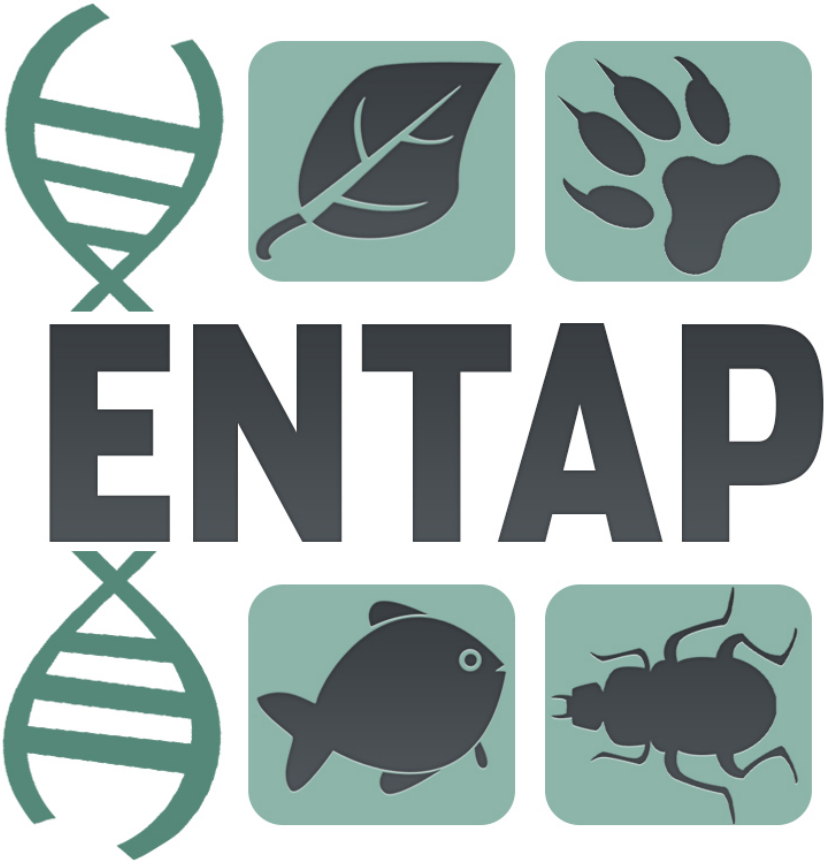
EnTAP Documentation

Release beta/0.10.9

Alex Hart, Jill Wegrzyn

Jul 06, 2023

Getting Started



EnTAP is an eukaryotic non-model annotation pipeline developed by Alexander Hart and Dr. Jill Wegrzyn of the Plant Computational Genomics Lab at the University of Connecticut.

Version 0.10.9-Beta

How to cite: Hart AJ, Ginzburg S, Xu M, et al. EnTAP: Bringing faster and smarter functional annotation to non-model eukaryotic transcriptomes. Mol Ecol Resour. 2020;20:591–604. https://doi.org/10.1111/1755-0998.13106

CHAPTER 1

Introduction

The Eukaryotic Non-Model Transcriptome Annotation Pipeline (*EnTAP*) is designed to improve the accuracy, speed, and flexibility of functional gene annotation for de novo assembled transcriptomes in non-model eukaryotes.

This software package addresses the fragmentation and related assembly issues that result in inflated transcript estimates and poor annotation rates. Following filters applied through assessment of true expression and frame selection, open-source tools are leveraged to functionally annotate the translated proteins.

Downstream features include fast similarity search across multiple databases, protein domain assignment, orthologous gene family assessment, Gene Ontology term assignment, and KEGG pathway annotation.

The final annotation integrates across multiple databases and selects an optimal assignment from a combination of weighted metrics describing similarity search score, taxonomic relationship, and informativeness. Researchers have the option to include additional filters to identify and remove potential contaminants and prepare the transcripts for enrichment analysis. This fully featured pipeline is easy to install, configure, and runs much faster than comparable functional annotation packages. It is developed to contend with many of the issues in existing software solutions.

EnTAP is optimized to generate extensive functional information for the gene space of organisms with limited or poorly characterized genomic resources.

How to cite:

- Hart AJ, Ginzburg S, Xu M, et al. EnTAP: Bringing faster and smarter functional annotation to non-model eukaryotic transcriptomes. Mol Ecol Resour. 2020;20:591–604. https://doi.org/10.1111/1755-0998.13106

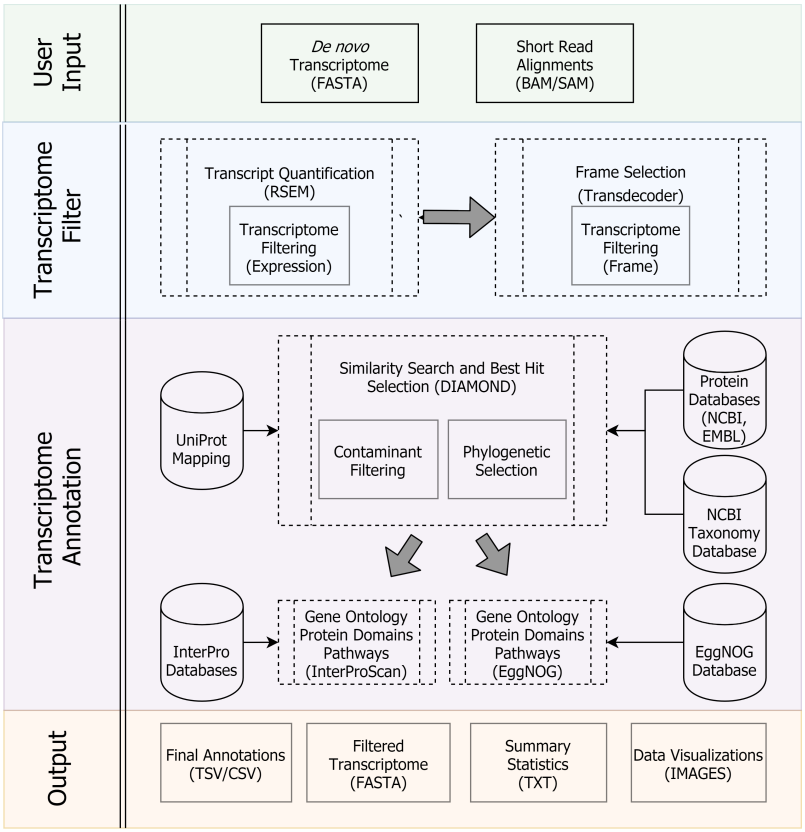
1.1 Pipeline Stages

- **Transcriptome Filtering: designed to remove assembly artifacts and identify true CDS (complete and partial genes)**
    1. Expression Filtering (RSEM)
    2. Frame Selection (TransDecoder by default, or GeneMarkS-T)
- **Transcriptome Annotation: designed to assign functional information to sequences (homology, Gene Ontology, KEGG)**

3. Similarity Search: optimized search against user-selected databases (DIAMOND).
4. Contaminant Filtering and Best Hit Selection: selects final annotation and identifies potential contaminants
5. Orthologous Group Assignment: independent assignment of translated protein sequences to gene families (eggNOG). Includes protein domains (SMART/Pfam), Gene Ontology (GO) terms, and KEGG pathway assignment.
6. InterProScan (optional): sequence search against the families of InterPro databases to assign protein domains, Gene Ontology terms, and pathway information

Note: For information/bug reports, contact Alexander Hart at entap.dev@gmail.com or visit the Slack channel at entap-help.slack.com.

The figure below represents the typical EnTAP pipeline. A more detailed version with all output files can be seen within Interpreting the Results.



1.2 Citations

[1] A. Mitchell, H.-Y. Chang, and L. Daugherty, "The InterPro protein families database: the classification resource after 15 years," Nucleid Acids Research, vol. 43, no. D1, 2015.

[2] B. Buchfink, Xie C., D. Huson, "Fast and sensitive protein alignment using DIAMOND", Nature Methods 12, 59-60 (2015).

[3] eggNOG 4.5: a hierarchical orthology framework with improved functional annotations for eukaryotic, prokaryotic and viral sequences. Jaime Huerta-Cepas, Damian Szklarczyk, Kristoffer Forslund, Helen Cook, Davide Heller, Mathias C. Walter, Thomas Rattei, Daniel R. Mende, Shinichi Sunagawa, Michael Kuhn, Lars Juhl Jensen, Christian von Mering, and Peer Bork. Nucl. Acids Res. (04 January 2016) 44 (D1): D286-D293. doi: 10.1093/nar/gkv1248

[4] G. O. Consortium, "Gene Ontology Consortium: going forward," (in eng), Nucleic Acids Res, vol. 43, no. Database issue, pp. D1049-56, Jan 2015.

