3.1.

3.2 SEED

The SEED classification system is a curated resource that organizes genes into functional subsystems based on their roles in various biological processes. Unlike automated approaches, SEED relies on expert curation to group genes into hierarchically structured subsystems, such as metabolic pathways, regulatory networks, and cellular processes. This framework allows for

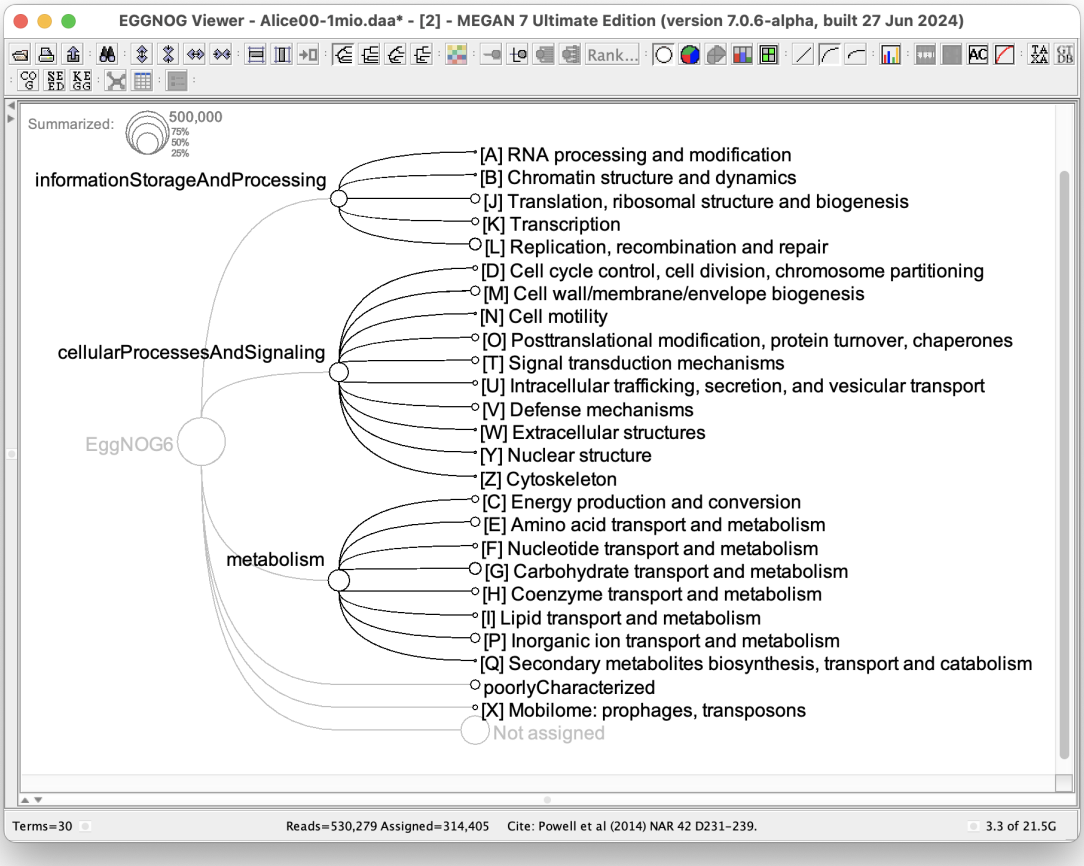

Figure 3.1: Here we show an example of an eggNOG functional binning, collapsed at the second level of the hierarchy.

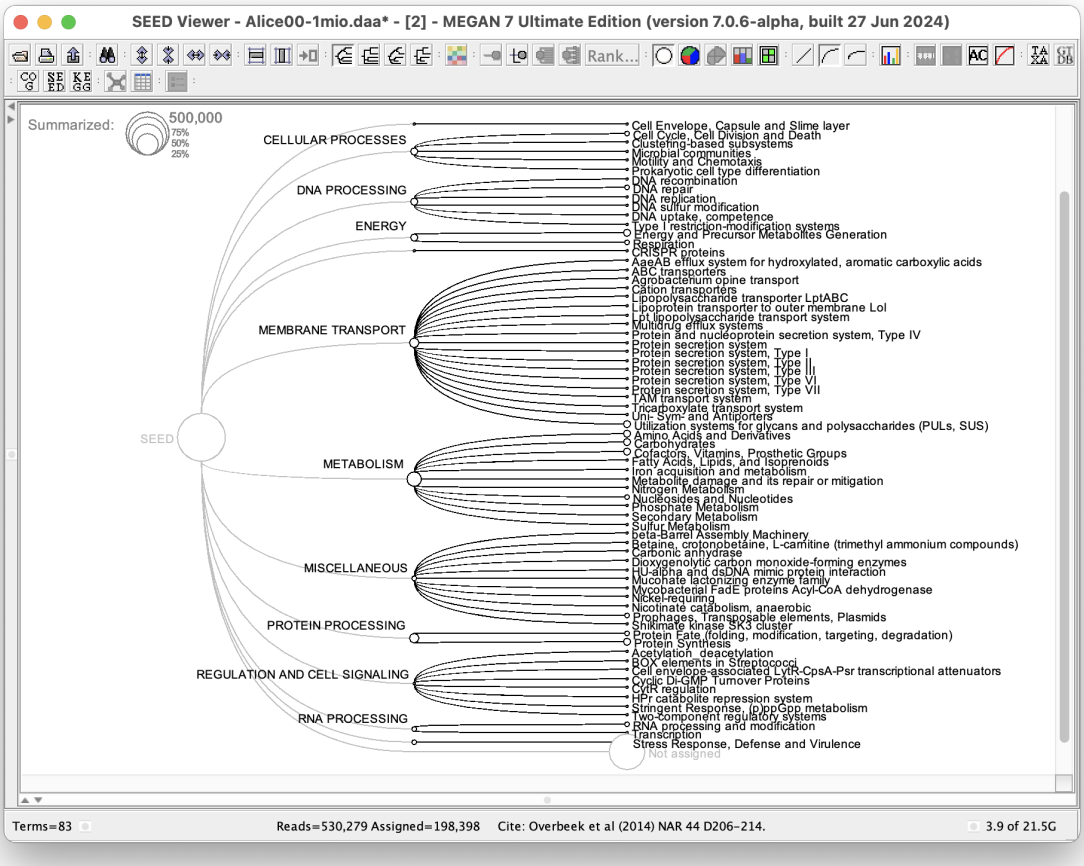

Figure 3.2: Here we show an example of an SEED functional binning, collapsed at the second level of the hierarchy.

high-quality functional annotations and insights into the interconnectedness of different biological functions. SEED's emphasis on context-specific functionality aids researchers in understanding the roles of genes within the broader landscape of cellular activities, making it a valuable tool for genome annotation, metabolic modeling, and systems biology research. We use SEED as integrated into the Pathosystems Resource Integration Center (PATRIC).

During meganization of a DIAMOND DAA file that represents the alignment of metagenomic reads or contigs against the NCBI-nr or similar database, the alignments of sequences to reference proteins that have a SEED annotation are analyzed and reads are assigned to SEED functional roles.

The MEGAN representation of SEED contains around 820 functional roles that form the leaves of a hierachical classification that contains 997 nodes in total. The result of such a classification is shown in the Figure 3.2.

[The following is *Ultimate Edition Only.*]

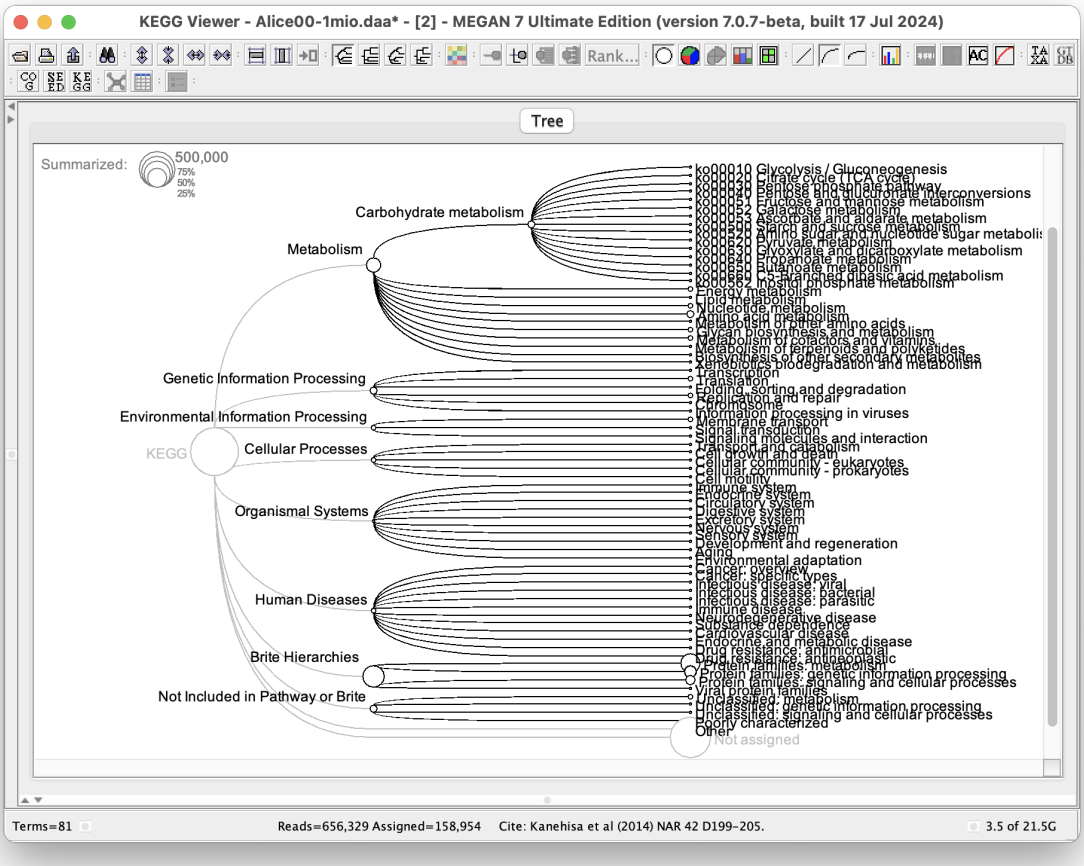

Figure 3.3: Here we show an example of an KEGG functional binning, collapsed at the second level of the hierarchy, and the third level below Carbohydrate metabolism, using MEGAN UE.

3.3 KEGG (Ultimate Edition)

The KEGG (Kyoto Encyclopedia of Genes and Genomes) classification of metagenomic pathways involves categorizing functional annotations of genes found within metagenomic data into various biological pathways. These pathways are grouped into broad categories such as metabolism, genetic information processing, environmental information processing, cellular processes, organismal systems, human diseases, and drug development. Each category is further subdivided into more specific pathways that describe detailed biochemical processes, cellular mechanisms, and organismal functions. This classification aids in understanding the complex interactions within microbial communities, their functional capabilities, and their potential impacts on their environment and host organisms [Kanehisa et al., 2018].

KEGG is only included in the Ultimate Edition of MEGAN. The MEGAN representation of KEGG contains 622 nodes internal nodes and around 28,000 KEGG orthologous groups. The result of such a classification is shown in the Figure 3.3.

The KEGG viewer also provides visualizations of how reads map to different members in metabolic pathways, such as the citrate cycle, see Fig. 3.4

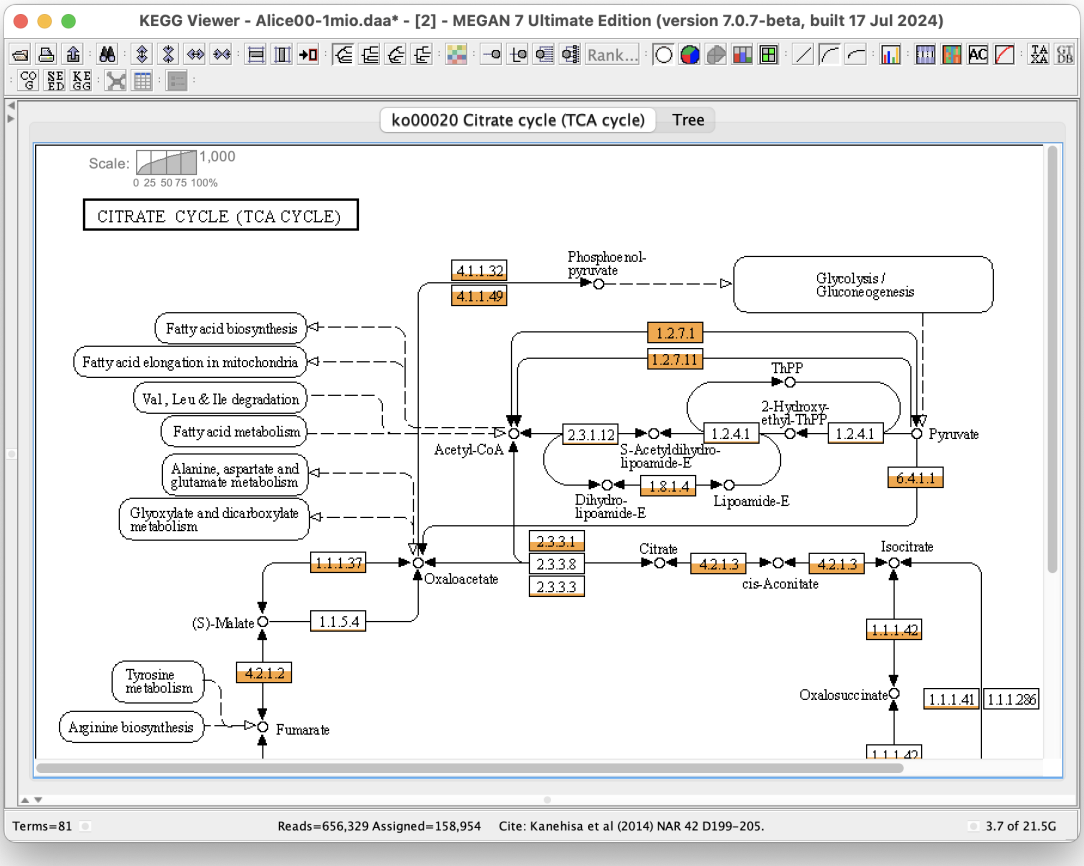

Figure 3.4: Reads mapping to different elements of the citrate cycle, using MEGAN UE.

4 Comparing samples

4.1 Comparing Samples

MEGAN supports the simultaneous analysis of multiple metagenomic samples through its flexible and interactive comparison features.

To begin a comparison, select `File > Compare...`. This opens the **Compare Dialog**, shown in Figure 4.1, which lists all datasets currently loaded in the session. In addition, you can add files that are not yet open. You can select any subset of these to include in the comparison by selecting them.

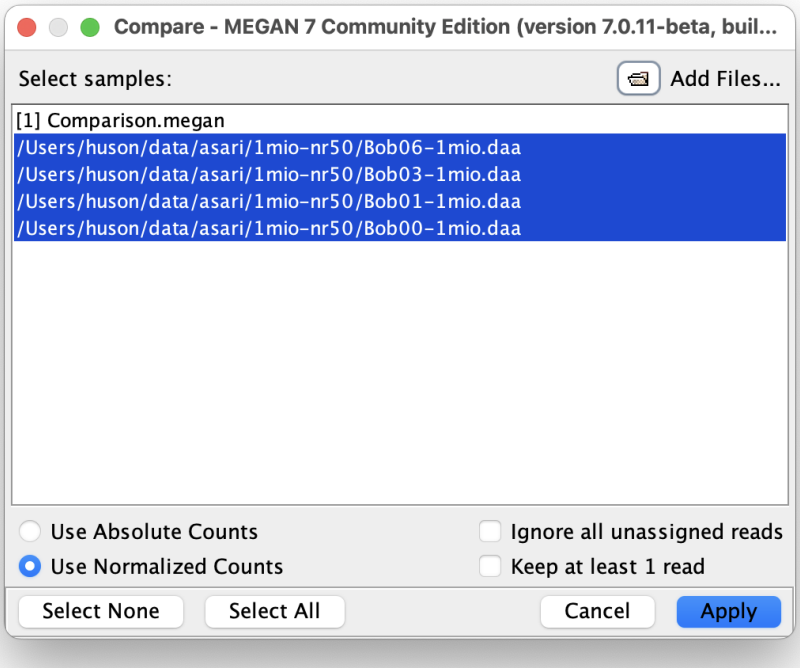

Figure 4.1: The Compare dialog allows selection of multiple samples for comparison.

Once a group of samples is selected, MEGAN creates a new *comparison document*. This document serves as a unified interface for the selected datasets, allowing joint exploration of taxonomic and functional assignments. Each sample retains its identity within the document, making it easy to distinguish and compare across the group.

In the comparison document, MEGAN's full suite of analytical features is available, including:

- Taxonomic and functional tree views where each node shows read counts or relative abundances for each sample.
- Statistical summaries of assigned reads, diversity indices, and unassigned fractions.
- Comparative visualizations, including bar charts, stacked charts, bubble plots, heatmaps, and PCA.
- Interactive filters and color-coding to highlight differences and similarities between samples.

An example comparison of 12 samples is shown in Figure 4.2, where samples are displayed in the taxonomy tree view.

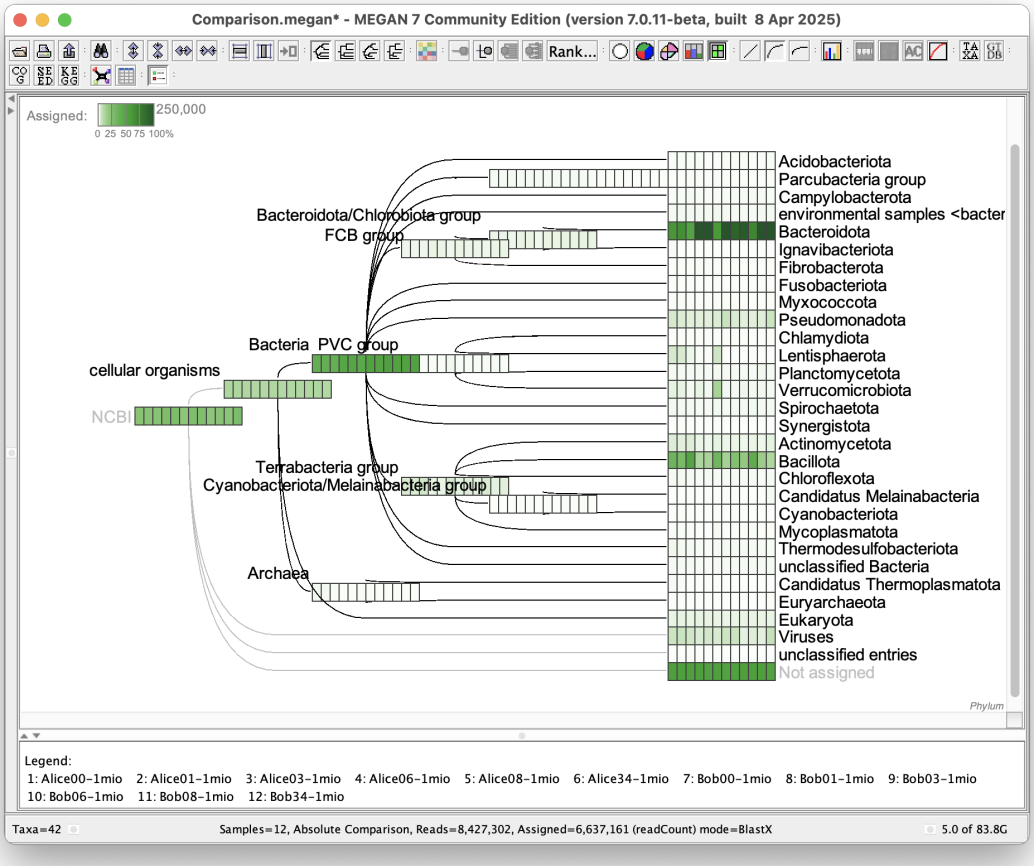

Figure 4.2: A comparison of 12 samples in the taxonomic tree view. Here, a heatmap is used to indicate relative read abundance for each sample.

Users can save comparison documents and the resulting files are very small. This makes them ideal for documenting complex analyses or sharing results with collaborators.

5 Charts

The **Charts Viewer** in MEGAN provides a versatile platform for generating a wide range of interactive plots to visualize taxonomic and functional profiles across metagenomic samples. This viewer is central to comparative analysis and is designed to support both exploratory and publication-ready data presentation.

Overview

The Charts Viewer can be launched via the menu option `Window > Charts`, or is automatically opened when using the `Bar Chart`, `Heatmap`, or other visualization tools are requested from the tool bar. The viewer consists of a chart panel, configuration sidebar, and toolbar for export and customization.

MEGAN supports the following chart types within the Charts Viewer:

- **Bar Chart:** Compare absolute or relative abundances across samples.
- **Stacked Bar Chart:** Visualize the composition of sample categories.
- **Heatmap:** Display abundance patterns using color gradients.
- **Bubble Chart:** Show relative abundance in a circular overlay on tree structures.
- **Line Chart and Stacked Line Chart:** Track changes over time or gradients.
- **Pie Chart:** Visual summary of categorical distribution in a single sample.
- **Box Chart:** Summarize distribution of values across groups.
- **Word Cloud:** Highlight dominant taxa or functions by abundance-weighted font size (see Fig. 5.1).
- **Co-occurrence and Correlation Plots:** Analyze relationships between taxa or functional categories.

Working with the Viewer

Once a chart is displayed, the user can:

- Select and filter samples, taxonomic levels, or functional groups.
- Adjust normalization methods (raw counts, relative abundance, or log scale).
- Customize chart aesthetics, including font sizes, color schemes, and axis settings.

Interactive features include zooming, tooltips for data values, and dynamic legend toggles. The viewer can also synchronize with the taxonomic or functional tree view to maintain focus on selected groups.

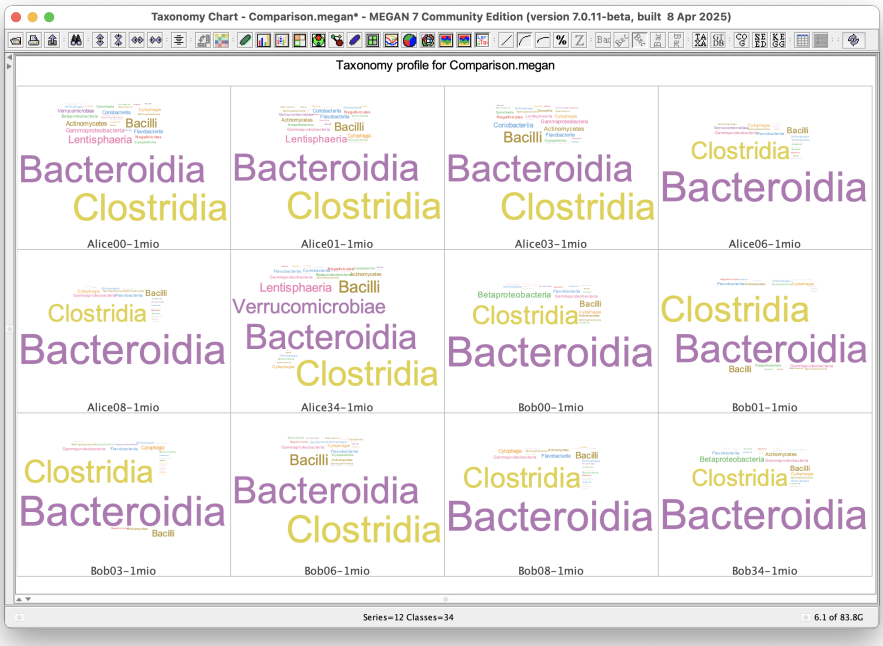

Figure 5.1: A word cloud visualization generated in MEGAN, showing the most abundant taxa in a sample. Font size corresponds to relative abundance, allowing quick identification of dominant groups.

Exporting and Saving

Charts created in the Charts Viewer can be exported using the `Export Image` button or `File > Export Image` menu. MEGAN supports export in:

- `.png` – Raster image format for general use.
- `.svg` – Scalable vector format for editing in graphic software.
- `.pdf` – Ideal for inclusion in scientific publications.

You can also save the current chart configuration as part of the project, allowing you to reopen the viewer later with the same settings.

Use Cases

The Charts Viewer is particularly useful for:

- Visualizing taxonomic shifts between environmental samples.
- Comparing functional profiles across treatments.
- Identifying dominant taxa or rare biosphere members.
- Preparing figures for presentations, reports, or publications.

With its range of supported chart types and intuitive interface, the MEGAN Charts Viewer serves as a powerful tool for comparative metagenomics.

6 Cluster Analysis

MEGAN provides a dedicated Cluster Analysis Viewer for comparing microbial communities based on their taxonomic or functional composition. This tool enables users to assess overall similarities between samples and to visualize complex relationships using a range of clustering and ordination methods.

Overview

The Cluster Analysis Viewer is designed to help users:

- Explore beta diversity across multiple samples.
- Identify grouping patterns based on taxonomic profiles.
- Visualize sample relationships using trees, networks, and multidimensional scaling.

The dialog can be launched from the `Window > Cluster Analysis` menu item. The viewer displays results using four different visualizations:

6.1 Principal Coordinates Analysis (PCoA)

The PCoA plot (Figure 6.1) [Gower, 1966] projects high-dimensional taxonomic or functional profiles into a two- or three-dimensional space, where each point represents a sample. The distance between points reflects dissimilarity in composition.

- Axes represent the principal coordinates explaining the most variation.
- Samples that cluster together in the plot share more similar communities.
- Tooltips and color-coding help identify groupings or outliers.

6.2 UPGMA Tree

The UPGMA (Unweighted Pair Group Method with Arithmetic Mean) tree [Sneath and Sokal, 1957] is a hierarchical clustering method that groups samples by average pairwise distance.

- The resulting dendrogram reflects hierarchical relationships among samples.
- Branch lengths correspond to dissimilarity.
- Useful for identifying broad group-level similarities.

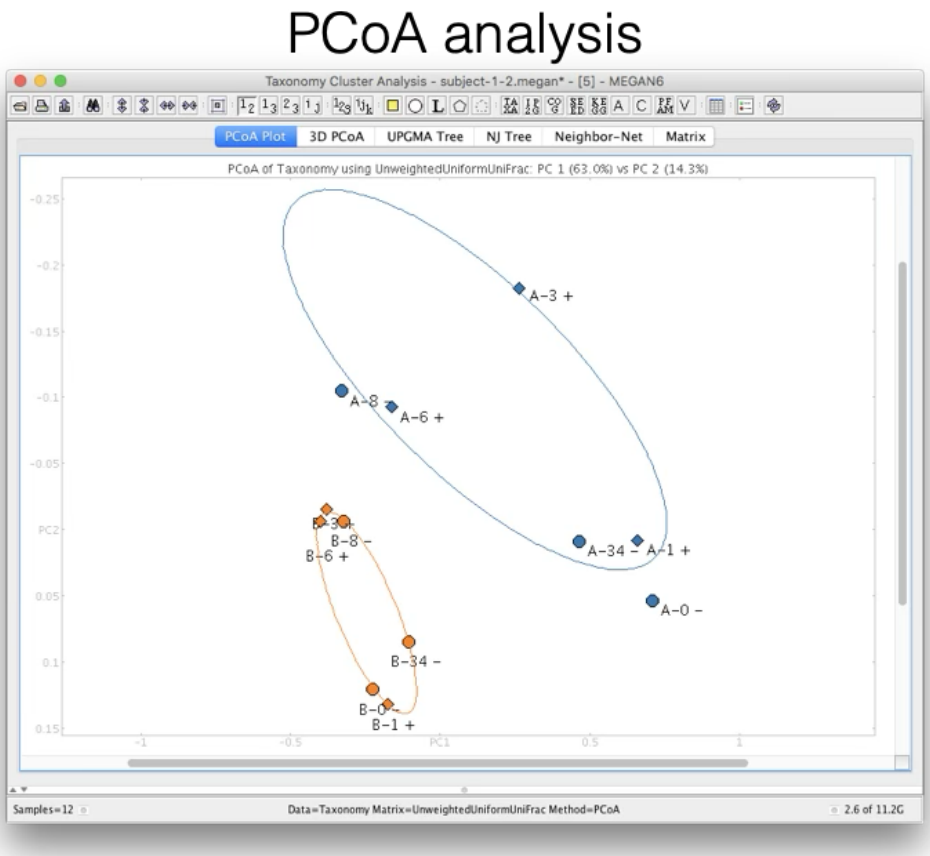

Figure 6.1: PCoA plot showing sample clustering based on taxonomic composition.

6.3 Neighbor Joining Tree

The Neighbor Joining (NJ) method [Saitou and Nei, 1987] produces a tree based on pairwise distances between samples, emphasizing minimal total branch length.

- Often used to approximate evolutionary relationships among samples.
- Branching patterns may differ from UPGMA if the distance data are non-ultrametric.

6.4 Neighbor-Net and Phylogenetic Outline

MEGAN's implementation of Neighbor-Net [Bryant and Moulton, 2004] gives rise to a *phylogenetic outline* [Bağcı et al., 2021], which is a network representation of the relationships among samples.

- Unlike trees, this network allows visualization of conflicting or ambiguous signals in the data.
- Particularly useful when samples show intermediate similarity to multiple groups.
- Phylogenetic outlines provide a powerful tool for identifying gradients and transitions in microbial community structure.

6.5 Interactivity and Export

All views in the Taxonomy Cluster Analysis Viewer are interactive:

- Hovering displays metadata and sample identifiers.
- Color assignments can be changed to reflect groups or metadata categories.
- Each view can be exported as a high-quality image (`.png`, `.svg`, or `.pdf`).

7 Sample Viewer

The Sample Viewer provides a tabular view of all samples present in a document. The samples can have multiple attributes and these attributes can be modified. They can also be used to color the samples. Samples can be extracted or merged in a number of different ways.

The Sample Viewer has a number of specific menus:

7.0.1 The Attributes Menu

The Attributes menu contains the following items:

- The `Attributes > Set Color...` item: Set the color for all selected items.
- The `Attributes > Set Value...` item: Set value for all selected items.
- The `Attributes > New...` item: Create a new attribute (column) in the data table.
- The `Attributes > Import From File...` item: Import one or more attributes from a file into the data table.
- The `Attributes > Duplicate...` item: Duplicate an existing attribute (column).
- The `Attributes > Rename...` item: Rename an existing attribute (column).
- The `Attributes > Delete...` item: Delete an existing attribute (column).
- The `Attributes > Select All Same` item: Select all cells that have the same attribute and value.
- The `Attributes > Compare Absolute` item: Compare samples based on the values of a selected attribute. For example, if the attribute is *gender* and there are some selected samples with value *f* and others with value *m*, then this menu item opens a new comparison document in which two new composite samples labeled *gender:f* and *gender:m* are compared with each other.
- The `Attributes > Compare Relative` item: Same as previous item, except that new samples are scaled so as to have the same size as the smallest new sample.

7.0.2 The Samples Menu

The Samples menu contains the following items:

- The `Samples > Node Shape` submenu.
- The `Samples > Group Nodes` item: Group selected nodes in PCoA plot.
- The `Samples > Ungroup All` item: Ungroup nodes in PCoA plot.
- The `Samples > Add...` item: Add samples from open document.

- The `Samples > Add From File...` item: Add samples from another document.
- The `Samples > Open RMA File...` item: Open the original source RMA file or MEGAN-server file.
- The `Samples > Show All` item: Show all samples.
- The `Samples > Show Selected` item: Show selected samples.
- The `Samples > Hide Selected` item: Hide selected samples.
- The `Samples > Hide Unselected` item: Hide samples.
- The `Samples > Duplicate...` item: Duplicate selected samples (rows).
- The `Samples > Rename...` item: Rename selected samples (rows).
- The `Samples > Delete...` item: Delete an existing sample (row).
- The `Samples > Move Up` item: Move Up.
- The `Samples > Move Down` item: Move samples down.
- The `Samples > Set Color...` item: Set the color for all selected samples.
- The `Samples > Color By Attribute` item: Color samples by attribute states.

7.0.3 The Node Shape Submenu

The Node Shape menu contains the following items:

- The `Node Shape > Circle` item: Circle node shape.
- The `Node Shape > Square` item: Square node shape.
- The `Node Shape > Triangle` item: Triangle node shape.
- The `Node Shape > Diamond` item: Diamond node shape.

7.0.4 The Algorithms Menu

The Algorithms menu contains the following items:

- The `Algorithms > Extract Samples...` item: Extract selected samples to a new document.
- The `Algorithms > Compute Core Biome...` item: Determine taxa and functions that appear in a majority of the selected samples.
- The `Algorithms > Compute Total Biome...` item: Determine total (union) taxonomic and functional content of the selected samples.
- The `Algorithms > Compute Rare Biome...` item: Determine taxa and functions that appear in a minority of the selected samples.
- The `Algorithms > Resample...` item: Resample selected samples to a new document.

8 Concepts and Terminology

MEGAN is designed to help users explore the taxonomic and functional content of metagenomic datasets. To effectively use MEGAN, its important to understand the core concepts and terminology employed by the program.

8.1 Reads

A **read** is a short DNA or RNA sequence obtained from a sequencing experiment. MEGAN typically works with files containing millions of reads that are aligned against a reference database (e.g., NCBI-nr).

8.2 Alignments

An **alignment** refers to the result of matching a read against a reference sequence using a tool like DIAMOND or BLAST. These alignments are stored in formats such as DAA or BLAST tabular format and are imported into MEGAN for analysis.

8.3 Taxonomic Classification

MEGAN assigns reads to taxa based on alignment information and a taxonomy database (e.g., the NCBI taxonomy). It uses algorithms such as the **LCA (Lowest Common Ancestor)** approach, which assigns a read to the most specific taxonomic node that is common to all high-scoring alignments.

8.4 Functional Classification

In addition to taxonomic classification, MEGAN allows reads to be assigned to functional categories. Supported functional databases include:

- **SEED:** A curated set of gene functions organized into subsystems.
- **eggNOG:** Orthologous groups derived from multiple species.
- **KEGG:** Pathways and functional hierarchies (Ultimate Edition only).

8.5 MEGAN Files

MEGAN uses several internal file formats:

- `.megan6` – MEGAN file, a light-weight file that stores the taxonomic and functional counts associated with a single or multiple samples.
- `.rma6` – MEGAN's internal binary format for storing alignment and classification data.

8.6 DAA Files

`.daa` files are binary alignment archives generated by the DIAMOND aligner. These files contain all read-to-reference alignments and can be "meganized", that, processed by the `daa-meganizer` tool, to prepare them for MEGAN.

8.7 LCA Algorithm

The **LCA algorithm** assigns reads to taxa based on all significant alignments. Rather than choosing the best hit, MEGAN assigns a read to the lowest taxonomic node that encompasses all valid hits above a certain threshold (bit score, percent identity, etc.).