[5] J. Huerta-Cepas et al., "Fast genome-wide functional annotation through orthology assignment by eggNOG-mapper," (in eng), Mol Biol Evol, Apr 2017.

[6] M. Kanehisa, M. Furumichi, M. Tanabe, Y. Sato, and K. Morishima, "KEGG: new perspectives on genomes, pathways, diseases and drugs," (in eng), Nucleic Acids Res, vol. 45, no. D1, pp. D353-D361, Jan 2017

[7] B. Li and C. N. Dewey, "RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome," (in eng), BMC Bioinformatics, vol. 12, p. 323, Aug 2011.

[8] P. Jones et al., "InterProScan 5: genome-scale protein function classification," (in eng), Bioinformatics, vol. 30, no. 9, pp. 1236-40, May 2014.

[9] S. Tang, A. Lomsadze, and M. Borodovsky, "Identification of protein coding regions in RNA transcripts," Nucleic Acids Research, 2015

[10] https://github.com/TransDecoder/TransDecoder/releases

1.3 Software contained or used within this pipeline:

- RSEM
- DIAMOND
- EggNOG
- GeneMarkS-T
- TransDecoder
- InterProScan
- TCLAP
- cereal

CHAPTER 2

Installation

EnTAP is packaged with all of the software necessary to fully annotate a set of transcripts. It is optimized to allow a single-command execution for all steps in the pathway, including paramterization by the user. EnTAP does not have a graphical user interface but it does generate visual summaries for the user at each stage as well as detailed summary files and logs. EnTAP must be installed and configured in order to begin annotating! A test dataset comes with EnTAP to ensure it has been configured properly. Before full EnTAP installation, dependencies must be checked to see if they are included in your system (many are by default) and the accompanying pipeline software will need to be installed (unless already present on the system).

1. *System Requirements*
2. *Dependency Check*
3. *Pipeline Software*
4. *EnTAP Installation*

After installation is complete, EnTAP must be configured in order to start using it. Configuration will simply download the necessary databases that are used by EnTAP.

2.1 System Requirements

- Operating System
  - UNIX-based systems
  - Tested on 64 bit systems: ubuntu 16.04, Rocks 6.1, Centos 6.3
- Storage Minimum
  - EnTAP Database (Gene Ontology References + UniProt Mapping + NCBI Taxonomy): 1.5Gb
  - EggNOG Databases: 24Gb
  - DIAMOND Databases: ~13Gb (with RefSeq Complete Protein + Uniprot Swiss-Prot)
  - Additional storage for files generated depending on transcriptome size: upwards of 15Gb

- Memory
  - At least 16 Gb of RAM (will vary depending on DIAMOND database sizes). More memory is highly recommended to reduce execution times.

2.2 Dependency Check

Before continuing on in the installation process, ensure that the following dependencies are fully installed on your system:

- C++11 compiler (GCC 4.8.1 or later)
- CMake (3.00 or later)
- **Python (2.7.12 or later) with support for the following modules**
  - Matplotlib (figures generated by EnTAP)
- Unix wget (generally included in most distros)
- Unix gzip/tar (generally included in most distros)

2.3 Pipeline Software

EnTAP leverages several software distributions within the pipeline to provide the best quality annotations. The packages used (and their current/tested versions) can be seen below. This is not to say that newer versions will not be compatible, however they have not been tested yet with EnTAP. By default, EnTAP will use Transdecoder for frame selection, however both TransDecoder and GeneMarkS-T are supported and you may install either.

Note: If the software is already installed on your system, this stage can be skipped

---

Software:

- RSEM (Expression Filtering with alignment file): version 1.3.3 packaged with EnTAP
  - Version 1.3.0
- TransDecoder (Frame Selection): version 5.3.0 packaged with EnTAP
- GeneMarkS-T (Frame Selection): version 5.1 must be installed separately (if not using TransDecoder)
- DIAMOND (Similarity Search): version 0.9.9 packaged with EnTAP
  - Version 0.8.31
  - Version 0.9.19
  - Version 0.9.9
- InterProScan (Protein Databases): version 5.19 must be installed separately

2.4 EnTAP Download

First, download and extract the latest release(tagged) version from GitLab: https://gitlab.com/EnTAP/EnTAP/tags

Each of the pipeline software mentioned above (with the exception of GeneMarkS-T and InterProScan) are contained within the /libs directory of the EnTAP repo. GeneMarkS-T must be acquired from the website linked previously due to licensing (free for academic use).

RSEM and DIAMOND both require compilation from source code while GeneMarkS-T does not. To compile these, follow the directions below. These are also found on the respective GitHub pages and are subject to change depending on the version.

2.5 DIAMOND Installation

From root EnTAP directory...

```
cd libs/diamond-0.8.31
mkdir bin
cd bin
cmake ..
```

Run the following command to install globally:

```
make install
```

Run the following command to compile:

```
make
```

All set! Ensure that DIAMOND has been properly setup and add the correct path to the entap_config.txt file. If installed globally, add 'diamond' (without quotes) to the file. If installed locally, add 'path/to/EnTAP/libs/diamond-0.9.9/bin/diamond'.

2.6 RSEM Installation

From root EnTAP directory...

```
cd libs/RSEM-1.3.0
make
make ebseq
```

Run the following command to install globally:

```
make install
```

All set! Ensure that RSEM has been properly setup and add the correct path to the entap_config.txt file. If installed globally keep blank. If installed locally, add 'path/to/EnTAP/libs/RSEM-1.3.0/'.

2.7 EnTAP Installation

Once dependencies and pipeline software have been installed, you can now continue to install EnTAP!

Within the main directory, execute the following command:

```
cmake CMakeLists.txt
```

This will generate a MakeFile. Then execute:

```
make
```

Or to install to a destination directory:

```
cmake CMakeLists.txt -DCMAKE_INSTALL_PREFIX=/destination/dir
```

```
make install
```

If you receive no errors, please move on to the last stage in installation, configuration.

CHAPTER 3

Configuration

After installation is complete, EnTAP must be configured for use. This stage will simply download and configure the necessary databases for full functionality. Configuration can be ran at anytime by a user or the system admin if they would like to update databases. It is in the installation section because it might be a bit easier to setup common databases beforehand that can be shared by multiple users.

I'll break this up into two sections, *Ini File* and *Usage*. The Ini File section will just describe how to ensure EnTAP is reading from the correct paths, which can be easily changed in the entap_config.ini (more on that later!). It will also go over the directories included in the installation. The Usage sections will go over the basic usage during the Configuration stage of EnTAP and how to setup reference databases.

3.1 Ini File

From here on out, the "execution", or "EnTAP", directory will refer to the directory containing the EnTAP install (or binary file). Typically, this will just be at the root directory that was downloaded from the repository. All paths mentioned in this documentation will be relative to this directory.

Why is this important? EnTAP relies on several accompanying software packages and databases in order to run properly. Correct recognition of these paths is crucial and, as such, needed an entire section! The entap_config.ini is the answer to this pathing issue. It contains all of the necessary paths required for EnTAP (among many other commands) to run and can be configured as seen fit.

When a user is trying to execute EnTAP, they must specify the path to this ini file with the ini flag. By default, the ini file comes with some preset paths based on the installation directory. However, these should be checked for validity. If the ini file is not specified and there is not one in the working directory, an empty entap_config.ini will be generated with the following presets for execution paths. The ini file contains many other commands, but only the execution paths are required for configuration (specifically DIAMOND), so I will get to the others later on.

- diamond-exe=/EnTAP/libs/diamond-0.9.9/bin/diamond
- rsem-sam-validator=/EnTAP/libs/RSEM-1.3.0/rsem-sam-validator
- rsem-calculate-expression=/EnTAP/libs/RSEM-1.3.3/rsem-calculate-expression
- rsem-prepare-reference=/EnTAP/libs/RSEM-1.3.3/rsem-prepare-reference

- rsem-convert-sam-for-rsem=/EnTAP/libs/RSEM-1.3.3/convert-sam-for-rsem
- genemarkst-exe=/EnTAP/libs/gmst_linux_64/gmst.pl
- transdecoder-long-exe=EnTAP/libs/TransDecoder-v5.3.0/TransDecoder.LongOrfs
- transdecoder-predict-exe=EnTAP/libs/TransDecoder-v5.3.0/TransDecoder.Predict
- interpro_exe_path=interproscan.sh

If something is globally installed, such as "interproscan-exe" above, put how you'd normally run the software after the '='. As an example, running DIAMOND through a global installation may simply be "diamond". The Ini File line for DIAMOND will simply read:

- diamond-exe=diamond

Warning: Be sure to at least set the DIAMOND path before moving on

Below is a sample ini file with all of the defaults that should be included in the EnTAP repository.

```
#-------------------------------
# [ini_instructions]
#When using this ini file keep the following in mind:
#        1. Do not edit the input keys to the left side of the '=' sign
#        2. Be sure to use the proper value type (either a string, list, or number)
#        3. Do not add unecessary spaces to your input
#        4. When inputting a list, only add a ',' between each entry
#-------------------------------
# [configuration]
#-------------------------------
#Specify which EnTAP database you would like to download/generate or use throughout␣
↪execution. Only one is required.
#    0. Serialized Database (default)
#    1. SQLITE Database
#It is advised to use the default Serialized Database as this is fastest.
#type:list (integer)
data-type=0,
#-------------------------------
# [entap]
#-------------------------------
#Path to the EnTAP binary database
#type:string
entap-db-bin=/bin/entap_database.bin
#Path to the EnTAP SQL database (not needed if you are using the binary database)
#type:string
entap-db-sql=/databases/entap_database.db
#Path to the EnTAP graphing script (entap_graphing.py)
#type:string
entap-graph=/src/entap_graphing.py
#-------------------------------
# [expression_analysis]
#-------------------------------
#Specify the FPKM threshold with expression analysis. EnTAP will filter out␣
↪transcripts below this value. (default: 0.5)
#type:decimal
fpkm=0.5
#Specify this flag if your BAM/SAM file was generated through single-end reads
#Note: this is only required in expression analysis
```

(continues on next page)

(continued from previous page)

```
#Default: paired-end
#type:boolean (true/false)
single-end=false
#-------------------------------
# [expression_analysis-rsem]
#-------------------------------
#Execution method of RSEM Calculate Expression.
#Example: rsem-calculate-expression
#type:string
rsem-calculate-expression=/libs/RSEM-1.3.3//rsem-calculate-expression
#Execution method of RSEM SAM Validate.
#Example: rsem-sam-validator
#type:string
rsem-sam-validator=/libs/RSEM-1.3.3//rsem-sam-validator
#Execution method of RSEM Prep Reference.
#Example: rsem-prepare-reference
#type:string
rsem-prepare-reference=/libs/RSEM-1.3.3//rsem-prepare-reference
#Execution method of RSEM Convert SAM
#Example: convert-sam-for-rsem
#type:string
convert-sam-for-rsem=/libs/RSEM-1.3.3//convert-sam-for-rsem
#-------------------------------
# [frame_selection]
#-------------------------------
#Select this option if all of your sequences are complete proteins.
#At this point, this option will merely flag the sequences in your output file
#type:boolean (true/false)
complete=false
#Specify the Frame Selection software you would like to use. Only one flag can be␣
↪specified.
#Specify flags as follows:
#    1. GeneMarkS-T
#    2. Transdecoder (default)
#type:integer
frame-selection=2
#-------------------------------
# [frame_selection-genemarks-t]
#-------------------------------
#Method to execute GeneMarkST. This may be the path to the executable.
#type:string
genemarkst-exe=gmst.pl
#-------------------------------
# [frame_selection-transdecoder]
#-------------------------------
#Method to execute TransDecoder.LongOrfs. This may be the path to the executable or␣
↪simply TransDecoder.LongOrfs
#type:string
transdecoder-long-exe=TransDecoder.LongOrfs
#Method to execute TransDecoder.Predict. This may be the path to the executable or␣
↪simply TransDecoder.Predict
#type:string
transdecoder-predict-exe=TransDecoder.Predict
#Transdecoder only. Specify the minimum protein length
#type:integer
transdecoder-m=100
#Specify this flag if you would like to pipe the TransDecoder command '--no_refine_
↪starts' when it is executed. Default: False
```

(continues on next page)

(continued from previous page)

```
#This will 'start refinement identifies potential start codons for 5' partial ORFs␣
↪using a PWM, process on by default.'
#type:boolean (true/false)
transdecoder-no-refine-starts=false
#-------------------------------
# [general]
#-------------------------------
#Specify the output format for the processed alignments.Multiple flags can be␣
↪specified:
#    1. TSV Format (default)
#    2. CSV Format
#    3. FASTA Amino Acid (default)
#    4. FASTA Nucleotide (default)
#    5. Gene Enrichment Sequence ID vs. Effective Length TSV (default)
#    6. Gene Enrichment Sequence ID vs. GO Term TSV (default)
#type:list (integer)
output-format=1,3,4,5,6,
#-------------------------------
# [ontology]
#-------------------------------
# Specify the ontology software you would like to use
#Note: it is possible to specify more than one! Just usemultiple --ontology flags
#Specify flags as follows:
#    0. EggNOG (default)
#    1. InterProScan
#type:list (integer)
ontology=0,
#Specify the Gene Ontology levels you would like printed
#A level of 0 means that every term will be printed, while a level of 1 or higher
#means that that level and anything higher than it will be printed
#It is possible to specify multiple flags as well
#Example/Defaults: --level 0 --level 1
#type:list (integer)
level=0,1,
#-------------------------------
# [ontology-eggnog]
#-------------------------------
#Path to the EggNOG SQL database that was downloaded during the Configuration stage.
#type:string
eggnog-sql=/databases/eggnog.db
#Path to EggNOG DIAMOND configured database that was generated during the␣
↪Configuration stage.
#type:string
eggnog-dmnd=/bin/eggnog_proteins.dmnd
#-------------------------------
# [ontology-interproscan]
#-------------------------------
#Execution method of InterProScan. This is how InterProScan is generally ran on your␣
↪system.  It could be as simple as 'interproscan.sh' depending on if it is globally␣
↪installed.
#type:string
interproscan-exe=interproscan.sh
#Select which databases you would like for InterProScan. Databases must be one of the␣
↪following:
#    -tigrfam
#    -sfld
#    -prodom
```

(continues on next page)

(continued from previous page)

```
#     -hamap
#     -pfam
#     -smart
#     -cdd
#     -prositeprofiles
#     -prositepatterns
#     -superfamily
#     -prints
#     -panther
#     -gene3d
#     -pirsf
#     -coils
#     -morbidblite
#Make sure the database is downloaded, EnTAP will not check!
#--protein tigrfam --protein pfam
#type:list (string)
protein=
#-------------------------------
# [similarity_search]
#-------------------------------
#Method to execute DIAMOND. This can be a path to the executable or simply 'diamond'␣
↪if installed globally.
#type:string
diamond-exe=/libs/diamond-0.9.9/bin/diamond
#Specify the type of species/taxon you are analyzing and would like alignments closer␣
↪in taxonomic relevance to be favored (based on NCBI Taxonomic Database)
#Note: replace all spaces with underscores '_'
#type:string
taxon=
#Select the minimum query coverage to be allowed during similarity searching
#type:decimal
qcoverage=50
#Select the minimum target coverage to be allowed during similarity searching
#type:decimal
tcoverage=50
#Specify the contaminants you would like to flag for similarity searching.␣
↪Contaminants can be selected by species or through a specific taxon (insecta) from␣
↪the NCBI Taxonomy Database. If your taxon is more than one word just replace the␣
↪spaces with underscores (_).
#Note: since hits are based upon a multitide of factors, a contaminant might end up␣
↪being the best hit for an alignment. In this scenario, EnTAP will flag the␣
↪contaminant and it can be removed if you would like.
#type:list (string)
contam=
#Specify the E-Value that will be used as a cutoff during similarity searching.
#type:decimal
e-value=1e-05
#List of keywords that should be used to specify uninformativeness of hits during␣
↪similarity searching. Generally something along the lines of 'hypothetical' or
↪'unknown' are used. Each term should be separated by a comma (,) This can be used␣
↪if you would like to tag certain descriptions or would like to weigh certain␣
↪alignments differently (see full documentation)
#Example (defaults):
#conserved, predicted, unknown, hypothetical, putative, unidentified, uncultured,␣
↪uninformative, unnamed
#type:list (string)
uninformative=conserved,predicted,unknown,unnamed,hypothetical,putative,unidentified,
↪uncharacterized,uncultured,uninformative,
```

(continues on next page)

(continued from previous page)

3.2 Preparing Your Reference Databases

All source databases must be provided in FASTA format (protein) so that they can be indexed for use by DIAMOND. This can be completed independent of EnTAP with DIAMOND (- - makedb flag) or as part of the Configuration phase of EnTAP. This section will focus on downloading and preparing some of the more common FASTA source databases. If you already have DIAMOND databases configured, you can skip to *Running Configuration*. Even if you have a DIAMOND database already configured, Configuration must still be ran!