8.8 Comparative Analysis

MEGAN enables the comparison of multiple datasets via bar charts, heatmaps, clustering, and principal component analysis (PCA). This is useful for comparing microbial communities across conditions, locations, or sample types.

8.9 Interactive Visualization

MEGAN includes a number of views for exploring data:

- **Taxonomy tree:** Browse and drill down into the taxonomic hierarchy.
- **Functional tree:** Navigate functional classification systems.
- **Inspector panels:** Show detailed information about selected nodes.
- **Charts and plots:** Visually compare samples and categories.
- **Cluster viewer:** Explore samples using PCoA and clustering techniques.

These foundational concepts will be revisited throughout this manual. .

9 Using MEGAN

This chapter introduces the graphical user interface (GUI) of MEGAN and describes how to carry out core workflows, including importing data, exploring classifications, and generating analyses.

9.1 Graphical User Interface (GUI) Overview

When you start MEGAN, the main window appears, consisting of several key areas:

- **Toolbar:** Quick access to commonly used operations such as importing files, saving projects, and switching views.
- **Menu Bar:** Full access to MEGAN's features via the `File`, `Edit`, `View` and other menus.
- **Tree Viewer:** Shows the taxonomic or functional classification tree for a selected sample.
- **Status Bar:** Displays progress, basic stats and memory usage.

9.2 Workflow Walkthroughs

9.2.1 Importing Alignment Files

MEGAN supports input from DIAMOND, BLAST, and LAST:

1. Choose `File > Meganize DAA File` or `File > Import from BLAST`.
2. Select a `.daa` or `.blast` file.
3. Specific the MEGAN `.mbd` mapping file.
4. Optionally, set parameters for taxonomic and functional classification (e.g., LCA thresholds).
5. MEGAN will process the file and then open it, displaying the NCBI taxonomy classification tree.

9.2.2 Taxonomic Classification

- Taxonomic classification is based on alignments and a taxonomy file.
- You can adjust classification parameters via `Options > LCA Parameters`.
- Click on any node to view associated reads, hits, and additional statistics.

9.2.3 Functional Classification

- Functional classifications are based on mappings from reference sequences to systems such as SEED and EggNOG, or KEGG (Ultimate Edition).
- Select the desired classification system using the toolbar.
- Visualize functional categories in a hierarchical tree or as flat bar charts.

9.2.4 Exploring

- Right-click on nodes to inspector data or export data.
- Use the `Inspector` for in-depth read-level exploration.

9.2.5 Exporting Results

MEGAN supports export of:

- Taxonomic and functional profiles as text files.
- Graphics in PNG, SVG, or PDF formats.
- Read lists and sample summaries.
- Reads, matches, corrected reads (long reads mode) and gene-centric assembled reads (short read mode).

To export, use the `File > Export` submenu or right-click context menus in any visualization panel.

[The following is *Ultimate Edition Only.*]

Exporting to SQLITE

The UE program supports export of data to a SQLITE file (see Figure 9.1. There are four different levels of detail:

- Small- save a table of counts
- Medium - save a table of read assignments
- Large - for each read, save all matched accessions
- X-Large - Save all read sequences.

The user can select for which classifications (e.g. taxonomy, KEGG, etc) data should be saved. By default, the data for all reads is saved. Alternatively, only data for selected nodes is saved (this takes a long time, due to the way that classifications are represented).

9.3 Analysis Features

MEGAN includes a rich set of analytical tools:

9.3.1 Comparative Analysis

- Use the Chart viewer or Cluster viewer to compare samples across taxonomic or functional groups.
- Select taxa or categories of interest to highlight variation between samples.

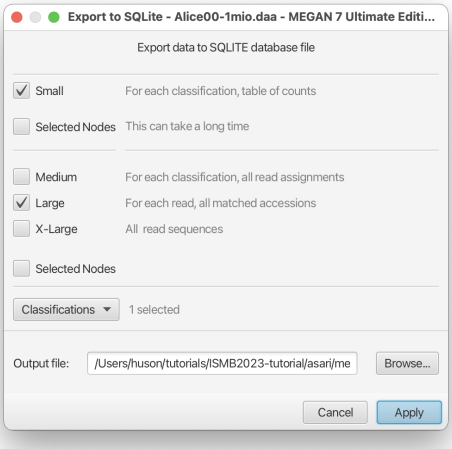

Figure 9.1: The Export to SQLite dialog.

9.3.2 Rarefaction and Diversity

- MEGAN can calculate diversity indices and generate rarefaction curves.
- These tools help assess sampling depth and community richness.

9.3.3 Principal Coordinate Analysis (PCoA)

- PCoA plots are available using the cluster viewer.
- These plots project high-dimensional classification profiles into 2D for visual comparison.

9.3.4 Clustering and Dendrograms

- Cluster samples using UPGMA, neighbor-joining or neighbor-net.
- Dendrograms can reveal similarities and groupings in community structure.

These features allow you to gain insights into the structure and function of your microbial communities, directly within MEGAN's intuitive graphical environment.

9.4 Graphical User Interface (GUI) Details

MEGAN features a comprehensive graphical user interface (GUI) that allows users to perform interactive metagenomic analysis. The GUI is organized into several panels and toolbars, each designed to support a specific part of the workflow.

9.4.1 Main Window Layout

The main window consists of the following key components:

- **Menu Bar** – Located at the top, this provides access to all functionality, organized into menus `File`, `Edit`, `Select`, `Options`, `Layout`, `Tree`, `Window`, and `Help`.
- **Toolbar** – A set of icons beneath the menu bar that offer quick access to frequently used actions like opening files, modifying the visualization, and switching between views.

- **Tree View Panel** – The central panel used for exploring the taxonomic or functional hierarchy of a selected sample. It provides interactive trees with expandable/collapsible nodes.
- **Inspector Window** – Shows detailed information about the currently selected node in the tree, including assigned reads, functional annotations, and alignment statistics.
- **Chart Window** – Displays bar charts, heatmaps, PCA plots, and other comparative visualizations, depending on the active view.
- **Status Bar** – Located at the bottom, this displays messages about import progress, memory usage, and other notifications.

9.4.2 Navigation and Interactivity

MEGAN's GUI is highly interactive:

- Right-clicking on nodes opens context menus for exporting data, collapsing branches, or inspecting data.
- Tooltips appear when hovering over elements, providing summaries and guidance.
- The toolbar the top of the tree panel allow users to switch to different classification viewers (e.g., taxonomy, SEED and EggNOG, or KEGG).

9.4.3 Views and Modes

MEGAN supports multiple viewing modes:

- **Taxonomic View:** Displays reads assigned to taxa based on the LCA algorithm.
- **Functional View:** Displays assignments based on SEED and eggNOG, KEGG or other databases.
- **Comparison View:** Enables comparative analysis across multiple samples using charts and statistical tools.
- **Inspector View:** Lists all reads assigned to the currently selected node with options for further inspection.

9.5 Analysis Features

MEGAN provides a rich set of tools for analyzing and visualizing taxonomic and functional data derived from metagenomic samples. This section outlines the main features available under the `Chart` menu and through various panels in the GUI.

9.5.1 Charts

After opening one sample, or several samples as a comparison document, you can open the Chart Viewer to display several different types of charts.

Attribute Correlation Plot: Displays pairwise correlations between sample attributes using a matrix view.

Bar Chart: Compares read counts or abundances across samples and categories using horizontal or vertical bars.

Box Chart: Shows statistical summaries (median, quartiles, etc.) of distributions, useful for comparing data spread across groups.

Bricks Chart: Uses colored bricks to represent abundance, arranged in a grid to show category composition across samples.

Bubble Chart: Visualizes abundance using circle size over a tree structure, helpful for spotting dominant taxa or functions.

Co-Occurrence Plot: Displays the co-occurrence of taxa or functions across samples, useful for detecting associations.

Correlation Plot: Visualizes correlations between abundance profiles of taxa/functions across samples.

Heat Map: A color-coded matrix showing the relative abundance of taxa or functions across samples.

Line Chart: Plots trends over an ordered series (e.g., time points or sample groups) for selected taxa/functions.

Pie Chart: Represents the proportional abundance of categories within a single sample.

Radial Tree Chart: Displays taxonomic or functional hierarchies in a circular (radial) layout for compact overview.

Stacked Bar Chart: Shows relative contributions of categories stacked within bars across multiple samples.

Stacked Line Chart: Visualizes changing relative abundances across an ordered axis, like time or gradient.

Word Cloud: Displays categories (e.g., taxa or functions) with font sizes reflecting abundance, offering a quick visual summary.

9.5.2 Statistical Analysis Tools

Rarefaction Curves

- Access via `Compare > Rarefaction`.
- Assess sequencing depth and species richness.
- Useful for evaluating sample completeness.

Alpha and Beta Diversity

- Calculate metrics such as Shannon and Simpson diversity.
- View diversity values for individual samples and compare across groups.

9.5.3 Ordination and Clustering

Principal Coordinate Analysis (PCoA)

- Projects high-dimensional abundance data into two or three dimensions.
- Helps visualize relationships between samples.

Hierarchical Clustering

- Samples are grouped based on similarity in taxonomic or functional profiles.
- Methods include UPGMA, neighbor joining and neighbor net.

9.5.4 Tree and Graph Views

MEGAN provides multiple ways to interact with classification hierarchies:

- **Collapsible Trees:** Navigate large hierarchies easily by collapsing/expanding nodes.
- **Node Inspector:** View detailed statistics and read assignments for selected nodes.
- **Highlighting and Selection:** Select multiple nodes to aggregate data or compare across views.

9.5.5 Exporting Visualizations

- Use `File > Export Image` to save any chart or tree as PNG, SVG, or PDF.
- Export data tables using `File > Export Analysis` or node context menus.

These tools support a wide range of metagenomic comparisons, from simple abundance summaries to complex multivariate analyses. For even more advanced use cases, see the next chapter on command-line integration and scripting.

10 Tools

The Linux and MacOS X releases of MEGAN7 provide a number of commandline tools provided in the `tools` directory.

Although these programs are command-line tools, some may require an X server to function. If you are running a tool on a server that lacks an X window system, you can use the X virtual frame buffer" (XVFB) to emulate one. To do this, prepend the following command to your tool's command line:

```
xvfb-run --auto-servernum --server-num=1
```

Here are the tools that are available in MEGAN 7:

10.1 AAdder Build

The `aadder-build` commandline program.

This is used to build an index that can be used to perform functional binning of reads that have been aligned against genomic sequence. It parses files in GFF3 format and creates an index to be used with the `aadder-run` program.

```
SYNOPSIS
        aadder-build [options]
DESCRIPTION
        Build the index for AAdder
OPTIONS
 Input Output
        -igff, --inputGFF [string(s)]        Input GFF3 files or directory (.gz ok). Mandatory
        ↪  option.
        -d, --index [string]                 Index directory. Mandatory option.
 Classification mapping:
        -mdb, --mapDB [string]               MEGAN mapping DB (file megan-map.mdb).
 Deprecated classification mapping options:
        -a2t, --acc2taxa [string]            Accession-to-Taxonomy mapping file.
        -a2eggnog, --acc2eggnog [string]     Accession-to-EGGNOG mapping file.
        -a2gtdb, --acc2gtdb [string]         Accession-to-GTDB mapping file.
        -a2kegg, --acc2kegg [string]         Accession-to-KEGG mapping file.
        -a2seed, --acc2seed [string]         Accession-to-SEED mapping file.
 Other:
        -ex, --extraStrict                   When given an input directory, look inside every
        ↪  input file to check that it is indeed in GFF3 format. Default value: false.
        -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
        -v, --verbose                        Echo commandline options and be verbose. Default
        ↪  value: false.
        -h, --help                           Show program usage and quit.
```

10.2 AAdder Run

The `aadder-run` commandline program.

This is used to add functional assignments to DNA alignments, using an index created using `aadder-build`.

```
SYNOPSIS
        aadder-run [options]
DESCRIPTION
        Adds functional accessions to DNA alignments
OPTIONS
 Input Output
        -i, --input [string(s)]              Input SAM file(s) (.gz ok). Mandatory option.
        -d, --index [string]                 AAdd index directory. Mandatory option.
        -o, --output [string(s)]             Output file(s) (.gz ok) or directory. Mandatory
        ↪  option.
 Other:
        -c, --percentToCover [number]        Percent of alignment that must be covered by
        ↪  protein. Default value: 90.0.
        -rnf, --reportNotFound               Report the names of DNA references for which no
        ↪  functional accession is available. Default value: false.
        -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
        -v, --verbose                        Echo commandline options and be verbose. Default
        ↪  value: false.
        -h, --help                           Show program usage and quit.
```

10.3 Blast to RMA

The `blast2rma` commandline program.

This takes an alignment file as input (can also run on multiple input files), classifies the taxonomic and functional content and then writes out the reads and alignments, together with the classifications and indices to file in RMA (read-match archive) format that can then be opened in MEGAN. For DAA files produced by DIAMOND, use the `daa-meganizer` program instead.

```
SYNOPSIS
        blast2rma [options]
DESCRIPTION
        Computes MEGAN RMA files from  BLAST (or similar) files
OPTIONS
 Input
        -i, --in [string(s)]                 Input BLAST file[s] (.gz ok). Mandatory option.
        -f, --format [string]                Input file format. Mandatory option. Legal values:
        ↪  Unknown, DAA, BlastText, BlastXML, BlastTab, LastMAF, RapSearch2Aln,
        ↪  IlluminaReporter, RDPAssignmentDetails, RDPStandalone, Mothur, SAM,
        ↪  References_as_FastA
        -bm, --blastMode [string]            Blast mode. Default value: Unknown. Legal values:
        ↪  Unknown, BlastN, BlastP, BlastX, Classifier
        -r, --reads [string(s)]              Reads file(s) (fasta or fastq, .gz ok).
        -mdf, --metaDataFile [string(s)]     Files containing metadata to be included in RMA6
        ↪  files.
 Output
        -o, --out [string(s)]                Output file(s), one for each input file, or a
        ↪  directory. Mandatory option.
        -c, --useCompression                 Compress reads and matches in RMA file (smaller
        ↪  files, longer to generate. Default value: true.
 Reads
        -p, --paired                         Reads are paired. Default value: false.
        -ps, --pairedSuffixLength [number]   Length of name suffix used to distinguish between
        ↪  name of read and its mate. Default value: 0.
```

```
       -pof, --pairedReadsInOneFile         Are paired reads in one file (usually they are in
       ↪  two). Default value: false.
Parameters
       -lg, --longReads                     Parse and analyse as long reads. Default value:
       ↪  false.
       -m, --maxMatchesPerRead [number]     Max matches per read. Default value: 100.
       -class, --classify                   Run classification algorithm. Default value: true.
       -ms, --minScore [number]             Min score. Default value: 50.0.
       -me, --maxExpected [number]          Max expected. Default value: 0.01.
       -mpi, --minPercentIdentity [number]   Min percent identity. Default value: 0.0.
       -top, --topPercent [number]          Top percent. Default value: 10.0.
       -supp, --minSupportPercent [number]   Min support as percent of assigned reads (0==off).
       ↪  Default value: 0.01.
       -sup, --minSupport [number]          Min support (0==off). Default value: 0.
       -mrc, --minPercentReadCover [number]   Min percent of read length to be covered by
       ↪  alignments. Default value: 0.0.
       -mrefc, --minPercentReferenceCover [number]   Min percent of reference length to be
       ↪  covered by alignments. Default value: 0.0.
       -mrl, --minReadLength [number]       Minimum read length. Default value: 0.
       -alg, --lcaAlgorithm [string]        Set the LCA algorithm to use for taxonomic
       ↪  assignment. Default value: naive. Legal values: naive, weighted, longReads
       -lcp, --lcaCoveragePercent [number]   Set the percent for the LCA to cover. Default
       ↪  value: 100.0.
       -ram, --readAssignmentMode [string]   Set the read assignment mode. Default value:
       ↪  alignedBases in long read mode, readCount else.
       -cf, --conFile [string]              File of contaminant taxa (one Id or name per line).
Classification support:
       -mdb, --mapDB [string]               MEGAN mapping DB (file megan-map.mdb).
       -on, --only [string(s)]              Use only named classifications (if not set: use
       ↪  all).
Deprecated classification support:
       -tn, --parseTaxonNames               Parse taxon names. Default value: true.
       -a2t, --acc2taxa [string]            Accessopm-to-Taxonomy mapping file.
       -s2t, --syn2taxa [string]            Synonyms-to-Taxonomy mapping file.
       -t4t, --tags4taxonomy [string]       Tags for taxonomy id parsing (must set to activate
       ↪  id parsing).
       -a2eggnog, --acc2eggnog [string]     Accession-to-EGGNOG mapping file.
       -s2eggnog, --syn2eggnog [string]     Synonyms-to-EGGNOG mapping file.
       -t4eggnog, --tags4eggnog [string]    Tags for EGGNOG id parsing (must set to activate id
       ↪  parsing).
       -a2gtdb, --acc2gtdb [string]         Accession-to-GTDB mapping file.
       -s2gtdb, --syn2gtdb [string]         Synonyms-to-GTDB mapping file.
       -t4gtdb, --tags4gtdb [string]        Tags for GTDB id parsing (must set to activate id
       ↪  parsing).
       -a2kegg, --acc2kegg [string]         Accession-to-KEGG mapping file.
       -s2kegg, --syn2kegg [string]         Synonyms-to-KEGG mapping file.
       -t4kegg, --tags4kegg [string]        Tags for KEGG id parsing (must set to activate id
       ↪  parsing).
       -a2seed, --acc2seed [string]         Accession-to-SEED mapping file.
       -s2seed, --syn2seed [string]         Synonyms-to-SEED mapping file.
       -t4seed, --tags4seed [string]        Tags for SEED id parsing (must set to activate id
       ↪  parsing).
       -fwa, --firstWordIsAccession         First word in reference header is accession number
       ↪  (set to 'true' for NCBI-nr downloaded Sep 2016 or later). Default value: true.
       -atags, --accessionTags [string(s)]   List of accession tags. Default value(s): 'gb|'
       ↪  'ref|'.
Other:
       -t, --threads [number]               Number of threads. Default value: 8.
       -cs, --cacheSize [number]            Cache size for SQLITE (use with care). Default
       ↪  value: -10000.
       -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
       -v, --verbose                        Echo commandline options and be verbose. Default
       ↪  value: false.
```

```
       -h, --help                          Show program usage and quit.
```

10.4 Compute Comparison

The `compute-comparison` commandline program.