While any protein FASTA database can be used, it is recommended to use NCBI (Genbank) sourced databases such as RefSeq databases or NR. In addition, EnTAP can easily accept EBI databases such as UniProt/SwissProt.

EnTAP can recognize the species information from these header formats ONLY (NCBI and UniProt):

- [homo sapiens]
- OS=homo sapiens

If the individual FASTAs in a custom database you create do not adhere to one of these two formats, it will not be possible to weight taxonomic or contaminant status from them. You will need to change the headers to ensure they align.

The following FTP sites contain common reference databases that EnTAP can recognize:

- RefSeq: ftp://ftp.ncbi.nlm.nih.gov/refseq/release/complete/
- Plant RefSeq: ftp://ftp.ncbi.nlm.nih.gov/refseq/release/plant/
- Mammalian Vertebrate RefSeq: ftp://ftp.ncbi.nlm.nih.gov/refseq/release/vertebrate_mammalian/
- Other Vertebrate RefSeq: https://ftp.ncbi.nih.gov/refseq/release/vertebrate_other/
- Invertebrate RefSeq: https://ftp.ncbi.nih.gov/refseq/release/invertebrate/
- NR: ftp://ftp.ncbi.nlm.nih.gov/blast/db/
- SwissProt: ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/complete/uniprot_sprot.fasta.gz
  - Reviewed
  - It is highly recommended to use the UniProt SwissProt database as EnTAP will map all UniProt alignments to additional database cross-references
- TrEMBL: ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/complete/uniprot_trembl.fasta.gz
  - Unreviewed

Both Uniprot databases (SwissProt and TrEMBL) can be downloaded on a Unix system through the following command:

```
wget ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/
↪complete/uniprot_sprot.fasta.gz
```

Or, for the TrEMBL database:

```
wget ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/
↪complete/uniprot_trembl.fasta.gz
```

Alternatively, the NCBI databases must be downloaded in separate, smaller files, and concatenated together. As an example, the following commands will download and combine the NR database files:

Download:

```
wget ftp://ftp.ncbi.nlm.nih.gov/blast/db/nr.*.tar.gz
```

Decompress/Concatenate:

```
tar -xvzf nr.*

cat nr.* > nr_database.fasta
```

It is generally recommended that a user select at least three databases with varying levels of curation. Unless the species is very non-model (i.e. does not have close relatives in databases such as RefSeq, it is not necessary to use the full NR database which is less curated). Once your FASTA databases are ready, move on to *Running Configuration*.

3.3 Running Configuration

Once you have your protein FASTA database ready, you can begin to run the Configuration stage. As mentioned before, Configuration will only need to be run once prior to Execution unless you would like to configure/update more databases.

To run configuration with a FASTA database to output directory path/to/output (default is current working directory), the command is as follows (additional databases can be specified if necessary with the -d flag and threads with the -t flag):

```
EnTAP --config -d path/to/database.fasta -d path/to/database2.fasta --out-dir path/to/
↪output -t 8 --ini path/to/ini
```

If your databases are already indexed for DIAMOND, you can simply provide the paths in the .ini file and run the following command with 8 threads:

```
EnTAP --config -t 8 --ini path/to/ini
```

Note: This is the only stage that requires connection to the Internet.

In both cases, the following databases will be downloaded and configured:

- **EnTAP Binary Database:**
  - Comprised of Gene Ontology, UniProt, and Taxonomic mappings for use during Execution. FTP downloaded file.
  - Downloaded from https://treegenesdb.org/FTP/EnTAP/latest/databases/entap_database.bin.gz
  - Filename: entap_database.bin
  - The SQL version is the same database, but formatted as a SQL database. Only one version of the database is needed (binary is used by default)
- **EggNOG DIAMOND Reference:**
  - Reference database containing EggNOG database entries

  - FASTA file is downloaded and configured for DIAMOND from http://eggnog5.embl.de/download/eggnog_4.1/eggnog-mapper-data/eggnog4.clustered_proteins.fa.gz
  - Filename: eggnog_proteins.dmnd

- **EggNOG SQL Database:**
  - SQL database containing EggNOG mappings
  - Downloaded from http://eggnog5.embl.de/download/eggnog_4.1/eggnog-mapper-data/eggnog.db.gz
  - Filename: eggnog.db

Note: Either the EnTAP binary database (default) or the EnTAP SQL database is required for execution. Both are not needed.

The EnTAP Binary Database is downloaded from the FTP addresses below. By default, the binary version will be downloaded and used. Only one version is required. If you experience any trouble in downloading, you can simply specify the - - data-generate flag during Configuration to configure it locally (more on that later). The database for the newest version of EnTAP will always reside in the "latest" FTP directory. Keep in mind, if you are using an older version of EnTAP, you do not want to download from the "latest" directory. Instead, you will need to consider the version you are using. The FTP will always be updated only when a new database version is created. For example, if you see v0.8.2 and v0.8.5 on the FTP while you are using v0.8.3, you will download the database located in the v0.8.2 directory.

- https://treegenesdb.org/FTP/EnTAP/latest/databases/entap_database.bin.gz
- https://treegenesdb.org/FTP/EnTAP/latest/databases/entap_database.db.gz

Warning: DIAMOND databases must be configured and eventually executed with the same version of DIAMOND.

CHAPTER 4

Configuration Flags

These are the flags for the configuration process of EnTAP. These will be used via the command line (denoted CMD) or ini file (denoted INI).

4.1 Required Flags

These are the required flags that must be used during Configuration. These will be used via the command line (denoted CMD) or ini file (denoted INI).

4.1.1 –config [CMD]

- Must be specified in order for EnTAP to run Configuration as opposed to normal execution

4.1.2 –ini [string] [CMD]

- Point to entap_config.ini to specify file paths
- DIAMOND is the only path necessary during Configuration
- Default: entap_config.ini residing in the current working directory

4.2 Basic Flags

These are the general flags that can be used during Configuration. These will be used via the command line (denoted CMD) or ini file (denoted INI).

4.2.1 -d / –database [string] [CMD]

- Specify any number of FASTA formatted databases you would like to configure for EnTAP
- Not necessary if you already have DIAMOND configured databases (.dmnd)

4.2.2 –out-dir [string] [CMD]

- Specify an output directory for the databases to be sent to (recommended)
- This will send the EnTAP database and DIAMOND databases to this location

4.2.3 -t / –threads [number] [CMD]

- Specify thread number for Configuration

4.2.4 –data-generate [CMD]

- Specify this flag is you would like to generate the EnTAP database rather than downloading from FTP (default)
- I'd only use this if you're having issues with the FTP

4.2.5 –data-type [flag] [INI]

- Specify which databases you'd like to generate/download
  - 0. Binary Database (default) - This will be much quicker and is recommended
  - 1. SQL Database - Slower although will be more easily compatible with every system
- This can be flagged multiple times (ex: - - data-type 0 - - data-type 1)
- I would not use this flag unless you are experiencing issues with the EnTAP Binary Database

CHAPTER 5

Test Data

Before continuing to running EnTAP, it is advised to do a test run to ensure that everything is properly configured. There should be no errors in the test run. The test data resides within the /test_data directory of the main EnTAP directory. This will walk you through configuring a database for DIAMOND (if you haven't already done so) and executing EnTAP with and without frame selection.

Before we begin, make sure that the paths in the configuration file are correct. Since we are running the configuration stage, EnTAP will check to make sure you have the other databases downloaded (which should have been done prior to this). To begin the test, execute the following command to configure the test DIAMOND database:

```
EnTAP --config -d /test_data/swiss_prot_test.fasta --out-dir /test_data --ini path/to/
↪ini_file
```

This should finish very shortly without any errors and you should find a swiss_prot_test.dmnd file within the /test_data directory.

Next up is verifying the main execution stage! Once again, first ensure that the Config File has all of the correct paths. We are going to check an execution with and without frame selection. If you are not going to use frame selection, you may skip this test!