This is run on a collection of RMA files, MEGAN files and/or meganized DAA files, to obtain a MEGAN file that contains a comparison of the data in the input files.

```
SYNOPSIS
       compute-comparison [options]
DESCRIPTION
       Computes the comparison of multiple megan, RMA or meganized DAA files
OPTIONS
 Input and Output:
       -i, --in [string(s)]                Input RMA and/or meganized DAA files (single
       ↪  directory ok). Mandatory option.
       -o, --out [string]                  Output file. Default value: comparison.megan.
       -mdf, --metaDataFile [string]       Metadata file.
 Options:
       -s, --allowSameNames                All the same sample name to appear multiple times
       ↪  (will add -1, -2 etc). Default value: false.
       -n, --normalize                     Normalize counts. Default value: true.
       -iu, --ignoreUnassignedReads        Ignore unassigned, no-hit or contaminant reads.
       ↪  Default value: false.
       -k1, --keepOne                      In a normalized comparison, non-zero counts are
       ↪  mapped to 1 or more. Default value: false.
       -P, --propertiesFile [string]       Properties file. Default value: Megan.def.
 Other:
       -v, --verbose                       Echo commandline options and be verbose. Default
       ↪  value: false.
       -h, --help                          Show program usage and quit.
```

10.5 DAA Meganizer

The `daa-meganizer` commandline program.

This is run on a single DAA file (or multiple files) computed by DIAMOND, which contain all reads and their alignments to a protein reference database such as NCBI-nr. The program analyzes the alignments and performs taxonomic and functional binning of the reads. The resulting classifications and indexes are attached to the end of the DAA file and the resulting file is referred to as a "meganized DAA file" .

```
SYNOPSIS
       daa-meganizer [options]
DESCRIPTION
       Prepares ('meganizes') a DIAMOND .daa file for use with MEGAN
OPTIONS
 Files
       -i, --in [string(s)]                Input DAA file(s). Each is meganized separately.
       ↪  Mandatory option.
       -mdf, --metaDataFile [string(s)]    Files containing metadata to be included in files.
 Mode
       -lg, --longReads                    Parse and analyse as long reads. Default value:
       ↪  false.
 Parameters
       -class, --classify                  Run classification algorithm. Default value: true.
       -ms, --minScore [number]            Min score. Default value: 50.0.
       -me, --maxExpected [number]         Max expected. Default value: 0.01.
       -mpi, --minPercentIdentity [number]  Min percent identity. Default value: 0.0.
```

```
       -top, --topPercent [number]          Top percent. Default value: 10.0.
       -supp, --minSupportPercent [number]   Min support as percent of assigned reads (0==off).
       ↪  Default value: 0.01.
       -sup, --minSupport [number]          Min support (0==off). Default value: 0.
       -mrc, --minPercentReadCover [number]   Min percent of read length to be covered by
       ↪  alignments. Default value: 0.0.
       -mrefc, --minPercentReferenceCover [number]   Min percent of reference length to be
       ↪  covered by alignments. Default value: 0.0.
       -mrl, --minReadLength [number]       Minimum read length. Default value: 0.
       -alg, --lcaAlgorithm [string]        Set the LCA algorithm to use for taxonomic
       ↪  assignment. Default value: naive. Legal values: naive, weighted, longReads
       -lcp, --lcaCoveragePercent [number]   Set the percent for the LCA to cover. Default
       ↪  value: 100.0.
       -ram, --readAssignmentMode [string]   Set the read assignment mode. Default value:
       ↪  alignedBases in long read mode, readCount else.
       -cf, --conFile [string]              File of contaminant taxa (one Id or name per line).
Classification support:
       -mdb, --mapDB [string]               MEGAN mapping DB (file megan-map.mdb).
       -on, --only [string(s)]              Use only named classifications (if not set: use
       ↪  all).
Deprecated classification support:
       -tn, --parseTaxonNames               Parse taxon names. Default value: true.
       -a2t, --acc2taxa [string]            Accession-to-Taxonomy mapping file.
       -s2t, --syn2taxa [string]            Synonyms-to-Taxonomy mapping file.
       -t4t, --tags4taxonomy [string]       Tags for taxonomy id parsing (must set to activate
       ↪  id parsing).
       -a2eggnog, --acc2eggnog [string]     Accession-to-EGGNOG mapping file.
       -s2eggnog, --syn2eggnog [string]     Synonyms-to-EGGNOG mapping file.
       -t4eggnog, --tags4eggnog [string]    Tags for EGGNOG id parsing (must set to activate id
       ↪  parsing).
       -a2gtdb, --acc2gtdb [string]         Accession-to-GTDB mapping file.
       -s2gtdb, --syn2gtdb [string]         Synonyms-to-GTDB mapping file.
       -t4gtdb, --tags4gtdb [string]        Tags for GTDB id parsing (must set to activate id
       ↪  parsing).
       -a2kegg, --acc2kegg [string]         Accession-to-KEGG mapping file.
       -s2kegg, --syn2kegg [string]         Synonyms-to-KEGG mapping file.
       -t4kegg, --tags4kegg [string]        Tags for KEGG id parsing (must set to activate id
       ↪  parsing).
       -a2seed, --acc2seed [string]         Accession-to-SEED mapping file.
       -s2seed, --syn2seed [string]         Synonyms-to-SEED mapping file.
       -t4seed, --tags4seed [string]        Tags for SEED id parsing (must set to activate id
       ↪  parsing).
       -fwa, --firstWordIsAccession         First word in reference header is accession number
       ↪  (set to 'true' for NCBI-nr downloaded Sep 2016 or later). Default value: true.
       -atags, --accessionTags [string(s)]   List of accession tags. Default value(s): 'gb|'
       ↪  'ref|'.
Other:
       -t, --threads [number]               Number of threads. Default value: 8.
       -cs, --cacheSize [number]            Cache size for SQLITE (use with care). Default
       ↪  value: -10000.
       -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
       -v, --verbose                        Echo commandline options and be verbose. Default
       ↪  value: false.
       -h, --help                           Show program usage and quit.
```

10.6 DAA to GFF3

The `daa2gff3` commandline program.

This produces an annotation of long reads or contigs, given a meganized DAA file as input. The output is in GFF3 format.

```
SYNOPSIS
        Daa2Gff3 [options]
DESCRIPTION
        Extracts a GFF3 annotation file from a meganized DAA file
OPTIONS
 Input and Output
        -i, --in [string]                    Input meganized DAA file. Mandatory option.
        -o, --out [string]                   Output file (stdout or .gz ok). Default value:
        ↪  stdout.
 Options
        -c, --classification [string]        Name of classification to report, or 'all'. Default
        ↪  value: all.
        -k, --incompatible                   Include incompatible. Default value: false.
        -d, --dominated                      Include dominated. Default value: false.
        -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
 Other:
        -v, --verbose                        Echo commandline options and be verbose. Default
        ↪  value: false.
        -h, --help                           Show program usage and quit.
```

10.7 DAA to Info

The `daa2info` commandline program.

This is run on a single DAA file computed by DIAMOND and possibly meganized, to obtain information or to extract data such as reads, alignments or classifications.

```
SYNOPSIS
        daa2info [options]
DESCRIPTION
        Analyses a DIAMOND file
OPTIONS
 Input and Output
        -i, --in [string]                    Input DAA file. Mandatory option.
        -o, --out [string]                   Output file (stdout or .gz ok). Default value:
        ↪  stdout.
 Commands
        -l, --list                           List general info about file. Default value: false.
        -m, --listMore                       List more info about file (if meganized). Default
        ↪  value: false.
        -c2c, --class2count [string(s)]      List class to count for named classification(s)
        ↪  (Possible values: EGGNOG GTDB KEGG SEED Taxonomy).
        -r2c, --read2class [string(s)]       List read to class assignments for named
        ↪  classification(s) (Possible values: EGGNOG GTDB KEGG SEED Taxonomy).
        -n, --names                          Report class names rather than class Id numbers.
        ↪  Default value: false.
        -p, --paths                          Report class paths rather than class Id numbers.
        ↪  Default value: false.
        -r, --prefixRank                     When reporting class paths for taxonomy, prefix
        ↪  single letter to indicate taxonomic rank. Default value: false.
        -mro, --majorRanksOnly               Only use major taxonomic ranks. Default value:
        ↪  false.
        -bo, --bacteriaOnly                  Only report bacterial reads and counts in taxonomic
        ↪  report. Default value: false.
        -vo, --virusOnly                     Only report viral reads and counts in taxonomic
        ↪  report. Default value: false.
        -u, --ignoreUnassigned               Don't report on reads that are unassigned. Default
        ↪  value: true.
        -s, --sum                            Use summarized rather than assigned counts when
        ↪  listing class to count. Default value: false.
        -es, --extractSummaryFile [string]   Output a MEGAN summary file (contains all
        ↪  classifications, but no reads or alignments).
```

```
       -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
 Other:
       -v, --verbose                        Echo commandline options and be verbose. Default
       ↪  value: false.
       -h, --help                           Show program usage and quit.
```

10.8 DAA to RMA

The `daa2rma` commandline program. Please use `daa-meganizer` instead.

This takes a DAA alignment file produced by DIAMOND, performs taxonomic and classification of reads based on the alignments, and writes out the reads and alignments, together with the classifications and indices to file in RMA (read-match archive) format that can then be opened in MEGAN. Do not use this legacy program, rather use the `daa-meganizer` program instead.

```
SYNOPSIS
       add2rma [options]
DESCRIPTION
       Computes a MEGAN .rma6 file from a DIAMOND .daa file
OPTIONS
 Input
       -i, --in [string(s)]                 Input DAA file. Mandatory option.
       -mdf, --metaDataFile [string(s)]     Files containing metadata to be included in RMA6
       ↪  files.
 Output
       -o, --out [string(s)]                Output file(s), one for each input file, or a
       ↪  directory. Mandatory option.
       -c, --useCompression                 Compress reads and matches in RMA file (smaller
       ↪  files, longer to generate. Default value: true.
 Reads
       -p, --paired                         Reads are paired. Default value: false.
       -ps, --pairedSuffixLength [number]   Length of name suffix used to distinguish between
       ↪  name (i.e. first word in header) of read and its mate (use 0 if read and mate have
       ↪  same name). Default value: 0.
       -pof, --pairedReadsInOneFile         Are paired reads in one file (usually they are in
       ↪  two). Default value: false.
 Parameters
       -lg, --longReads                     Parse and analyse as long reads. Default value:
       ↪  false.
       -m, --maxMatchesPerRead [number]     Max matches per read. Default value: 100.
       -class, --classify                   Run classification algorithm. Default value: true.
       -ms, --minScore [number]             Min score. Default value: 50.0.
       -me, --maxExpected [number]          Max expected. Default value: 0.01.
       -mpi, --minPercentIdentity [number]   Min percent identity. Default value: 0.0.
       -top, --topPercent [number]          Top percent. Default value: 10.0.
       -supp, --minSupportPercent [number]   Min support as percent of assigned reads (0==off).
       ↪  Default value: 0.01.
       -sup, --minSupport [number]          Min support (0==off). Default value: 0.
       -mrc, --minPercentReadCover [number]   Min percent of read length to be covered by
       ↪  alignments. Default value: 0.0.
       -mrefc, --minPercentReferenceCover [number]   Min percent of reference length to be
       ↪  covered by alignments. Default value: 0.0.
       -mrl, --minReadLength [number]       Minimum read length. Default value: 0.
       -alg, --lcaAlgorithm [string]        Set the LCA algorithm to use for taxonomic
       ↪  assignment. Default value: naive. Legal values: naive, weighted, longReads
       -lcp, --lcaCoveragePercent [number]   Set the percent for the LCA to cover. Default
       ↪  value: 100.0.
       -ram, --readAssignmentMode [string]   Set the read assignment mode. Default value:
       ↪  alignedBases in long read mode, readCount else.
       -cf, --conFile [string]              File of contaminant taxa (one Id or name per line).
 Classification support:
       -mdb, --mapDB [string]               MEGAN mapping DB (file megan-map.mdb).
```

```
       -on, --only [string(s)]              Use only named classifications (if not set: use
       ↪  all).
 Deprecated classification support:
       -tn, --parseTaxonNames               Parse taxon names. Default value: true.
       -a2t, --acc2taxa [string]            Accessopm-to-Taxonomy mapping file.
       -s2t, --syn2taxa [string]            Synonyms-to-Taxonomy mapping file.
       -t4t, --tags4taxonomy [string]       Tags for taxonomy id parsing (must set to activate
       ↪  id parsing).
       -a2eggnog, --acc2eggnog [string]     Accession-to-EGGNOG mapping file.
       -s2eggnog, --syn2eggnog [string]     Synonyms-to-EGGNOG mapping file.
       -t4eggnog, --tags4eggnog [string]    Tags for EGGNOG id parsing (must set to activate id
       ↪  parsing).
       -a2gtdb, --acc2gtdb [string]         Accession-to-GTDB mapping file.
       -s2gtdb, --syn2gtdb [string]         Synonyms-to-GTDB mapping file.
       -t4gtdb, --tags4gtdb [string]        Tags for GTDB id parsing (must set to activate id
       ↪  parsing).
       -a2kegg, --acc2kegg [string]         Accession-to-KEGG mapping file.
       -s2kegg, --syn2kegg [string]         Synonyms-to-KEGG mapping file.
       -t4kegg, --tags4kegg [string]        Tags for KEGG id parsing (must set to activate id
       ↪  parsing).
       -a2seed, --acc2seed [string]         Accession-to-SEED mapping file.
       -s2seed, --syn2seed [string]         Synonyms-to-SEED mapping file.
       -t4seed, --tags4seed [string]        Tags for SEED id parsing (must set to activate id
       ↪  parsing).
       -fwa, --firstWordIsAccession         First word in reference header is accession number
       ↪  (set to 'true' for NCBI-nr downloaded Sep 2016 or later). Default value: true.
       -atags, --accessionTags [string(s)]   List of accession tags. Default value(s): 'gb|'
       ↪  'ref|'.
 Other:
       -t, --threads [number]               Number of threads. Default value: 8.
       -cs, --cacheSize [number]            Cache size for SQLITE (use with care). Default
       ↪  value: -10000.
       -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
       -v, --verbose                        Echo commandline options and be verbose. Default
       ↪  value: false.
       -h, --help                           Show program usage and quit.
```

10.9 Extract Biome

The `extract-biome` commandline program.

This can be used to extract the total, core or rare biome from a MEGAN comparison file.

```
SYNOPSIS
       ExtractBiome [options]
DESCRIPTION
       Extracts the total, core or rare biome from a MEGAN comparison file
OPTIONS
 Input and Output:
       -i, --in [string]                    Input MEGAN comparison file (.megan file).
       ↪  Mandatory option.
       -o, --out [string]                   Output file. Default value: biome.megan.
 Options:
       -b, --biome [string]                 Biome type to compute. Default value: total. Legal
       ↪  values: total, core, rare
       -s, --samples [string(s)]            Samples to use or 'ALL'. Default value(s): 'ALL'.
       -stp, --sampleThresholdPercent [number]   Min or max percent of samples that class must
       ↪  be present in to be included in core or rare biome, resp.. Default value: 50.0.
       -ctp, --classThresholdPercent [number]   Min percent of sample that reads assigned to
       ↪  class must achieve for class to be considered present in sample. Default value: 0.1.
       -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
 Other:
```

```
       -v, --verbose                          Echo commandline options and be verbose. Default
       ↪  value: false.
       -h, --help                             Show program usage and quit.
```

10.10 GC Assembler

The `gc-assembler` commandline program.

This can be used to perform gene-centric assembly, also called protein-alignment-guided assembly [Huson et al., 2017]. Genes are assembled on-the-fly, based on the alignment of all reads against a protein reference database such as NCBI-nr. Specifically, the user selects a gene family based on a classification such as KEGG and all reads binned to that gene family are assembled.

```
SYNOPSIS
       gc-assembler [options]
DESCRIPTION
       Gene-centric assembly
OPTIONS
 Input and output
       -i, --input [string]                   Input DAA or RMA6 file. Mandatory option.
       -o, --output [string]                  Output filename template, use %d or %s to represent
       ↪  class id or name, respectively. Default value: input-%d.fasta.
 Classification
       -fun, --function [string]              Name of functional classification (choices: EGGNOG,
       ↪  GTDB, KEGG, SEED, none). Mandatory option.
       -id, --ids [string(s)]                 Names or ids of classes to assemble, or keyword ALL
       ↪  for all. Mandatory option.
 Options
       -mor, --minOverlapReads [number]       Minimum overlap for two reads. Default value: 20.
       -len, --minLength [number]             Minimum contig length. Default value: 200.
       -reads, --minReads [number]            Minimum number of reads. Default value: 2.
       -mac, --minAvCoverage [number]         Minimum average coverage. Default value: 1.
       -c, --overlapContigs                   Attempt to overlap contigs. Default value: true.
       -moc, --minOverlapContigs [number]     Minimum overlap for two contigs. Default value: 20.
       -mic, --minPercentIdentityContigs [number]   Mininum percent identity to merge contigs.
       ↪  Default value: 98.0.
 Other:
       -t, --threads [number]                 Number of worker threads. Default value: 8.
       -vv, --veryVerbose                     Report program is very verbose detail. Default
       ↪  value: false.
       -P, --propertiesFile [string]          Properties file. Default value: Megan.def.
       -v, --verbose                          Echo commandline options and be verbose. Default
       ↪  value: false.
       -h, --help                             Show program usage and quit.
```

10.11 Megan Server

The `megan-server` commandline program.

This can serve files that are hosted on a server to instances of MEGAN running on personal computers, over the web [Ruscheweyh and Huson, 2015]. This uses a lightweight Representational State Transfer - Application Programming Interface (REST-API)-based framework written in Java. This allows researchers to analyze files produced by the DIAMOND+MEGAN pipeline on their local computer, without any need to download data from a server directory. Data is sent and received via REST which makes reuse of HTTP technology. While users can access the files using a web browser, they will usually access the files via the MEGAN, which acts as a client and communicates with instances of MeganServer.

```
SYNOPSIS
        megan-server [options]
DESCRIPTION
        Serves MEGAN files over the web via HTTP
OPTIONS
 Input
        -i, --input [string]                Input directory. Mandatory option.
        -r, --recurse                       Recursively visit all input subdirectories. Default
        ↪  value: true.
        -x, --extensions [string(s)]        Input file extensions. Default value(s): '.daa'
        ↪  '.rma' '.rma6' '.megan' '.megan.gz'.
 Server
        -e, --endpoint [string]             Endpoint name. Default value: megan7server.
        -p, --port [number]                 Server port. Default value: 8001.
        -g, --allowGuest                    Allow guest login (name: guest, pwd: guest).
        ↪  Default value: false.
 Other:
        -u, --usersFile [string]            File containing list of users. Default value:
        ↪  MeganServerUsers.def.
        -bl, --backlog [number]             Set the socket backlog. Default value: 100.
        -pt, --pageTimeout [number]         Number of seconds to keep pending pages alive.
        ↪  Default value: 10000.
        -rpp, --readsPerPage [number]       Number of reads per page to serve. Default value:
        ↪  100.
        -t, --threads [number]              Number of threads. Default value: 8.
        -d, --debug                         Debug mode. Default value: false.
        -v, --verbose                       Echo commandline options and be verbose. Default
        ↪  value: false.
        -h, --help                          Show program usage and quit.
```

10.12 Merge Files

The `merge-files` commandline program.

This takes multiple RMA or meganized DAA files as input and produces a single MEGAN file as output that represents the merged set of all reads in the input files.

```
SYNOPSIS
        MergeFiles [options]
DESCRIPTION
        Computes the comparison of multiple megan, RMA or meganized DAA files
OPTIONS
 Input and Output:
        -i, --in [string(s)]                Input RMA and/or meganized DAA files (single
        ↪  directory ok). Mandatory option.
        -o, --out [string]                  Output file. Default value: merged.megan.
        -mdf, --metaDataFile [string]       Metadata file.
        -P, --propertiesFile [string]       Properties file. Default value: Megan.def.
 Other:
        -v, --verbose                       Echo commandline options and be verbose. Default
        ↪  value: false.
        -h, --help                          Show program usage and quit.
```

10.13 Read Extractor

The `read-extractor` commandline program.

This can be used to extract reads from meganized DAA files or RMA files based on their assignment to taxonomic or functional classes. In the case of long reads or contigs, it can also be used to extract frame-shift corrected versions of the sequences [Arumugam et al., 2019].

```
SYNOPSIS
        read-extractor [options]
DESCRIPTION
        Extracts reads from a DAA or RMA file by classification
OPTIONS
 Input and Output
        -i, --input [string(s)]              Input DAA and/or RMA file(s). Mandatory option.
        -o, --output [string(s)]             Output file(s). Use %f for input file name, %t for
        ↪  class name and %i for class id. (Directory, stdout, .gz ok). Default value(s):
        ↪  'stdout'.
 Options
        -fsc, --frameShiftCorrect            Extract frame-shift corrected reads. Default value:
        ↪  false.
        -c, --classification [string]        The classification to use.  Legal values: EGGNOG,
        ↪  GTDB, KEGG, SEED, Taxonomy
        -n, --classNames [string(s)]         Names (or ids) of classes to extract reads from
        ↪  (default: extract all classes).
        -b, --allBelow                       Report all reads assigned to or below a named
        ↪  class. Default value: false.
        -a, --all                            Extract all reads (not by class). Default value:
        ↪  false.
 Other:
        -IE, --ignoreExceptions              Ignore exceptions and continue processing. Default
        ↪  value: false.
        -gz, --gzipOutputFiles               If output directory is given, gzip files written to
        ↪  directory. Default value: true.
        -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
        -v, --verbose                        Echo commandline options and be verbose. Default
        ↪  value: false.
        -h, --help                           Show program usage and quit.
```

10.14 Reanalyzer

The `reanalyzer` commandline program.

This is used to rerun taxonomic and functional binning on an RMA or meganized DAA file.

```
SYNOPSIS
        Reanalyzer [options]
DESCRIPTION
        Reanalyze DAA and RMA files
OPTIONS
        -i, --input [string(s)]              Input  file. Mandatory option.
 Parameters
        -lg, --longReads                     Parse and analyse as long reads. Default value:
        ↪  false.
        -class, --classify                   Run classification algorithm. Default value: true.
        -ms, --minScore [number]             Min score (-1: no change). Default value: -1.0.
        -me, --maxExpected [number]          Max expected (-1: no change). Default value: -1.0.
        -mpi, --minPercentIdentity [number]   Min percent identity (-1: no change). Default
        ↪  value: -1.0.
        -top, --topPercent [number]          Top percent (-1: no change). Default value: -1.0.
        -supp, --minSupportPercent [number]   Min support as percent of assigned reads (0: off,
        ↪  -1: no change). Default value: -1.0.
        -sup, --minSupport [number]          Min support (0: off, -1; no change). Default value:
        ↪  -1.
        -mrc, --minPercentReadCover [number]   Min percent of read length to be covered by
        ↪  alignments (-1: no change). Default value: -1.0.
        -mrefc, --minPercentReferenceCover [number]   Min percent of reference length to be
        ↪  covered by alignments (-1: no change). Default value: -1.0.
        -alg, --lcaAlgorithm [string]        Set the LCA algorithm to use for taxonomic
        ↪  assignment. Default value: naive. Legal values: naive, weighted, longReads
```

```
       -lcp, --lcaCoveragePercent [number]   Set the percent for the LCA to cover (-1: no
       ↪  change). Default value: -1.0.
       -ram, --readAssignmentMode [string]   Set the read assignment mode. Default value:
       ↪  alignedBases in long read mode, readCount else.
       -cf, --conFile [string]               File of contaminant taxa (one Id or name per line).
       -pr, --paired                         Reads are paired. Default value: false.
       -P, --propertiesFile [string]         Properties file. Default value: Megan.def.
 Other:
       -v, --verbose                         Echo commandline options and be verbose. Default
       ↪  value: false.
       -h, --help                            Show program usage and quit.
```

10.15 RMA to Info

The `rma2info` commandline program.

This is run on a single RMA file to obtain information or to extract data such as reads, alignments or classifications.

```
SYNOPSIS
       rma2info [options]
DESCRIPTION
       Analyses an RMA file
OPTIONS
 Input and Output
       -i, --in [string]                     Input RMA file. Mandatory option.
       -o, --out [string]                    Output file (stdout or .gz ok). Default value:
       ↪  stdout.
 Commands
       -l, --list                            List general info about file. Default value: false.
       -m, --listMore                        List more info about file (if meganized). Default
       ↪  value: false.
       -c2c, --class2count [string(s)]       List class to count for named classification(s)
       ↪  (Possible values: EGGNOG GTDB KEGG SEED Taxonomy).
       -r2c, --read2class [string(s)]        List read to class assignments for named
       ↪  classification(s) (Possible values: EGGNOG GTDB KEGG SEED Taxonomy).
       -n, --names                           Report class names rather than class Id numbers.
       ↪  Default value: false.
       -p, --paths                           Report class paths rather than class Id numbers.
       ↪  Default value: false.
       -r, --ranks                           When reporting taxonomy, report taxonomic rank
       ↪  using single letter (K for Kingdom, P for Phylum etc). Default value: false.
       -mro, --majorRanksOnly                Only use major taxonomic ranks. Default value:
       ↪  false.
       -bo, --bacteriaOnly                   Only report bacterial reads and counts in taxonomic
       ↪  report. Default value: false.
       -vo, --virusOnly                      Only report viral reads and counts in taxonomic
       ↪  report. Default value: false.
       -u, --ignoreUnassigned                Don't report on reads that are unassigned. Default
       ↪  value: true.
       -s, --sum                             Use summarized rather than assigned counts when
       ↪  listing class to count. Default value: false.
       -es, --extractSummaryFile [string]    Output a MEGAN summary file (contains all
       ↪  classifications, but no reads or alignments).
       -P, --propertiesFile [string]         Properties file. Default value: Megan.def.
 Other:
       -v, --verbose                         Echo commandline options and be verbose. Default
       ↪  value: false.
       -h, --help                            Show program usage and quit.
```

10.16 Sam to RMA

The `sam2rma` commandline program.

This takes an alignment file in SAM format, produced by MALT, as input (can also run on multiple input files), classifies the taxonomic and functional content and then writes out the reads and alignments, together with the classifications and indices to file in RMA (read-match archive) format that can then be opened in MEGAN.

```
SYNOPSIS
        sam2rma [options]
DESCRIPTION
        Computes a MEGAN RMA (.rma) file from a SAM (.sam) file that was created by DIAMOND or
        ↪  MALT
OPTIONS
 Input
        -i, --in [string(s)]                 Input SAM file[s] generated MALT (gzipped ok).
        ↪  Mandatory option.
        -r, --reads [string(s)]              Reads file(s) (fasta or fastq, gzipped ok).
        -mdf, --metaDataFile [string(s)]     Files containing metadata to be included in RMA6
        ↪  files.
 Output
        -o, --out [string(s)]                Output file(s), one for each input file, or a
        ↪  directory. Mandatory option.
        -c, --useCompression                 Compress reads and matches in RMA file (smaller
        ↪  files, longer to generate. Default value: true.
 Reads
        -p, --paired                         Reads are paired. Default value: false.
        -ps, --pairedSuffixLength [number]   Length of name suffix used to distinguish between
        ↪  name of read and its mate. Default value: 0.
 Parameters
        -lg, --longReads                     Parse and analyse as long reads. Default value:
        ↪  false.
        -m, --maxMatchesPerRead [number]     Max matches per read. Default value: 100.
        -class, --classify                   Run classification algorithm. Default value: true.
        -ms, --minScore [number]             Min score. Default value: 50.0.
        -me, --maxExpected [number]          Max expected. Default value: 0.01.
        -top, --topPercent [number]          Top percent. Default value: 10.0.
        -supp, --minSupportPercent [number]   Min support as percent of assigned reads (0==off).
        ↪  Default value: 0.01.
        -sup, --minSupport [number]          Min support. Default value: 0.
        -mrc, --minPercentReadCover [number]   Min percent of read length to be covered by
        ↪  alignments. Default value: 0.0.
        -mrefc, --minPercentReferenceCover [number]   Min percent of reference length to be
        ↪  covered by alignments. Default value: 0.0.
        -mrl, --minReadLength [number]       Minimum read length. Default value: 0.
        -alg, --lcaAlgorithm [string]        Set the LCA algorithm to use for taxonomic
        ↪  assignment. Default value: naive. Legal values: naive, weighted, longReads
        -lcp, --lcaCoveragePercent [number]   Set the percent for the LCA to cover. Default
        ↪  value: 100.0.
        -ram, --readAssignmentMode [string]   Set the read assignment mode. Default value:
        ↪  alignedBases in long read mode, readCount else.
        -cf, --conFile [string]              File of contaminant taxa (one Id or name per line).
 Classification support:
        -mdb, --mapDB [string]               MEGAN mapping DB (file megan-map.mdb).
        -on, --only [string(s)]              Use only named classifications (if not set: use
        ↪  all).
 Deprecated classification support:
        -tn, --parseTaxonNames               Parse taxon names. Default value: true.
        -a2t, --acc2taxa [string]            Accession-to-Taxonomy mapping file.
        -s2t, --syn2taxa [string]            Synonyms-to-Taxonomy mapping file.
        -t4t, --tags4taxonomy [string]       Tags for taxonomy id parsing (must set to activate
        ↪  id parsing).
```

```
       -a2eggnog, --acc2eggnog [string]     Accession-to-EGGNOG mapping file.
       -s2eggnog, --syn2eggnog [string]     Synonyms-to-EGGNOG mapping file.
       -t4eggnog, --tags4eggnog [string]    Tags for EGGNOG id parsing (must set to activate id
       ↪  parsing).
       -a2gtdb, --acc2gtdb [string]         Accession-to-GTDB mapping file.
       -s2gtdb, --syn2gtdb [string]         Synonyms-to-GTDB mapping file.
       -t4gtdb, --tags4gtdb [string]        Tags for GTDB id parsing (must set to activate id
       ↪  parsing).
       -a2kegg, --acc2kegg [string]         Accession-to-KEGG mapping file.
       -s2kegg, --syn2kegg [string]         Synonyms-to-KEGG mapping file.
       -t4kegg, --tags4kegg [string]        Tags for KEGG id parsing (must set to activate id
       ↪  parsing).
       -a2seed, --acc2seed [string]         Accession-to-SEED mapping file.
       -s2seed, --syn2seed [string]         Synonyms-to-SEED mapping file.
       -t4seed, --tags4seed [string]        Tags for SEED id parsing (must set to activate id
       ↪  parsing).
       -fwa, --firstWordIsAccession         First word in reference header is accession number
       ↪  (set to 'true' for NCBI-nr downloaded Sep 2016 or later). Default value: true.
       -atags, --accessionTags [string(s)]   List of accession tags. Default value(s): 'gb|'
       ↪  'ref|'.
 Other:
       -t, --threads [number]               Number of threads. Default value: 8.
       -cs, --cacheSize [number]            Cache size for SQLITE (use with care). Default
       ↪  value: -10000.
       -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
       -v, --verbose                        Echo commandline options and be verbose. Default
       ↪  value: false.
       -h, --help                           Show program usage and quit.
```

10.17 Taxonomy to Function

The `taxonomy2function` commandline program.

This is used to extract taxonomy-by-function classifications.

```
SYNOPSIS
       taxonomy2function [options]
DESCRIPTION
       Reports taxonomy-by-function classification
OPTIONS
 Input and Output
       -i, --in [string(s)]                 Input file(s). Mandatory option.
       -o, --out [string]                   Output file (stdout or .gz ok). Default value:
       ↪  stdout.
 Options
       -a, --firstClassification [string]   First classification name. Default value: Taxonomy.
       ↪  Legal values: EGGNOG, GTDB, KEGG, SEED, Taxonomy
       -ac, --firstClasses [string(s)]      Class IDs in first classification?. Default
       ↪  value(s): 'all'.
       -b, --secondClassification [string]   Second classification name. Default value: EGGNOG.
       ↪  Legal values: EGGNOG, GTDB, KEGG, SEED, Taxonomy
       -bc, --secondClasses [string(s)]     Class IDs in second classifications?. Default
       ↪  value(s): 'all'.
       -af, --firstFormat [string]          Format to report first classification class.
       ↪  Default value: name. Legal values: name, id, path
       -bf, --secondFormat [string]         Format to report second classification class.
       ↪  Default value: name. Legal values: name, id, path
       -l, --list [string]                  List counts or read names?. Default value: counts.
       ↪  Legal values: counts, reads
       -mro, --majorRanksOnly               Only use major ranks for NCBI taxonomy. Default
       ↪  value: false.
       -s, --separator [string]             Separator. Default value: tab. Legal values: tab,
       ↪  comma, semi-colon
```

```
        -au, --includeFirstUnassigned        include reads unassigned in first classification.
        ↪  Default value: true.
        -bu, --includeSecondUnassigned       include reads unassigned second classification.
        ↪  Default value: true.
 Other:
        -ar, --firstRank [string]            If the first classification is Taxonomy, report at
        ↪  specified rank.  Legal values: Domain, Kingdom, Phylum, Class, Order, Family, Genus,
        ↪  Species
        -br, --secondRank [string]           If the second classification is Taxonomy, report at
        ↪  specified rank.  Legal values: Domain, Kingdom, Phylum, Class, Order, Family, Genus,
        ↪  Species
        -sh, --showHeadline                  Show a headline in the output naming
        ↪  classifications and files. Default value: false.
        -ps, --pathSeparator [string]        Separator used when reporting paths. Default value:
        ↪  ::. Legal values: ::, |, tab, comma, semi-colon
        -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
        -v, --verbose                        Echo commandline options and be verbose. Default
        ↪  value: false.
        -h, --help                           Show program usage and quit.
```

10.18 Megan Server

The `megan-server` commandline program.

```
SYNOPSIS
 megan-server [options]
DESCRIPTION
 Serves MEGAN files over the web via HTTP
OPTIONS
 Input
 -i, --input [string]                 Input directory. Mandatory option.
 -r, --recurse                        Recursively visit all input subdirectories. Default value: true.
 -x, --extensions [string(s)]         Input file extensions. Default value(s): '.daa' '.rma' '.rma6' '.megan' '
 Server
 -e, --endpoint [string]              Endpoint name. Default value: megan7server.
 -p, --port [number]                  Server port. Default value: 8001.
 -g, --allowGuest                     Allow guest login (name: guest, pwd: guest). Default value: false.
 Other:
 -u, --usersFile [string]             File containing list of users. Default value: /Users/huson/Library/Prefere
 -bl, --backlog [number]              Set the socket backlog. Default value: 100.
 -pt, --pageTimeout [number]          Number of seconds to keep pending pages alive. Default value: 10000.
 -rpp, --readsPerPage [number]        Number of reads per page to serve. Default value: 100.
 -t, --threads [number]               Number of threads. Default value: 8.
 -d, --debug                          Debug mode. Default value: false.
 -v, --verbose                        Echo commandline options and be verbose. Default value: false.
 -h, --help                           Show program usage and quit.
```

[The following is *Ultimate Edition Only.*]

10.19 Megan Server (Ultimate Edition)

The `megan-server` commandline program for UE supports multiple end points, multiple roles for users, and data download via MEGAN UE.

```
SYNOPSIS
        MeganServer [options]
DESCRIPTION
        Serves MEGAN files over the web via HTTP
OPTIONS
 Command serve | users | help
        serve                   run as server
```

```
       users              add users
       help              show help
 Input (serve command)
       -i, --input [string(s)]              Input directories. Mandatory option.
       -r, --recurse                        Recursively visit all input subdirectories. Default
       ↪  value: true.
       -x, --extensions [string(s)]         Input file extensions. Default value(s): '.daa'
       ↪  '.rma' '.rma6' '.megan' '.megan.gz'.
 Server (serve command)
       -e, --endpoints [string(s)]          Endpoint names (one per input directory), first has
       ↪  role-free access. Default value(s): 'megan7server'.
       -p, --port [number]                  Server port. Default value: 8001.
       -g, --allowGuest                     Allow guest login (name: guest, pwd: guest).
       ↪  Default value: false.
 Options (users command)
       -a, --allowReplace                   Replace existing users. Default value: false.
 Other:
       -u, --usersFile [string]             File containing list of users. Default value:
       ↪  MeganServerUsers.def.
       -t, --threads [number]               Number of threads. Default value: 8.
       -d, --debug                          Debug mode. Default value: false.
       -v, --verbose                        Echo commandline options and be verbose. Default
       ↪  value: false.
       -h, --help                           Show program usage and quit.
```

10.20 Setup License

The `setup-license` commandline program.