Note: The following tests will take longer as they will be testing the entire pipeline and running against the larger EggNOG database.

To test EnTAP with frame selection, execute the following command:

```
EnTAP --runP -i /test_data/trinity.fnn -d /test_data/bin/swiss_prot_test.dmnd --ini␣
↪path/to/ini_file
```

To test EnTAP without frame selection, execute the following command:

```
EnTAP --runP -i /test_data/trinity.faa -d /test_data/swiss_prot_test.dmnd --ini path/
↪to/ini_file
```

These should run without error and you should have several files within the created /entap_outfiles directory. The final_annotations_lvl0.tsv file should resemble the test_data/final_annotations_test.tsv file.

If any failures were seen during the above executions, be sure to go through each stage of installation and configuration to be sure everything was configured correctly before continuing. If you have no received any errors, the following pages will go through the main annotation portion of the pipeline.

Note: The ini file may be distributed to each user to ensure they are pointing to the correct database and execution paths.

CHAPTER 6

Execution

Execution is the main stage of EnTAP that will annotate a transcriptome input by the user. After following the installation/configuration steps at least once, all of the required databases have been downloaded and configured so that Execution can be ran. Configuration will not need to be ran again unless you would like to update any databases or configure more.

The following stages will be run:

1. Expression Filtering (optional)
2. Frame Selection (optional)
3. Similarity Search
4. Orthologous Group Assignment
5. InterProScan (optional)

6.1 Input Files:

Required:

- .FASTA formatted transcriptome file (either protein or nucleotide)
- .dmnd (DIAMOND) indexed databases, which can be formatted in the Configuration stage.

Optional:

- **.BAM/.SAM alignment file. If left unspecified expression filtering will not be performed.**
  - This can be generated by software that does not perform gapped alignments such as Bowtie (not Bowtie2). All you need to generate an alignment file is a pair of reads and your assembled transcriptome!

6.2 Sample Run:

A specific run flag (**runP/runN**) must be used:

- runP: Indicates blastp. Frame selection will be ran if nucleotide sequences are input
- runN: Indicates blastx. Frame selection will not be ran with this input

An example run with a nucleotide transcriptome (transcriptome.fasta), two reference DIAMOND databases, an alignment file (alignment.sam), and 8 threads:

```
EnTAP --runP -i path/to/transcriptome.fasta -d path/to/database.dmnd -d path/to/
↪database2.dmnd -a path/to/alignment.sam --ini path/to/ini_file -t 8
```

With the above command, the entire EnTAP pipeline will run. Both Frame Selection and Expression Filtering can be skipped if preferred by the user. If a protein transcriptome is input with the runP flag, or the runN flag is used, Frame Selection will be skipped. If there is not a short read alignment file provided in SAM/BAM format, then Expression Filtering via RSEM will be skipped.

6.3 Expression Analysis

The goal of expression filtering, or transcript quantification, is to determine the relative abundance levels of transcripts when taking into account the sequenced reads and how they map back to the assembled transcriptome and using this information to filter out suspect expression profiles possibly originated from poor or incomplete assemblies. Filtering is done through the use of the FPKM (fragments per kilobase per of million mapped reads) , or a measurable number of expression. This can be specified with the - -fpkm flag as specified above. EnTAP will use this FPKM value and remove any sequences that are below the threshold.

6.4 Frame Selection

Frame selection is the process of determining the coding region of a transcript. Oftentimes, due to assembly errors or other factors, a coding region may not be found for a transcript and EnTAP will remove this sequence. When a coding region is found, EnTAP will include the sequence for further annotation.

6.5 Taxonomic Favoring and Contaminant Filtering

Taxonomic contaminant filtering (as well as taxonomic favoring) is based upon the NCBI Taxonomy database. In saying this, all species/genus/lineage names must be contained within this database in order for it to be recognized by EnTAP.

Contaminant Filtering:

Contaminants can be introduced during collection or processing of a sample. A contaminant is essentially a species that is not of the target species you are collecting. Some common contaminants are bacteria and fungi that can sometimes be found within collected samples. Transcripts flagged as contaminants will be written to a file appended with "_contam", but not removed from the final annotations file. Oftentimes, researchers would like to remove these sequences from the dataset.

One or more contaminants can be specified in the ini file (separated by a comma). An example of flagging bacteria and fungi as contaminants can be seen below:

```
contam=fungi,bacteria
```

Some common contaminants: * insecta * fungi * bacteria

Taxonomic Favoring

During best hit selection of similarity searched results, taxonomic consideration can utilized. If a certain lineage (pinus) is specified, hits closer in taxonomic lineage to this selection will be chosen. Any lineage such as species/kingdom/phylum can be utilized as long as it is contained within the Taxonomic Database. If it is not located within the database, EnTAP will stop the execution immediately and let you know!

This feature can be utilized via the ini file. An example can be seen below (Note: replace any spaces with an underscore):

```
taxon=pinus_taeda
```

Another example could be:

```
taxon=pinus
```

Keep in mind, EnTAP will weigh the E-Value (within a database)and Coverage of the alignment before taxonomic weight in order to provide the most accurate result. If both the E-Value and Coverage are relatively similar, EnTAP will leverage taxonomic information.

6.6 Picking Up Where You Left Off

In order to save time and make it easier to do different analyses of data, EnTAP allows for picking up where you left off if certain stages were already ran and you'd like analyze data with different contaminant flags or taxonomic favoring. As an example, if similarity searching was ran previously you can skip aligning against the database and analyze the data to save time. However, the - - overwrite flag will not allow for this as it will remove previous runs and not recognize them.

In order to pick up and skip re-running certain stages again, the files that were ran previously **must** be in the same directories and have the same names. With an input transcriptome name of 'transcriptome' and example database of 'complete.protein':

- **Expression Filtering**
  - transcriptome.genes.results
- **Frame Selection**
  - transcriptome.fasta.faa
  - transcriptome.fasta.fnn
  - transcriptome.fasta.lst
- **Similarity Search**
  - blastp_transcriptome_complete.protein.faa.out
- **Gene Family**
  - blastp_transcriptome_eggnog_proteins.out (for runP)
  - blastp_transcriptome_eggnog_proteins.out (for runN)

Since file naming is based on your input as well, the flags below **must** remain the same:

- (- - runN / - - runP)

- (- - ontology)
- (- - protein)
- (-i / - - input)
- (-a / - - align)
- **(-d / - - database)**
  - Does not necessarily need to remain the same. If additional databases are added, EnTAP will recognize the new ones and run similarity searching on them whilst skipping those that have already been ran
- (- - qcoverage)
- (- - tcoverage)
- (- - no-trim)
- (- - out-dir)

6.7 State Control

Warning: This is experimental and certain configurations may not work. This is not needed if you'd like to run certain portions because of "picking up where you left off!"

State control of EnTAP allows you to further customize your runs. This is separate from the exclusion of - - align flag to skip expression filtering, or runP, instead of runN, to skip frame selection. You probably will never actually have to use this feature! Nonetheless, state control is based around the following stages of EnTAP:

1. Expression Filtering
2. Frame Selection
3. Transcriptome Filtering (selection of final transcriptome)
4. Similarity Search
5. Gene Ontology / Gene Families

With this functionality of EnTAP, you can execute whatever states you would like with certain commands. Using a '+' will execute from that state to the end, while using a 'x' will stop at that state. These basic commands can be combined to execute whatever you would like. It's easier if I lay out some examples:

- **(- - state 1+)**
  - This will start at expression filtering and continue to the end of the pipeline
- **(- - state 1+4x)**
  - This will start at expression filtering and stop after similarity search
- **(- - state 4x)**
  - This will just execute similarity search and stop
- **(- - state 1+3x5)**
  - This will essentially execute every stage besides similarity searching

The default 'state' of EnTAP is merely '+'. This executes every stage of the pipeline (or attempts to if the correct commands are in place).

CHAPTER 7

Execution Flags

These are the flags for the execution process of EnTAP. Flags will be repeated throughout these categories where relevant. These will be used via the command line (denoted CMD) or ini file (denoted INI).

7.1 Required Flags

These are the required flags that must be used during Execution. All of these commands must be used via the command line.

7.1.1 –runP / –runN [CMD]

- Specify a blastp or blastx annotation
- If - -runP is selected with a nucleotide input, frame selection will be ran and annotation stages will be executed with protein sequences (blastp)
- If - -runP is selected with a protein input, frame selection will not be ran and annotation will be executed with protein sequences (blastp)
- If - -runN is selected with nucleotide input, frame selection will not be ran and annotation will be executed with nucleotide sequences (blastx)

7.1.2 -i / –input [string] [CMD]

- Path to the transcriptome file (either nucleotide or protein)

7.1.3 -d / –database [multi-string] [CMD]

- Specify up to 5 DIAMOND indexed (.dmnd) databases to run similarity search against

7.1.4 –ini [string] [CMD]

- Point to entap_config.ini to specify file paths
- Default: entap_config.ini residing in the current working directory

7.2 Basic Usage Flags

These are just some basic/common flags to use during Execution. You'll see these repeated on the other pages. All these flags can only be used via the command line.

7.2.1 –runP / –runN [CMD]

- Specify a blastp or blastx annotation
- If - -runP is selected with a nucleotide input, frame selection will be ran and annotation stages will be executed with protein sequences (blastp)
- If - -runP is selected with a protein input, frame selection will not be ran and annotation will be executed with protein sequences (blastp)
- If - -runN is selected with nucleotide input, frame selection will not be ran and annotation will be executed with nucleotide sequences (blastx)

7.2.2 -i / –input [string] [CMD]

- Path to the transcriptome file (either nucleotide or protein)

7.2.3 -d / –database [multi-string] [CMD]

- Specify up to 5 DIAMOND configured databases you would like to search agianst

7.2.4 –out-dir [string] [CMD]

- Specify an output directory for all of the files generated by EnTAP
- Default: Current working directory

7.2.5 -t / –threads [integer] [CMD]

- Specify thread count for execution

7.2.6 –ini [string] [CMD]

- Point to entap_config.ini to specify file paths
- Default: entap_config.ini residing in the current working directory

7.2.7 –version [CMD]

- Prints the current EnTAP version you are running

7.2.8 -a / –align [string] [CMD]

- Path to alignment file (either SAM or BAM format)
- **Note:** Ignoring this flag will skip expression filtering
- If you have ran alignment with single end reads be sure to use the - -single-end flag as well (paired-end is default)
- Be sure to specify an FPKM threshold

7.3 General EnTAP Flags

These are the general EnTAP flags that can be used during Execution. They are not specific to a certain stage of the pipeline. These will be used via the command line (denoted CMD) or ini file (denoted INI).

7.3.1 –runP / –runN [CMD]

- Specify a blastp or blastx annotation
- If - -runP is selected with a nucleotide input, frame selection will be ran and annotation stages will be executed with protein sequences (blastp)
- If - -runP is selected with a protein input, frame selection will not be ran and annotation will be executed with protein sequences (blastp)
- If - -runN is selected with nucleotide input, frame selection will not be ran and annotation will be executed with nucleotide sequences (blastx)

7.3.2 -i / –input [string] [CMD]

- Path to the transcriptome file (either nucleotide or protein)

7.3.3 -d / –database [multi-string] [CMD]

- Specify up to 5 DIAMOND indexed (.dmnd) databases to run similarity search against

7.3.4 –ini [string] [CMD]

- Point to entap_config.ini to specify file paths
- Default: entap_config.ini residing in the current working directory

7.3.5 –out-dir [string] [CMD]

- Specify an output directory for all of the files generated by EnTAP
- Default: Current working directory

7.3.6 -t / –threads [integer] [CMD]

- Specify thread number for execution

7.3.7 –out-dir [string] [CMD]

- Specify an output directory for all of the files generated by EnTAP
- Default: Current working directory

7.3.8 –overwrite [CMD]

- All previously ran files will be overwritten
- Without this flag EnTAP will recognize previous runs and skip portions that were already ran

7.3.9 –graph [CMD]

- This will check whether or not your system has graphing functionality supported and exit
- If Python with the Matplotlib module are installed on your system graphing should be enabled!
- This can be specified on its own

7.3.10 -t / –threads [integer]

- Specify the number of threads of execution

7.3.11 –no-trim [CMD]

- By default, EnTAP will trim your sequence headers to the first space to maintain compatbility across different software. Using this flag will instead retain the information of the header by removing all spaces.
- Example:
  - >TRINITY_231.1 protein12312_43 inform
  - >TRINITY_231.1protein12312_43inform
- A word of caution when using this flag. EnTAP may have difficulty matching sequence headers from BAM/SAM files to your input transcriptome during Expression Analysis. You will receive an error if this occurs.

7.3.12 –state [string] [CMD]

- Precise control over execution *stages*. This flag allows for certain parts to be ran while skipping others.
- Warning: This may cause issues depending on what you plan on running!

7.3.13 –version [CMD]

- Prints the current EnTAP version you are running

7.3.14 –no-check [CMD]

- EnTAP checks execution paths and inputs prior to annotating to prevent finding out your input was wrong until midway through a run. Using this flag will eliminate the check (not advised to use!)

7.3.15 –output-format [multi-integer] [INI]

- Specify multiple output file formats for each stage of the pipeline
  - 1. TSV File (default)
  - 2. CSV File
  - 3. FASTA Protein File (default)
  - 4. FASTA Nucleotide File (default)
  - 5. Gene Enrichment Gene ID + Effective Length (default)
  - 6. Gene Enrichment Gene ID + GO Terms (default)

7.4 Expression Analysis Flags

These are the flags specific to Expression Analysis using RSEM. These will be used via the command line (denoted CMD) or ini file (denoted INI).

7.4.1 -a / –align [string] [CMD]

- Path to alignment file (either SAM or BAM format)
- **Note:** Ignoring this flag will skip expression filtering
- If you have ran alignment with single end reads be sure to use the - -single-end flag as well (paired-end is default)
- Be sure to specify an FPKM threshold

7.4.2 –fpkm [decimal] [INI]

- Specify FPKM cutoff for expression filtering
- Default: 0.5

7.4.3 –single-end [INI]

- Signify your reads are single end for RSEM execution
- Default: paired-end

7.5 Frame Selection Flags

These are the flags specific to Frame Selection using either GeneMarkS-T or TransDecoder. These will be used via the command line (denoted CMD) or ini file (denoted INI).

7.5.1 General Flags

–frame-selection [integer] [INI]

- **Specify the frame selection software to use**
  - 1. GeneMarkS-T
  - 2. TransDecoder (default)

–complete [INI]

- Tell EnTAP to mark all of the transcripts as 'complete'. This will only be seen in the final output and will not affect the run.

7.5.2 TransDecoder Specific Flags

–transdecoder-m [INI]

- Specify the minimum protein length

–transdecoder-no-refine-starts [INI]

- Specify this flag if you would like to pipe the command, '–no_refine_starts' to TransDecoder.
- "This will 'start refinement identifies potential start codons for 5' partial ORFs using a PWM, process on by default". (From TransDecoder documentation)
- Default: False

7.6 Similarity Search Flags

These are the flags specific to Similarity Searching using DIAMOND. These will be used via the command line (denoted CMD) or ini file (denoted INI).

7.6.1 -d / –database [CMD]

- Specify any number of FASTA formatted databases you would like to configure for EnTAP
- Not necessary if you already have DIAMOND configured databases (.dmnd)

7.6.2 –data-type [INI]

- Specify which EnTAP database you'd like to use for execution (UniProt, Gene Ontology, and Taxonomy lookups)
  - 0. Binary Database (default) - This will be much quicker and is recommended
  - 1. SQL Database - Slower although will be more easily compatible with every system
- This can be flagged multiple times (ex: - - data-type 0 - - data-type 1)
- I would not use this flag unless you are experiencing issues with the EnTAP Binary Database

7.6.3 -c / –contam [multi-string] [INI]

- Specify *contaminant* level of filtering
- Multiple contaminants can be selected through repeated flags

7.6.4 –taxon [string] [INI]

- This flag will allow for *taxonomic* 'favoring' of hits that are closer to your target species or lineage. Any lineage can be used as referenced by the NCBI Taxonomic database, such as genus, phylum, or species.
- Format **must** replace all spaces with underscores ('_') as follows: "- -taxon homo_sapiens" or "- -taxon primates"

7.6.5 –level [multi-integer] [INI]

- Specify Gene Ontology levels you would like to normalize to (ex: 0, 1, 2, 3, 4)
- This should only be used as a rough idea, some of the levels can vary slightly
- A level of '0' indicates all levels will be printed while a level of 2 will indicate that all levels of 2 or higher will be printed.
- Any amount of these flags can be used
- Default: 0, 1
- More information at: http://geneontology.org/page/ontology-structure

7.6.6 -e [scientific] [INI]

- Specify minimum E-value cutoff for similarity searching (scientific notation)
- Default: 10E-5

7.6.7 –tcoverage [decimal] [INI]

- Specify minimum target coverage for similarity searching
- Default: 50%

7.6.8 –qcoverage [decimal] [INI]

- Specify minimum query coverage for similarity searching
- Default: 50%

7.6.9 –uninformative [string] [INI]

- Comma-deliminated list of terms you would like to be deemed "uninformative". Any alignments during Similarity Searching tagged as uninformative will be ranked lower.
- Default: conserved, predicted, unnamed, hypothetical, putative, unidentified, uncharacterized, unknown, uncultured, uninformative

7.7 Ontology Flags

These are flags specific to the Ontology portion of the pipeline using either EggNOG or InterProScan. These will be used via the command line (denoted CMD) or ini file (denoted INI).