Use this to setup the MEGAN UE license on a server. (UE only)

```
SYNOPSIS
       SetupLicense [options]
DESCRIPTION
       Setup MEGAN UE license on a server
OPTIONS
       -rc, --registrationCode [string]     Registration code (a long string of digits and
       ↪  letters).
       -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
 Other:
       -v, --verbose                        Echo commandline options and be verbose. Default
       ↪  value: false.
       -h, --help                           Show program usage and quit.
```

10.21 Column Join

The `column-join` commandline program.

Ultimate Edition only.

This is used to join tables on a common column.

```
SYNOPSIS
       column-join [options]
DESCRIPTION
       ColumnJoin: join two mapping files via common column
OPTIONS
       -a, --inputA [string]                First input mapping file. Mandatory option.
       -b, --inputB [string]                Second input mapping file. Mandatory option.
       -o, --output [string]                Output mapping file. Mandatory option.
       -ka, --keyColumnA [number]           Key column for first input file (1-based). Default
       ↪  value: 1.
```

```
       -va, --valueColumnsA [string(s)]     Value columns for first input file (1-based, if
       ↪  empty: report all non-key columns).
       -kb, --keyColumnB [number]           Key column for second input file (1-based). Default
       ↪  value: 1.
       -vb, --valueColumnsB [string(s)]     Value column for second input file (1-based, if
       ↪  empty: report all non-key columns).
       -of, --outputFormat [string]         Output format. Default value: valueA_valueB. Legal
       ↪  values: valueA_valueB, key_valueA, key_valueB
       -n, --null                           Report missing values as NULL. Default value: true.
       -s, --sort                           Requires sorting, as input files are not both
       ↪  sorted by key. Default value: false.
       -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
 Other:
       -v, --verbose                        Echo commandline options and be verbose. Default
       ↪  value: false.
       -h, --help                           Show program usage and quit.
```

10.22 Extract From NR

The `extract-from-nr` commandline program.

Ultimate Edition only.

This is used to extract a nr90 or n50 database from the full NCBI-nr database.

```
SYNOPSIS
       ExtractFromNR [options]
DESCRIPTION
       Extracts a clustered database from nr
OPTIONS
 Input and Output:
       -i, --input [string]                 Input file (usually nr.gz). Mandatory option.
       -a, --accessionList [string]         List of accessions to be extracted (obtained from
       ↪  e.g. megan-map-nr50-Feb2024.mdb). Mandatory option.
       -o, --output [string]                Output file. Default value: stdout.
 Options:
       -ap, --accessionPrefix [string]      Database accession pattern (e.g. NCBInr50_).
       ↪  Mandatory option.
       -u, --upper                          Convert all sequence letters to upper case. Default
       ↪  value: true.
 Other:
       -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
       -v, --verbose                        Echo commandline options and be verbose. Default
       ↪  value: false.
       -h, --help                           Show program usage and quit.
```

10.23 Fasta Extract By Hash

The `fasta-extract-by-hash` commandline program.

Ultimate Edition only.

This is used to extract FastA records from a set of files based on sequence hashes.

```
SYNOPSIS
       fasta-extract-by-hash [options]
DESCRIPTION
       FastaExtractByHash: Extracts sequences from a Fasta file using hashes
OPTIONS
 Input and Output
       -i, --input [string(s)]              FastA input file(s) (or directory). Mandatory
       ↪  option.
```

```
       -is, --inputSuffix [string]          Input file suffix (if directory name given).
       -a, --accHashFile [string]           Input accession-to-hash file. Mandatory option.
       -o, --output [string]                Output file. Default value: stdout.
 Options
       -p, --prefix [string]                Prefix for accession label.
       -s, --suffix [string]                Suffix for accession label.
       -u, --upper                          Convert all sequence letters to upper case. Default
       ↪  value: true.
       -f, --flip                           Input hash file is flipped, namely
       ↪  hash-to-accession. Default value: false.
       -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
 Other:
       -v, --verbose                        Echo commandline options and be verbose. Default
       ↪  value: false.
       -h, --help                           Show program usage and quit.
```

10.24 Fasta Hash

The `fasta-hash` commandline program.

Ultimate Edition only.

This is used to hash sequences provided in FastA records.

```
SYNOPSIS
       FastaHash [options]
DESCRIPTION
       FastaHash: Computes a sequence to hash mapping
OPTIONS
 Input and Output:
       -i, --input [string(s)]              Input file(s) (or directory). Mandatory option.
       -is, --suffix [string]               Input file suffix (if directory name given).
       -u, --upper                          Convert all sequence letters to upper case. Default
       ↪  value: true.
       -o, --output [string]                Output file. Default value: stdout.
 Options:
       -mf, --first                         Map first word in header lines to hashes. Default
       ↪  value: true.
       -mo, --other                         Map all words following an ASCII SOH character
       ↪  (code 1). Default value: false.
       -pa, --parseTags                     Map tag-defined accessions to hashes. Default
       ↪  value: false.
       -pt, --tags [string(s)]              Tags to parse. Default value(s): 'ref|' 'gb|'.
       -mh, --mapHeader                     Map header lines to hashes (supersedes above
       ↪  options). Default value: false.
       -ms, --mapSequence                   Map sequences line to hashes (supersedes above
       ↪  options). Default value: false.
       -f, --flip                           Report hash first, then key. Default value: false.
 Taxonomy:
       -tx, --taxonomy                      Extract taxon id from header line and report.
       ↪  Default value: false.
       -tp, --taxonomyPattern [string]      A regular expression for finding taxon ids. Default
       ↪  value: TaxID=(\d+).
 Other:
       -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
       -v, --verbose                        Echo commandline options and be verbose. Default
       ↪  value: false.
       -h, --help                           Show program usage and quit.
```

10.25 Make Acc to Kegg

The `make-acc2kegg` commandline program.

Ultimate Edition only.

This is used to compute an sequence accession to KEGG orthology groups mapping file.

```
SYNOPSIS
        make-acc2kegg [options]
DESCRIPTION
        Creates an Accession-to-KEGG-KO mapping file
OPTIONS
 Input
        -gpi, --inputGenesProteinId [string]   Genes to Id mapping (usually
        ↪  kegg/genes/links/genes_ncbi-proteinid.list.gz). Mandatory option.
        -gko, --inputGenesKO [string]        Genes to KO mapping (usually
        ↪  kegg/genes/links/genes_ko.list.gz). Mandatory option.
 Output
        -o, --output [string]                Name of output file. Default value:
        ↪  acc2kegg.tab.gz.
 Options:
        -ue, --onlyUE                        Format for use in UE. Default value: true.
        -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
 Other:
        -v, --verbose                        Echo commandline options and be verbose. Default
        ↪  value: false.
        -h, --help                           Show program usage and quit.
```

10.26 Make Kegg Tree

The `make-kegg-tree` commandline program.

Ultimate Edition only.

This creates the KEGG tree and map file from data downloaded from KEGG ftp.

```
SYNOPSIS
        make-kegg-tree [options]
DESCRIPTION
        Make a new KEGG tree for use with MEGAN
OPTIONS
 Input
        -d, --inputDir [string]              Input directory (usually kegg/brite/ko, downloaded
        ↪  from ftp://ftp.bioinformatics.jp). Mandatory option.
        -omf, --oldMap [string]              Old mapping file (usually kegg.map-old, extracted
        ↪  from megan/data.jar).
        -ig, --ignore [string(s)]            koXXXXX.keg files to ignore when constructing tree
        ↪  and map.
 Output
        -t, --tree [string]                  Output tree file. Default value: kegg.tre.
        -m, --map [string]                   Output map file. Default value: kegg.map.
 Options
        -ud, --uniformDepth                  Ensure that all leaves have same distance to root.
        ↪  Default value: false.
        -auc, --addUnclassified              Add unclassified KO numbers. Default value: true.
        -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
 Other:
        -v, --verbose                        Echo commandline options and be verbose. Default
        ↪  value: false.
        -h, --help                           Show program usage and quit.
```

10.27 Merge Mappings

The `merge-mappings` commandline program.

Ultimate Edition only.

This is used to merge multiple mapping tables in to a single table in preparation of creating a mapping database for MEGAN.

```
SYNOPSIS
        merge-mappings [options]
DESCRIPTION
        Write out the mapping table to be imported into the mapping-DB
OPTIONS
 Input
        -c, --classifications [string(s)]    List of names of classifications. Mandatory option.
        -i, --input [string(s)]              List of input .tab files for classifications.
        ↪  Mandatory option.
        -inf, --infoFiles [string(s)]        List of input .info files for classifications.
 Output
        -o, --output [string]                Output DB file. Default value: tab.gz.
        -s, --separator [string]             Separator. Default value: tab. Legal values: tab,
        ↪  ;, ,, |
 Options
        -supp, --supportedOnly               Only allow classification names supported by MEGAN.
        ↪  Default value: true.
 Other:
        -t, --threads [number]               Number of threads. Default value: 8.
        -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
        -v, --verbose                        Echo commandline options and be verbose. Default
        ↪  value: false.
        -h, --help                           Show program usage and quit.
```

10.28 Merge Multiple Accession Assignments

The `merge-multiple-accession-assignments` commandline program.

Ultimate Edition only.

This is used to merge multiple accession asignments from cluster members to single assignments for the cluster representation.

```
SYNOPSIS
        MergeMultipleAccessionAssignments [options]
DESCRIPTION
        Merge multiple accession assignments
OPTIONS
        -i, --in [string]                    Input file, each line containing a cluster
        ↪  accession followed  member accessions (stdin, .gz ok). Mandatory option.
        -o, --out [string]                   Output file, each line containing first accession
        ↪  and merged assignments (stdout or .gz ok). Default value: stdout.
        -p, --prefix [string]                Output accession prefix (e.g. NCBInr50_).
        -mdb, --mapDB [string]               MEGAN mapping DB (file megan-map.mdb). Mandatory
        ↪  option.
        -c, --classifications [string(s)]    Classifications to assign (ALL or list of names).
        ↪  Default value(s): 'ALL'.
        -ue, --onlyUltimateEdition [string(s)]   Classifications only for Ultimate Edition.
        ↪  Default value(s): 'KEGG'.
 Advanced
        -lpc, --linesPerCall [number]        Lines to process per call. Default value: 100.
        -apc, --accessionsPerQuery [number]   Maximum number of accessions per SQLITE query.
        ↪  Default value: 10000.
```

```
       -ao, --assignedOnly                  Only output asscessions that have at least one
       ↪  assignment. Default value: false.
 Other:
       -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
       -v, --verbose                        Echo commandline options and be verbose. Default
       ↪  value: false.
       -h, --help                           Show program usage and quit.
```

10.29 Taxdump Tree

The `taxdump-tree` commandline program.

Ultimate Edition only.

This is used to create the ncbi.tre and ncbi.map tree and mapping files from an NCBI taxdump file.

```
SYNOPSIS
       taxdump-tree [options]
DESCRIPTION
       computes the NCBI taxonomy files for MEGAN
OPTIONS
       -i, --input [string]                 taxdump file (Zip file, usually from:
       ↪  ftp://ftp.ncbi.nlm.nih.gov/pub/taxonomy/taxdmp.zip). Mandatory option.
       -t, --tree [string]                  Output tree file. Default value: ncbi.tre.
       -P, --propertiesFile [string]        Properties file. Default value: Megan.def.
 Other:
       -v, --verbose                        Echo commandline options and be verbose. Default
       ↪  value: false.
       -h, --help                           Show program usage and quit.
```

11 Advanced Topics

This chapter covers advanced features for experienced users who wish to integrate MEGAN into automated pipelines, work with large datasets, or extend its functionality.

11.1 Command-line Tools

MEGAN includes several command-line utilities for preprocessing and analysis:

daa-meganizer

- Prepares DIAMOND `.daa` files for use in MEGAN by assigning taxonomic and functional classifications.
- Example usage:

  ```
  daa-meganizer -i example.daa -mdb megan-map.db
  ```

- Output is a modified `.daa` file containing MEGAN annotations.

daa2info

- Extracts summary statistics or read lists from an `.daa` file.
- Can be used in scripts to generate reports.

blast2rma

- Imports a BLAST file to MEGAN's internal `.rma6` format.

rma2info

- Extracts summary statistics or read lists from an `.rma6` file.
- Can be used in scripts to generate reports.

11.2 Batch Processing

MEGAN supports automation of large-scale projects:

- Use shell scripts to run `daa-meganizer` on multiple files.
- Combine with other command-line tools (e.g., DIAMOND, seqtk) to form a complete pipeline.
- Store metadata and processing parameters alongside results for reproducibility.

11.3 Scripting and Custom Analysis

MEGAN Ultimate Edition has scripting built-in.

- MEGAN's GUI includes support for scripting certain operations.
- Use scripts to apply filters, export views, or generate charts automatically.
- Useful for reproducible analyses and complex comparisons across dozens of samples.

11.4 Cloud and HPC Integration

MEGAN is compatible with high-performance and cloud environments:

- Run `daa-meganizer` and related tools on HPC clusters or cloud VMs.
- Use shared storage to manage large alignment files.
- MEGAN GUI can be used locally to inspect results after remote processing.

11.5 Custom Classifications and Mapping Files

Advanced users can supply their own classification systems or mapping files:

- Mapping databases for functional classification are in SQLite format.
- Custom taxonomy files (in MEGAN format) allow use of alternate or user-defined hierarchies.
- Mappings from accessions or gene IDs to functional categories can be created with proper formatting.

11.6 Extending MEGAN

- While MEGAN does not currently support third-party plugins, experienced developers can request access to APIs or libraries for integration with other software.
- Feedback and suggestions for new advanced features are welcome via the project website.

These advanced features enable MEGAN to scale from desktop analysis to automated workflows and high-throughput environments, offering great flexibility for research at any scale.

12 File Formats

MEGAN supports a variety of file formats for input, output, and internal processing. This chapter describes the main formats you are likely to encounter when using MEGAN.

12.1 Input File Formats

Alignment Files

MEGAN accepts alignment files from external tools, particularly:

- `.daa` – Binary alignment format produced by DIAMOND.
- `.blast` – Different flavors of BLAST output.
- `.sam` – Can be used if preprocessed appropriately.

Metadata can be imported in a tab-separated format.

Functional Mapping Files

- `megan.mdb` – SQLite databases containing mappings from reference accessions to taxonomic (NCBI and GTDB) and functional categories (e.g., SEED, eggNOG and KEGG (Ultimate Edition)).

12.2 Internal File Formats

.rma6 Files

- MEGAN's internal binary format for storing reads, classifications, and metadata.
- Generated by the GUI or via `daa2rma`.
- Efficient for reloading large datasets.

.megan Files

- MEGAN files are lightweight files that contain only the taxonomic and function counts, for one or multiple samples, together with metatadata.

12.3 Output File Formats

MEGAN can export a variety of results and visualizations:

Text-Based Tables

- `.txt`, `.csv`, `.tsv` – Read count tables, taxonomic summaries, functional profiles, more than 20 different combinations.

Graphics

- `.png`, `.svg`, `.pdf` – Exported visualizations of trees, charts, and plots.

Read Lists and Trees

- `.fasta`, `.fastq` – Selected reads can be exported in sequence format.
- `.tre`, `.nwk` – Tree exports in Newick format.
- `.txt` - Alignments can be exported in BLAST text format.