7.7.1 General Flags

7.7.2 –level [multi-integer] [INI]

- Specify Gene Ontology levels you would like to normalize to (ex: 0, 1, 2, 3, 4)
- This should only be used as a rough idea, some of the levels can vary slightly
- A level of '0' indicates all levels will be printed while a level of 2 will indicate that all levels of 2 or higher will be printed.
- Any amount of these flags can be used
- Default: 0, 1
- More information at: http://geneontology.org/page/ontology-structure

–ontology [multi-integer] [INI]

- Specify which ontology packages you would like to use
  - 0 - EggNOG (default)
  - 1 - InterProScan
- Both or either can be specified with multiple flags
  - Ex: - - ontology 0 - - ontology 1
  - This will run both EggNOG and InterProScan

7.7.3 InterProScan Specific Flags

–protein [multi-string] [INI]

- Use this option if you would like to run InterProScan
- Specify databases to run against (you must have them already installed)
  - tigrfam
  - sfld
  - prodom
  - hamap
  - pfam
  - smart
  - cdd
  - prositeprofiles
  - prositepatterns
  - superfamily
  - prints
  - panther
  - gene3d
  - pirsf
  - coils
  - mobidblite

7.8 API Flags

These flags are used as part of an ongoing effort to create an EnTAP API to support software packages incorporating EnTAP within their framework.

7.8.1 –api-taxon [string] [CMD]

- Check whether a species is part of the EnTAP Taxonomy database specified in the ini file
- Returns json formatted text indicating whether the species was found
- Format **must** replace all spaces with underscores ('_') as follows: "- -taxon homo_sapiens" or "- -taxon primates"

CHAPTER 8

Interpreting the Results

EnTAP provides many output files at each stage of execution to better see how the data is being managed throughout the pipeline:

1. *Final Annotation Results*
2. *Log File / Statistics*
3. *Transcriptomes*
4. *Expression Filtering*
5. *Frame Selection*
6. *Similarity Searching*
7. *Orthologous Groups/Ontology*
8. *Protein Families* (optional)

The two files to check out first are the *final annotations* and *log file*. These files contain a summary of all the information collected at each stage, including statistical analyses. The remaining files are there for a more in depth look at each stage. All files will be contained in "entap_outfiles" directory as default, or different if the - - out-dir flag was specified.

8.1 Final Annotations

The final EnTAP annotations are contained within the */final_results* directory. These files are the summation of each stage of the pipeline and contain the combined information. So these can be considered the most important files!

All .tsv files in this section may have the following header information (from left to right) separated by each portion of the pipeline. Some headers will not be shown if that part of the pipeline was skipped or the information was not found for any of the input sequences.

General Header Information

- Query sequence ID

Frame Selection Header Information (optional)

- Open Reading Frame

Expression Analysis Header Information (optional)

- FPKM
- TPM
- Effective Length

Similarity Search Header Information

- Subject sequence ID
- Percentage of identical matches
- Alignment length
- Number of mismatches
- Number of gap openings
- Start of alignment in query
- End of alignment in query
- Start of alignment in subject
- End of alignment in subject
- Expect (e) value
- Query coverage
- Subject title
- Species
- Taxonomic Lineage
- Origin Database
- Contaminant (yes/no if the hit was flagged as a contaminant)
- Informative (yes/no if he hit was flagged as informative)

Similarity Search UniProt Header Information (optional if aligning against SwissProt database)

- UniProt Database Cross References
- UniProt Additional Information
- UniProt KEGG Terms
- UniProt GO Biological
- UniProt GO Cellular
- UniProt GO Molecular

Ontology EggNOG Header Information

- Seed Ortholog
- Seed E-Value
- Seed Score
- Predicted Gene

- Taxonomic Scope
- OGs (orthologous groups assigned)
- EggNOG Description (EggNOG)
- KEGG Terms (EggNOG)
- GO Biological (Gene Ontology normalized terms)
- GO Cellular (Gene Ontology normalized terms)
- GO Molecular (Gene Ontology normalized terms)
- BIGG Reaction

Ontology InterProScan Header Information

- IPScan GO Biological
- IPScan GO Cellular
- IPScan GO Molecular
- Pathways
- InterPro (InterPro database entry)
- Protein Database (database assigned. Ex: pfam)
- Protein Description (description of database entry)
- E Value (E-value of hit against protein database)

Gene ontology terms are normalized to levels based on the input flag from the user (or the default of 0,1). A level of 0 within the filename indicates that ALL GO terms will be printed to the annotation file. Any other level will print that level and anything higher than it. Normalization of GO terms to levels is generally done before enrichment analysis and is based upon the hierarchical setup of the Gene Ontology database. More information can be found at GO.

- final_annotations_lvlX.tsv
  - As mentioned above, the 'X' represents the normalized GO terms for the annotation (GO terms of X level and higher will be printed)
  - This .tsv file will have the headers as mentioned previously as a summary of the entire pipeline
- final_annotated.faa / .fnn
  - Nucleotide and protein fasta files containing all sequences that either hit databases through similarity searching or through the ontology stage
- final_unannotated.faa / .fnn
  - Nucleotide and protein fasta files containing all sequences that did not hit either through similarity searching nor through the ontology stage
- final_annotations_contam.faa / .fnn / .tsv
  - Nucleotide, protein, and tab-deliminated files containing all annotated sequences that were flagged as a contaminant
- final_annotations_no_contam.faa / .fnn / .tsv
  - Nucleotide, protein, and tab-deliminated files containing all annotated sequences that were not flagged as a contaminant
- final_annotations_lvlX_enrich_geneid_go.tsv
  - Tab-deliminated file that can be used for Gene Enrichment, normalized to gene ontology level

  - First column contains the gene ID and second column contains the Gene Ontology term corresponding to the gene ID
  - A level of 0 will print all GO terms assigned, while any other level will prnit that level and anything higher. Ex: 1 will print all terms of 1 and higher

- final_annotations_lvlX_enrich_geneid_len.tsv
  - Tab-deliminated file that can be used for Gene Enrichment, normalized to gene ontology levels
  - First column contains the gene ID and second columns contains the effective length from Expression Analysis. This file will not be printed if Expression Analysis has not been ran
  - A level of 0 will print all GO terms assigned, while any other level will prnit that level and anything higher. Ex: 1 will print all terms of 1 and higher

8.2 Log File / Statistics

The log file contains a statistical analysis of each stage of the pipeline that you ran. I'll give a brief outline of some of the stats performed:

1. Initial Statistics
   - Transcriptome statistics: n50, n90, average gene length, longest/shortest gene
   - Summary of user flags
   - Summary of execution paths (from config file)
2. Expression analysis
   - Transcriptome statistics: n50, n90, average gene length, longest/shortest gene
   - Summary of sequences kept/removed after filtering
3. Frame Selection
   - Transcriptome statistics: n50, n90, average gene length, longest/shortest gene
   - Summary of frame selection: Partial, internal, complete genes. Genes where no frame was found
4. Similarity Searching
   - Contaminant/uninformative/informative count
   - Phylogenetic/contaminant distribution of alignments
   - Alignment distribution based upon frame results (partial/internal/complete)
   - Sequence count that did not align against a database reference
   - Statistics calculated for each individual database and final results
5. Gene Family Assignment
   - Phylogenetic distribution of gene family assignments
   - Gene Ontology level distribution (note: level 0 means all levels)
   - Gene Ontology category distribution (biological processes, molecular function, cellular component)
6. InterProScan
   - Additional statistics coming soon!
7. Final Annotation Statistics

- Statistical summary of each stage
- Runtime

8.3 Transcriptomes

The */transcriptomes* contains the original, processed, and final transcriptomes being used by EnTAP. The files are as follows with the 'transcriptome' tag based upon the name of your input transcriptome:

- transcriptome.fasta
  - This file is essentially a copy of your input transcriptome. The sequence ID's may be changed depending on whether you selected the 'trim' flag or otherwise.
- transcriptome_expression_filtered.fasta
  - As the name implies, this transcriptome is the resultant of the Expression Filtering stage with sequences removed that fall under the FPKM threshold you have specified.
- transcriptome_frame_selected.fasta
  - This transcriptome is the resultant of Frame Selection. Sequences in which a frame was not selected are removed and those with a frame are kept in this file. As a result, this file will always be in protein format.
- transcriptome_final.fasta
  - This is your final transcriptome following the "Transcriptome Filtering" stage of EnTAP. **This transcriptome will be used for the later stages of the pipeline** (Similarity Searching and Ontology). Depending on which methods of execution you chose (runN / runP), the result here may be either protein or nucleotide with Frame Selection and/or Expression Filtering.