12.4 File Compatibility and Tips

- Always use the latest version of DIAMOND to generate `.daa` files for maximum compatibility.
- Mapping databases should be kept up to date to reflect current functional annotations.
- If alignment files are too large, consider splitting them or running on HPC and importing processed results.

Understanding these file formats is crucial for integrating MEGAN into automated workflows and for managing large projects with multiple samples and classifications.

13 Custom Classifications

MEGAN allows users to define and use their own custom classification systems, enabling the analysis of metagenomic reads in the context of specialized or user-defined taxonomies and functional hierarchies. This feature is particularly useful in cases where the standard classifications (e.g., NCBI taxonomy, SEED and eggNOG , or KEGG) are insufficient or not applicable.

Requirements

To add a custom classification to MEGAN, the user must supply the following files:

1. **Hierarchy file (Newick format):** A rooted tree described using the Newick format, where **all nodes (including internal nodes)** are labeled with unique **integer IDs**. This tree defines the hierarchical structure of the classification.
2. **Label mapping file (TSV):** A tab-separated file that maps each integer node ID to a human-readable name. Format:

   ```
   1    Root
   2    Category A
   3    Subcategory A1
   ...
   ```

3. **Optional: Accession mapping file (TSV):**

   A tab-separated file mapping reference sequence accession identifiers (e.g., NCBI or UniProt accessions) to the integer node IDs used in the Newick tree. This allows MEGAN to assign reads to the custom classification based on alignment results. Format:

   ```
   P12345    5
   Q67890    8
   ...
   ```

If you want to use the custom classification with existing ones, you will need to modify the corresponding mapping file by adding another column to the mappings table that indicates the mapping of accession keys to classification ids.

Using the Custom Classification in MEGAN

To use a custom classification:

- Place the Newick tree file, label mapping file, and (if available) accession mapping file in the same directory.
- In the MEGAN GUI, select `Edit > Preferences > Add Classification...` and add a new classification by specifying the file paths.

- Restart the program and the classification should be available when processing BLAST or DAA files.

Considerations

- Integer node IDs must be unique across the tree and consistent across all files.
- The Newick tree must be syntactically correct and rooted.
- The optional accession mapping file should use the same accession format as the reference database used for alignment.

With this feature, MEGAN becomes a flexible platform for classification systems beyond standard taxonomies, supporting custom pathways, gene ontologies, or domain-specific ontologies defined by the user.

14 Troubleshooting and FAQ

This chapter provides solutions to common problems encountered while using MEGAN, along with answers to frequently asked questions.

14.1 Installation Issues

MEGAN won't launch

- Ensure that your system meets the minimum requirements.
- If using the standalone version, confirm that Java 11 or later is available.
- On macOS, allow MEGAN to run via `System Preferences > Security & Privacy`.

Permission denied on Linux

- Make the binary executable using: `chmod +x MEGAN`
- Run MEGAN from the terminal to capture any error messages.

14.2 Data Import Problems

Error loading .daa file

- Ensure the file has been meganized using `daa-meganizer`.
- Confirm the mapping database (`.mdb`) is correct and up to date.

Taxonomy tree is empty or incomplete

- Check LCA parameters (min bit score, top percent).
- Ensure that taxonomic mapping is enabled during import.
- Make sure you are using a supported taxonomy database.

14.3 Performance Issues

MEGAN is slow or unresponsive

- Close unused windows or samples to free memory.
- Increase heap size by editing the file `Megan.vmoptions` and set the value to 20GB, say, using `-Xmx20G`.

Out of memory error

- Ensure sufficient RAM is available.
- Process large datasets on HPC or in batch mode using command-line tools.

14.4 General Usage Questions

Can I open multiple samples in MEGAN?

- Yes. Simply import or open additional files.
- To view and analyze multiple samples or files together, open them using the Compare dialog.

How do I export read lists or images?

- Right-click on any tree node or chart element to access export options.
- Use `File > Export Image` or `File > Export Analysis` for charts and summary tables.

Can I update or replace the taxonomy or mapping databases?

- Yes. Download new mapping databases from the MEGAN website and place them in your working directory or specify them during import.

14.5 Getting Help

If you encounter problems that are not addressed in this manual, consider the following support options:

- Visit the MEGAN website: https://software-ab.cs.uni-tuebingen.de/download/megan7
- Consult the online user forum https://megan.cs.uni-tuebingen.de.
- Contact the developers via the support email listed on the website.

Most issues in MEGAN can be resolved by checking input files, adjusting settings, and ensuring that the most recent versions of supporting tools and databases are used.

15 Scripting (Ultimate Edition)

[The following is *Ultimate Edition Only*.]

15.1 Command-Line Options

The Ultimate Edition (UE) of MEGAN allows to run the program in command-line mode.

MEGAN UE has the following command-line options:

```
Mode:
 -g, --commandLineMode              Run MEGAN in command-line mode. Default value: false.

 Input:
 -f, --files [string(s)]            MEGAN file(s) to open.

 Commands:
 -x, --execute [string]             Command to execute at startup
                                    (do not use for multiple commands).
 -c, --commandFile [string]         File of commands to execute in command-line mode.

 Configuration:
 -E, --quitOnException              Quit if exception thrown in command-line mode.
                                    Default value: false.
 -p, --propertiesFile [string]      Properties file. Default value: Megan.def.
 +w, --hideMessageWindow            Hide message window. Default value: false.
 -V, --version                      Show version string. Default value: false.
 -S, --silentMode                   Silent mode. Default value: false.
 -d, --debug                        Debug mode. Default value: false.
 +s, --hideSplash                   Hide startup splash screen. Default value: false.
 -rc, --registrationCode [string]   Enter registration code.

 Other:
 -v, --verbose                      Echo commandline options and be verbose.
                                    Default value: false.
 -h, --help                         Show program usage and quit.
```

When running in command-line mode, the program will first executing any command given with the `-x` option and then will read commands from the file specified using the `-c` command. If no such file is given, additional commands are read from standard input.

Please note that windows will still open when in command-line mode, but should not be used interactively. (This is necessary for the program to fully implement all graphical commands.) To prevent windows from opening, or to use the command-line mode on a server, please use the linux virtual frame buffer command xvfb-run, as shown here:

```
xvfb-run --auto-servernum --server-num=1 MEGAN +g
```

New features of MEGAN6 are implemented using JavaFX. Running MEGAN6 on a Linux server requires that the graphics toolkit GTK2.18 (or later) is installed, e.g. using the following command: `apt-get install xvfb libgtk2.0-0`

(There may be problems using `libgtk-3-0`)

Please be aware that the command-line version of the program uses the same properties file as the interactive version. So, any preferences set using the interactive version of the program will also apply to the command-line version of the program. It this is not desired, then please use the `-p` option to supply a different properties file.

Another important thing to note is that the command-parser operates in a line-by-line fashion. When processing commands in a given line, the parser makes note of required updates to the taxonomy and data-structures. These updates are not executed until all commands in the current input line have been processed. For example, if you want to open and MEGAN file and then to save a picture of the taxonomical analysis in a PDF file, then the two commands should be entered on separate lines because otherwise the taxonomy will be drawn before the data from the MEGAN file has been processed. Here is an example of the correct way to produce a picture of a taxonomic analysis:

```
open file='/Users/huson/data/megan/x.rma'
export image file='/Users/huson/data/megan/x.pdf' format=PDF replace=true
quit
```

Alternatively, the update command is used to explicitly force MEGAN to update all data-structures, in this case the commands show appear together on one line, e.g.:

```
open file='x.rma';update;exportimage file='x.pdf'format=PDF replace=true;
```

As described below, the update command takes a number of different parameters that is used to determine exactly what type of update is required.

Please use the `-x` option only to specify a single command, as updating may otherwise not work correctly.

One example of using MEGAN command-line mode is given in Section 15.2.1. Other examples and recipes for command-line scripts performing common use cases of MEGAN are available on the MEGAN community website.

15.2 Command-Line Commands

Each type of window that can be opened by MEGAN has its own command interpreter. Initially, on startup the program will open a Main window and all commands piped to the program will be executed using the command interpreter associated with the main window. The main window provides a number of commands for opening other windows. For example, the command `open viewer=SeedViewer;` will open the SEED classification viewer. To pipe commands to the SEED viewer, the command context has to be set to the SEED viewer, by entering `set context=seedviewer;`. After entering this command, all subsequent commands are handled by the interpreter associated with the SEED viewer. To obtain a list of all commands available for the current interpreter, enter `help;`. In obtain help on a particular command, for example on *export*, enter `help export;`. All command description lines that contain the word "export" (case insensitive) will be listed.

In the following we list all commands available in the Main viewer. Other viewers support many of these commands, too, but also other, viewer-specific ones. To determine which commands are available for a given window, run MEGAN in GUI mode, open the window of interest and then

select the `Window > Command Syntax...` item to obtain a listing of all commands available for the given window. Here are the commands that are available in the Main viewer:

```
Available commands (context=MainViewer):

File menu:
new; - Open a new empty document
open file=<filename> [readOnly={false|true}]; - Open a MEGAN file (ending on .rma, .meg or .megan)
show window=RemoteBrowser; - Open browser for remote files
show window=ImportBlast; - Show the 'Import from Blast' dialog
save file=<filename> [summary={true|false}]; - Save current data set
exportImage file=<filename> [descriptionFile=<filename>] [format={bmp|eps|gif|jpg|pdf|png|svg}] [replace={false|true}]
 [visibleOnly]={false|true}] [textAsShapes={false|true}] [title=<string>];
 - Export content of window to an image file
exportLegend file=<filename> [format={bmp|eps|gif|jpg|pdf|png|svg}] [replace={false|true}] [textAsShapes={false|true}];
 - Export content of legend window
show window=pagesetup; - Setup the page for printing
show window=print; - Print the main panel
extract what=document file=<megan-filename> [data={Taxonomy|INTERPRO2GO|EGGNOG|SEED|KEGG|...}]
 [ids=<SELECTED|numbers...>] [allBelow={false|true}];
 - Extract all reads and matches for all selected nodes to a new document
show window=ExtractReads; - Extract reads for the selected nodes
show window=properties; - Show document properties
close; - Close the window

Import sub-menu:
import csv={reads|summary} separator={comma|tab} file=<fileName> fNames={Taxonomy|INTERPRO2GO|EGGNOG|SEED|KEGG|...} [topPercent=<num>] [minScore=<num>] [minSupportPe
 - Load data in CSV (comma- or tab-separated value) format: READ_NAME,CLASS-NAME,SCORE or CLASS,COUNT(,COUNT...)
import format=biom file=<fileName>;
 - Import data from a table in BIOM 1.0 format (see http://biom-format.org/documentation/format_versions/biom-1.0.html)
import metaData=<file> [format={metaDataMapping}];
 - Import a metadata mapping file (as defined in http://qiime.org/documentation/file_formats.html)

Export sub-menu:
export what=CSV format={format} [separator={comma|tab}] [counts={assigned|summarized}] file=<filename>;
 - Export assignments of reads to nodes to a CSV (comma or tab-separated value) file
export format=biom data={{Taxonomy|INTERPRO2GO|EGGNOG|SEED|KEGG|...} file=<filename>;
 - Export data in BIOM 1.0 format (see http://biom-format.org/documentation/format_versions/biom-1.0.html)
export metaData=<file> [format={metaDataMapping}];
 - Export a metadata mapping file (as defined in http://qiime.org/documentation/file_formats.html)
export what=paths file=<filename>; - Export assignments of reads weighted taxonomic paths
export what=tree file=<filename> [simplify={false|true}] [showInternalLabels={true|false}] [showUassigned={true|false}];
 - Export induced tree (in Newick format)
export what=reads [data={{Taxonomy|INTERPRO2GO|EGGNOG|SEED|KEGG|...}] file=<filename>;
 - Export all reads to a text file (or only those for selected nodes, if any selected)
export what=matches [data={{Taxonomy|INTERPRO2GO|EGGNOG|SEED|KEGG|...}] file=<filename>;
 - Export all matches to a text file (or only those for selected nodes, if any selected)
export what=alignment file=<filename> data={{Taxonomy|INTERPRO2GO|EGGNOG|SEED|KEGG|...} classId={number[,number...]|selected} [asConsensus={false|true}]  [asContigs=
 [useEachReadOnlyOnce={true|false}] [useEachReferenceOnlyOnce={true|false}] [includeInsertions={true|false}]
 [refSeqOnly={false|true}] [contractGaps={false|true}] [translateCDNA={false|true}] [minReads={number}] [minLength={number}] [minAvCoverage={number}];
 - Calculate and export alignments for all selected leaves
export assembly file=<name> [minOverlap=<number>] [minReads=<number>] [minLength=<number>] [minAvCoverage=<number>] [maxPercentIdentity=<number>] [showGraph={false|t
 - Compute and export "gene-centric" assembly of reads for all selected nodes

Edit menu:
copyLegend; - Copy legend image to clipboard
set  description=<text>; - Edit or show the description of the data
show window=formatter; - Format nodes and edges
show findToolbar={true|false}; - Open the find toolbar

Preferences sub-menu:

Fix Taxon Mapping sub-menu:
changeMapping taxName=<taxon-name> taxId=<taxon-id>;
 - Change the taxon name to taxon id mapping for a given taxon
changemapping list; - List all changes
changemapping clear; - Clear all changes

Taxon Disabling sub-menu:
enable taxa=all; - Enable all taxa
disable taxa={selected|<name,...>}; - Disable all selected taxa or all named ones
enable taxa={selected|all|<name,...>}; - Enable all selected taxa or all named ones
list taxa=disabled; - List all disabled taxa


Select menu:
select nodes={all|none|leaves|internal|previous|subtree|leavesBelow|nodesAbove|intermediate|invert}
 - Select nodes
select nodes=previous; - Select from previous window

Taxonomic Rank sub-menu:

Options menu:
recompute [minSupportPercent=<number>] [minSupport=<number>] [minScore=<number>] [maxExpected=<number>] [topPercent=<number>] [weightedLCA={false|true}] [lcaCoverage
 [useMinimalCoverageHeuristic={false|true}] [longReads={false|new}] [pairedReads={false|true}] [useIdentityFilter={false|true}]
 [fNames={COG|KEGG|PFAM|SEED]; - Rerun the LCA analysis with different parameters
set totalReads=<num>;
 - Set the total number of reads in the analysis (will initiate recalculation of all classifications)
project rank={Domain|Kingdom|Phylum|Class|Order|Family|Genus|Species} [minPercent={number}];
 - Projects all taxonomic assignments onto a given rank
list summary nodes={all|selected} [outFile=<name>];
 - List summarized counts for nodes selected of tree
list paths nodes=selected [outFile=<name>]; - List path from root to node for all selected
compute index={Shannon|SimpsonReciprocal} [data={Taxonomy|SEED|KEGG}];
 - Compute the Shannon-Weaver diversity index
compare mode={ABSOLUTE|RELATIVE} [ignoreUnassigned={false|true}] [pid=<number> ...] [meganFile=<filename> ...];
 - Open compare dialog to produce a comparison of multiple datasets
show webpage taxon=<name|id>; - Open NCBI Taxonomy web site in browser
```

```
inspector taxa=selected; - Inspect the read-to-taxon assignments

Layout menu:
show legend={horizontal|vertical|none}; - Show horizontal or vertical legend, or hide
set fontSize={<number>|increase|decrease}; - Set the font size
set autoLayoutLabels={true|false}; - Layout labels
set scaleBy={Summarized|Assigned|None}; - Scale nodes by number of reads assigned
set maxNodeHeight=<number>; - Set the maximum node height in pixels
zoom what=selected; - Zoom to the selection
zoom what=fit; - Contract tree vertically
zoom what=full; - Expand tree vertically
set nodeDrawer={Summarized|Assigned|None}; - Draw data as pie charts
set scale={linear|percent|log}; - Show values on a linear scale
set magnifier={true|false}; - Turn the magnifier on or off
set drawLeavesOnly={true|false}; - Only draw leaves

Expand/Contract sub-menu:
expand direction={horizontal|vertical}; - Expand canvas horizontally
contract direction={horizontal|vertical}; - Contract view horizontally

Tree menu:
collapse nodes={SELECTED|name [name name ...]}; - Collapse nodes
collapse level=<num>; - Collapse all nodes at given depth in tree
collapse except={id...}; - Collapse all parts of tree that are not above or below the selected nodes
uncollapse nodes={all|selected|<name ...>} [subtree={false|true}]; - Uncollapse selected nodes
hide minSupport=<number>; - Hide all nodes that have low support
nodeLabels [names=<bool>] [ids=<bool>] [assigned=<bool>] [summarized=<bool>];
 - Determine what to label nodes with
show labels=selected; - Show labels for selected nodes
hide labels=selected; - Hide labels for selected nodes
show intermediate=<bool>; - Show intermediate labels at nodes of degree 2

Collapse At Taxonomic Rank sub-menu:

Window menu:
register licenseKey=<string>; - Register a license key
show window=howToCite; - Show how to cite the program
show window=message; - Open the message window
reset windowLocation; - Reset the location of a window
set windowSize=<width> x <height>; - Set the window size
show window=inspector; - Open inspector window
show window=aligner; - Show alignment of reads to a specified reference sequence
show window=mainViewer; - Brings the main viewer to the front
open viewer=EGGNOG; - Open eggNOG viewer
open viewer=SEED; - Open SEED viewer
open viewer=KEGG; - Open KEGG viewer
show window=sampleViewer; - Opens the Sample Viewer
show window=timeSeriesViewer; - Opens the Time Series Viewer
show window=groups; - Show groups viewer
show chart drawer={BarChart,BricksChart,BubbleChart,CoOccurrencePlot,HeatMap,LineChart,NormalizedBarChart,PieChart,Plot2D,RadialTreeChart,StackedBarChart,
   StackedLineChart,WordCloud} data={{Taxonomy|INTERPRO2GO|EGGNOG|SEED|KEGG|...|attributes}; - Show chart
show comparisonPlot [data={{Taxonomy|INTERPRO2GO|EGGNOG|SEED|KEGG|...]};
 - Plot pairwise comparison of assignments to classes
show window=clusterViewer; - Open a cluster analysis window
show rarefaction data={{Taxonomy|INTERPRO2GO|EGGNOG|SEED|KEGG|...};
 - Compute and chart a rarefaction curve based on the leaves of the tree shown in the viewer
help [keyword(s)]; - Show syntax of commands for current viewer

Additional commands:
addSample [sample=<name>] source=<filename|pid> ... [overwrite={false|true}];
 - Add samples from other documents
addServer url=<url> [user=<user>] [password=<password>];
 - Add a MEGAN server to the persistent list of known servers
apply majorityVote voteConfidence=<\%ofConfidence>
 - Apply a the majority vote filter. Reads with a defined percentage of matches assigned to one taxon will bypass the LCA and assign the read to this taxon.
collapse rank={SuperKingdom|Kingdom|Phylum|Class|Order|Family|Varietas|Genus|Species_group|Species|Subspecies}
 - Collapse tree at specific taxonomic rank
compare title=<string> name=<string> samples=<string ...> [name=<string> samples=<string ...>] ... [comparisonMode={ABSOLUTE|RELATIVE}]
         [ignoreUnassigned={false|true}] ; - Compare some groups of samples
compute profile={Projection|ReadSpreading} rank={Domain|Kingdom|Phylum|Class|Order|Family|Genus|Species [minPercent=number]};
 - Computes a taxonomic profile by projecting all counts on to a given rank
export overlapGraph file=<name> [minOverlap=<number>] [showGraph={false|true}];
 - Build and export the overlap graph for selected nodes
export readname2taxpath file=<file>; - Export readname to taxonomic path for all reads
export selected path file=<file>; - Export select Path
export taxonname_count separator={comma|tab} folder=<foldername> - Export assignments
export what=matchPatterns taxon=<id or name> rank=<name> file=<filename>;
 - Export all match signatures for the select node
extract samples=<name1 name2 ...>; - Extract samples to a new document
extract what=reads outDir=<directory> outFile=<filename-template> [data={{Taxonomy|INTERPRO2GO|EGGNOG|SEED|KEGG|...}][ids=<SELECTED|numbers...>]
 [names=<names...>] [allBelow={false|true}]; - Extract reads for the selected nodes
help [keyword(s)]; - Show syntax of commands for current viewer
import blastFile=<name> [,<name>...] [fastaFile=<name> [,<name>...]] meganFile=<name>  format={BlastText|BlastXML|BlastTab|RapSearch2Aln|RDPAssignmentDetails|
    RDPStandalone|Mothur|SAM|References_as_FastA}
 mode={BlastN|BlastP|BlastX|Classifier} [maxMatches=<num>] [minScore=<num>] [maxExpected=<num>]
 [topPercent=<num>] [minSupportPercent=<num>] [minSupport=<num>] [lcaAlgorithm={Naive|Weighted|NaiveLongReads}] [lcaCoveragePercent=<num>]
    [minComplexity=<num>] [useIdentityFilter={false|true}]
 [fNames={{Taxonomy|INTERPRO2GO|EGGNOG|SEED|KEGG|......} [paired={false|true} [pairSuffixLength={number}]]
 [description=<text>];
 - Import BLAST (or RDP or Silva or SAM) and reads files to create a new MEGAN file
list assigned nodes={all|selected} [outFile=<name>]; - List assigned counts for selected nodes of tree
list assignmentsToLevels [outFile=<name>];
 - List the number of reads assigned to each level of the taxonomy
listServers; - List all added servers
load colorFile=<filename>; - Load a palette of colors from a file (one RGB color per line)
load mapFile=<filename> mapType=<mapType> cName=<name> [parseTaxonNames={false|true}] ; - Loads a mapping file
load taxonomyFile=<filename> [mapFile=<filename>]; - Load taxonomy.tre and taxonomy.map files
mpAnalyzer what={lca-ranks|compare} infile=<filename> outfile=<filename>;
 - Compute the rank at which the LCA is found for each mate-pair, or preprocess comparison
```

```
open viewer=KEGG; - Open KEGG viewer
open viewer=Taxonomy; - Open Taxonomy viewer
queryServer url=<url> query={countFiles|listFiles}; - Query a known server
quit; - Quit the program
remoteServer url=<url>; - Add a MEGAN server to the persistent list of known servers
scrollTo node=<name>; - Scroll to a specific node
select id=<number> ...; - Select the nodes for the given ids
select name=<name> <name> ... [state={true|false}]; - Select the named nodes
select rank={SuperKingdom|Kingdom|Phylum|Class|Order|Family|Varietas|Genus|Species_group|Species|Subspecies}
 - Select nodes by rank
set color={<color>|null}; - Set the color of selected nodes and edges
set context={<window-type>|?};
 - Choose command context, i.e. the window that will parse subsequent commands. Use ? to list current context and all available contexts.
set dir=<directory> - Set the current directory
set drawer={RectangularCladogram,RectangularPhylogram,RoundedCladogram,RoundedPhylogram};
 - Set the tree drawer
set edgeShape={angular|straight|curved}; - Set the shape of selected edges
set edgeWidth=<integer>; - Set the width of selected edges
set fillColor={<color>|null} - Set the fill color of selected nodes
set font=<name-style-size>; - Set font nodes or edges, e.g. arial-italic-12
set fullScreen={false|true}; - Full Screen Mode
set groupNodes={none|selected}; - Group selected nodes in PCoA plot
set labelColor={<color>|null}; - Set the label color of selected nodes and edges
set labelFillColor={<color>|null}; - Set the label color of selected nodes and edges
set magnifyallmode={true|false}; - Magnify the whole tree
set margin [left=<number>] [right=<number>] [bottom=<number>] [top=<number>];
 - Set margins used in tree visualization
set nodeShape={none|circle|square|triangle|diamond}; - Set the node shape
set nodeSize=<integer>; - Set the size of selected nodes
set useMagnitude={true|false};
 - Use reads magnitude values to weight reads, if present in their FastA header lines
setProp <name>=<value>; - Set a property
show histogram classId=<num>; - Shows the distribution of matches for a given taxon
show keggTab id=<num,num,...>; - Show the specified KEGG pathway (UE version only, when KEGG license available)
show webpage classification=<name> id=<id>; - Search for selected items in browser
show window=about; - Display the 'about' window
show window=attributes; - Open Microbial Attributes window
show window=comparisonStats; - Open dialog to produce a statistical comparison of two datasets
toFront [file=name]; - Bring window to front
update [reProcess={false|true}] [reset={false|true}] [reInduce={false|true}]; - Update data
use cViewer=<name> state={true|false}; - Determine whether to perform a specific functional analysis
use mapType=<mapType> cName=<name> state=<true|false>; - Set activity state of map type
version; - Show version info
```

15.2.1 Writing Scripts

The best way to run scripts with MEGAN is to prepare a file of commands and then pipe these to MEGAN in command-line mode. Use of the -x option to supply commands is not encouraged because of update issues. MEGAN uses updates all windows etc after a line of commands has been entered and all commands provided using the -x option are considered to be contained in a single line.

Here is an example of how one would use MEGAN in command-line mode on a Mac to save some information on KEGG assignments:

```
/Applications/Megan/MEGAN.app/Contents/MacOS/JavaApplicationStub -g -E < commands.txt
```

where the file commands.txt contains the following lines:

```
open file='/Users/huson/data/megan/microbiome.rma';

open viewer=EggnogViewer;

set context=EggnogViewer;

update;

uncollapse nodes=all;

select nodes=leaves;

export what=CSV format=eggnogName_count separator=tab file='/Users/huson/data/megan/eggnog.txt';

quit;
```

The first line is used to open a MEGAN file. Please surround the file name with single quotes as shown here.
The second line opens the eggNOG window (or EggnogViewer, as it is referred to here).
The third line sets the command context to the EggnogViewer (so that subsequent commands are interpreted by the EggnogViewer). The argument of this command is case sensitive. Please

use `EggnogViewer` and not `eggnogviewer`.
The fourth line ensures that the eggNOG window is uptodate.
The fifth line uncollapses the whole eggNOG tree.
The sixth line selects all leaves of the eggNOG tree.
The seventh line exports all eggNOG names and read counts to a file in "Delimiter separated format".
The eight line quits the program.

More examples for using command scripts with MEGAN are available on the Community website.

A MEGAN Editions and Licensing

MEGAN is available in two editions: the **Community Edition** and the **Ultimate Edition**, each suited to different user needs.

MEGAN Community Edition

The Community Edition of MEGAN is open-source and distributed under the terms of the GNU General Public License (GPL). It provides access to the core functionality of MEGAN, including:

- Taxonomic and functional analysis using public resources such as SEED and eggNOG.
- Graphical user interface (GUI) for interactive exploration.
- Import of DIAMOND, BLAST, and LAST alignment files.
- Support for a variety of visualization and export options.

The Community Edition is free and can be downloaded from the official MEGAN website.

MEGAN Ultimate Edition

The Ultimate Edition of MEGAN includes all the features of the Community Edition, plus additional capabilities tailored to professional and enterprise-level workflows. Notable features include:

- Integrated support for KEGG functional classification (licensed for use in MEGAN).
- Full-featured command-line mode for automated and large-scale workflows.
- Support for commercial and industrial use.

The Ultimate Edition requires a commercial license, which includes the right to use KEGG pathways within MEGAN. Licenses for the Ultimate Edition can be obtained from:

Computomics GmbH
Eisenbahnstraße 1, 72072 Tübingen, Germany
Website: https://computomics.com

Which Edition Should I Use?

- Use the **Community Edition** if you need basic metagenomic analysis tools in a GUI.
- Choose the **Ultimate Edition** if you need KEGG integration, command-line automation, or a commercial partner.

To request a license, please visit the Computomics website.

MEGAN incorporates third-party libraries that may be licensed under separate open-source agreements (e.g., Apache, GPL, LGPL). A full list of third-party components and their respective licenses can be found in the MEGAN installation directory.

Bibliography

Krithika Arumugam, Caner Bagci, Irina Bessarab, Sina Beier, Benjamin Buchfink, Anna Gorska, Guanglei Qiu, Daniel H Huson, and Rohan BH Williams. Annotated bacterial chromosomes from frame-shift-corrected long read metagenomic data. *Microbiome*, 7(61), 2019.

Caner Bağcı, David Bryant, Banu Cetinkaya, and Daniel H. Huson. Microbial phylogenetic context using phylogenetic outlines. *Genome Biology and Evolution*, 13(9):evab213, 2021. doi: 10.1093/gbe/evab213.

D.A. Benson, I. Karsch-Mizrachi, D.J. Lipman, J. Ostell, and D.L. Wheeler. Genbank. *Nucleic Acids Res*, 1(33):D34–38, 2005.

David Bryant and Vincent Moulton. Neighbor-net: An agglomerative method for the construction of phylogenetic networks. *Molecular Biology and Evolution*, 21(2):255–265, 2004. doi: 10.1093/molbev/msh018.

B. Buchfink, C. Xie, and D.H. Huson. Fast and sensitive protein alignment using DIAMOND. *Nature Methods*, 12:59–60, 2015.

Benjamin Buchfink, Haim Ashkenazy, Klaus Reuter, John A. Kennedy, and Hajk-Georg Drost. Sensitive clustering of protein sequences at tree-of-life scale using diamond deepclust. *bioRxiv*, 2023. doi: 10.1101/2023.01.24.525373.

A. Gautam, H. Felderhoff, C. Bagci, and D. H. Huson. Using AnnoTree to get more assignments, faster, in DIAMOND+MEGAN microbiome analysis. *mSystems*, 7, 2021. URL `https://api.semanticscholar.org/CorpusID:244648952`.

J. J. Gillespie, A. R. Wattam, S. A. Cammer, J. L. Gabbard, M. P. Shukla, O. Dalay, T. Driscoll, D. Hix, S. P. Mane, C. Mao, E. K. Nordberg, M. Scott, J. R. Schulman, E. E. Snyder, D. E. Sullivan, C. Wang, A. Warren, K. P. Williams, T. Xue, H. S. Yoo, C. Zhang, Y. Zhang, R. Will, R. W. Kenyon, and B. W. Sobral. PATRIC: the comprehensive bacterial bioinformatics resource with a focus on human pathogenic species. *Infect Immun*, 79(11):4286–4298, November 2011. doi: 10.1128/IAI.00207-11.

J. C. Gower. Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika*, 53(3-4):325–338, 1966. doi: 10.1093/biomet/53.3-4.325.

D. H. Huson, A. F. Auch, J. Qi, and S. C. Schuster. MEGAN analysis of metagenomic data. *Genome Res*, 17(3):377–386, March 2007.

D. H. Huson, S. Mitra, N. Weber, H.-J. Ruscheweyh, and S. C. Schuster. Integrative analysis of environmental sequences using MEGAN 4. *Genome Research*, 21:1552–1560, 2011.

D. H. Huson, R. Tappu, A. L. Bazinet, C. Xie, ¿ P. Cummings, K. Nieselt, and R. Williams. Fast and simple protein-alignment-guided assembly of orthologous gene families from microbiome sequencing reads. *Microbiome*, 5(1):11, 2017.

Daniel H. Huson, Benjamin Albrecht, Caner Bağcı, Irina Bessarab, Anna Górska, Dino Jolic, and Rohan B. H. Williams. MEGAN-LR: new algorithms allow accurate binning and easy interactive exploration of metagenomic long reads and contigs. *Biology Direct*, 13(1):6, Apr 2018.

D.H. Huson, Sina Beier, Isabell Flade, Anna Górska, Mohamed El-Hadidi, Suparna Mitra, Hans-Joachim Ruscheweyh, and Rewati Tappu. MEGAN Community Edition - interactive exploration and analysis of large-scale microbiome sequencing data. *PLoS Comput Biol*, 12 (6):e1004957, 2016.

M. Kanehisa, Y. Sato, M. Furumichi, K. Morishima, and M. Tanabe. New approach for understanding genome variations in kegg. *Nucleic Acids Research*, 47(D1):D590–D595, 2018.

Ross Overbeek, Robert Olson, Gordon D. Pusch, Gary J. Olsen, James J. Davis, Terry Disz, Robert A. Edwards, Svetlana Gerdes, Bruce Parrello, Maulik Shukla, Veronika Vonstein, Alice R. Wattam, Fangfang Xia, and Rick Stevens. The SEED and the rapid annotation of microbial genomes using subsystems technology (RAST). *Nucleic Acids Research*, 2013.

Donovan H. Parks, Maria Chuvochina, Pierre-Alain Chaumeil, Christian Rinke, Aaron J. Mussig, and Philip Hugenholtz. A complete domain-to-species taxonomy for bacteria and archaea. *Nature Biotechnology*, 2020.

Sean Powell, Damian Szklarczyk, Kalliopi Trachana, Alexander Roth, Michael Kuhn, Jean Muller, Roland Arnold, Thomas Rattei, Ivica Letunic, Tobias Doerks, Lars Juhl Jensen, Christian von Mering, and Peer Bork. eggNOG v3.0: orthologous groups covering 1133 organisms at 41 different taxonomic ranges. *Nucleic Acids Research*, 40(Database-Issue):284–289, 2012.

Hans-Joachim Ruscheweyh and Daniel H. Huson. MeganServer - easy interactive access to large-scale metagenome data. Submitted, 2015.

Naruya Saitou and Masatoshi Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4(4):406–425, 1987. ISSN 0737-4038.

Conrad L Schoch, Stacy Ciufo, Mikhail Domrachev, Carol L Hotton, Sivakumar Kannan, Rogneda Khovanskaya, Detlef Leipe, Richard Mcveigh, Kathleen O'Neill, Barbara Robbertse, Shobha Sharma, Vladimir Soussov, John P Sullivan, Lu Sun, Seán Turner, and Ilene Karsch-Mizrachi. Ncbi taxonomy: a comprehensive update on curation, resources and tools. *Database (Oxford)*, 2020, Jan 2020.

P. H. A. Sneath and R. R. Sokal. Numerical taxonomy. *Nature*, 183(4659):548–550, 1957. doi: 10.1038/183548a0.

B. E. Suzek, Y. Wang, H. Huang, P. B. McGarvey, C. H. Wu, and the UniProt Consortium. UniRef clusters: a comprehensive and scalable alternative for improving sequence similarity searches. *Bioinformatics*, 31(6):926–932, 11 2014.

D. Wu, P. Hugenholtz, K. Mavromatis, R. Pukall, E. Dalin, N. N. Ivanova, V. Kunin, L. Goodwin, M. Wu, B. J. Tindall, S. D. Hooper, A. Pati, A. Lykidis, S. Spring, I. J. Anderson, P. D'Haeseleer, A. Zemla, M. Singer, A. Lapidus, M. Nolan, A. Copeland, C. Han, F. Chen, J.-F. Cheng, S. Lucas, C. Kerfeld, E. Lang, S. Gronow, P. Chain, D. Bruce, E. M. Rubin, N. C. Kyrpides, H.-P. Klenk, and J. A. Eisen. A phylogeny-driven Genomic Encyclopaedia of Bacteria and Archaea. *Nature*, 462(7276):1056–1060, Dec 2009.