8.4 Expression Filtering (RSEM)

The */expression* folder will contain all of the relevant information for this stage of the pipeline. This folder will contain the main files (results from expression analysis software), files processed from EnTAP (including graphs).

8.4.1 RSEM Files: */expression*

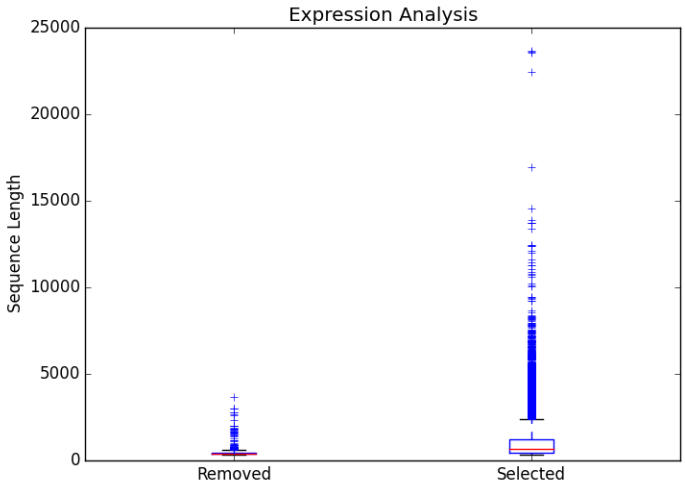
The */expression* directory will contain all of the output from RSEM including a converted BAM file (if you input a SAM) and the results of the expression analysis.

8.4.2 EnTAP Files: */processed*

This directory will contain all of the files produced from EnTAP concerning expression analysis. With a generic transcriptome input of "Species.fasta", these files will have the following format

- Species_removed.fasta
  - Fasta file of sequences that were under the specified FPKM threshold
- Species_kept.fasta
  - Fasta file of sequences that were kept after filtering (over the FPKM threshold)
- */figures*

  - Directory containing a box plot of sequence length vs the sequences that were removed and kept after expression analysis



8.5 Frame Selection (GeneMarkS-T or TransDecoder)

The */frame_selection* folder will contain all of the relevant information for the frame selection stage of the pipeline. This folder will contain results from frame selection software, files *processed* from EnTAP, and *figures* generated from EnTAP.

8.5.1 TransDecoder Files: */frame_selection*

The files within the root */frame_selection* directory contain the results from the frame selection portion of the pipeline. More information can be found at TransDecoder (the descriptions below are taken from there):

- transcripts.fasta.transdecoder.pep
  - Peptide sequences for the final candidate ORFs; all shorter candidates within longer ORFs were removed.
- transcripts.fasta.transdecoder.gff3
  - Positions within the target transcripts of the final selected ORFs
- transcripts.fasta.transdecoder.cds
  - Nucleotide sequences for coding regions of the final candidate ORFs
- .err and .out file
  - These files are will contain any error or general information produced from the TransDecoder run

8.5.2 GeneMarkS-T Files: */frame_selection*

The files within the root */frame_selection* directory contain the results from the frame selection portion of the pipeline. More information can be found at GeneMarkS-T. With a generic transcriptome input of "Species.fasta", these files will have the following format:

- Species.fasta.fnn
  - Nucleotide fasta formatted frame selected sequences
- Species.fasta.faa
  - Amino acid fasta formatted frame selected sequences
- Species.fasta.lst
  - Information on each sequence (partial/internal/complete/ORF length)
- .err and .out file
  - These files are will contain any error or general information produced from the GeneMarkS-T run

8.5.3 EnTAP Files: */processed*

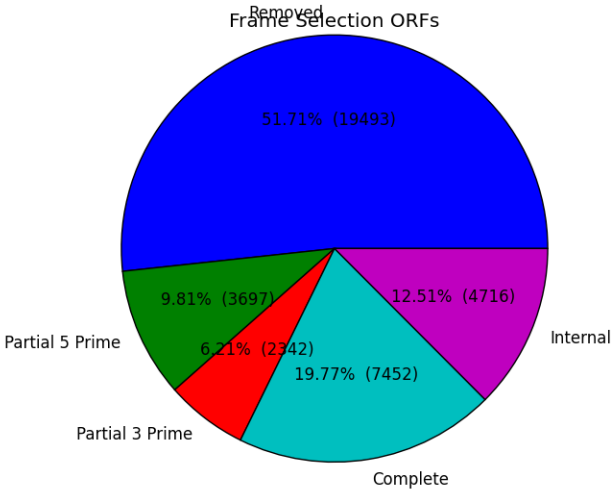
Files within the */processed* are generated by EnTAP and will contain ORF information based on the GeneMarkS-T execution.

- complete_genes.fasta
  - Amino acid sequences of complete genes from transcriptome
- partial_genes.fasta
  - Amino acid sequences of partial (5' and 3') sequences
- internal_genes.fasta
  - Amino acid sequences of internal sequences
- sequences_lost.fasta
  - Nucleotide sequences in which a frame was not found. These will not continue to the next stages of the pipeline

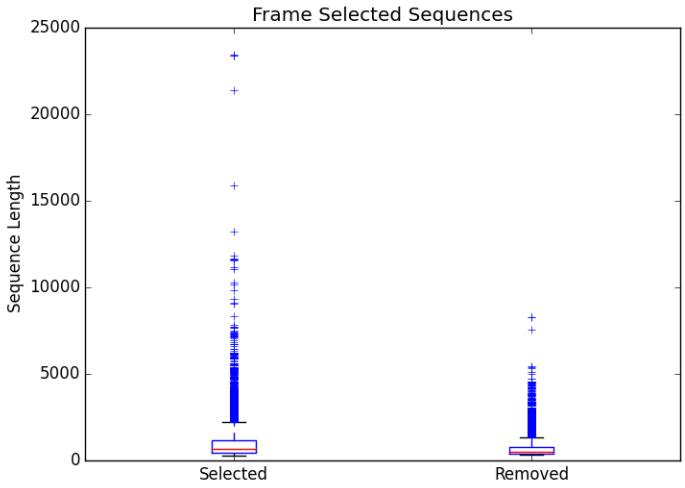
8.5.4 EnTAP Files: */figures*

In addition to files, EnTAP will generate figures within the */figures* directory. These are some useful visualizations of the information provided by GeneMarkS-T

- frame_results_pie.png
  - Pie chart representing the transcriptome (post expression filtering) showing complete/internal/partial/and sequences in which a frame was not found



- frame_selected_seq.png
  - Box plot of sequence length vs. the sequences that were lost during frame selection and the sequences in which a frame was found



8.6 Similarity Search (DIAMOND)

The */similarity_search* directory will contain all of the relevant information for the similarity searching stage of the pipeline. This folder will contain the *main files* (results from similarity search software), *files* analyzing hits from each database, *overall* results combining the information from each database, and *figures* generated from EnTAP.

8.6.1 DIAMOND Files: */similarity_search*

The files within the */similarity_search* directory contain the results from the similarity searching portion of the pipeline against each database you select. More information can be found at DIAMOND. With running _blastp (protein

similarity searching), a generic transcriptome input of "Species.fasta", with a database called "database" the files will have the following format:

- blastp_Species_database.out
  - This contains the similarity search information provided in the format from DIAMOND
  - Header information (from left to right):
    - Query Sequence ID
    - Subject Sequence ID
    - Percentage of Identical Matches
    - Alignment Length
    - Number of Mismatches
    - Number of gap openings
    - Start of alignment in query
    - End of alignment in query
    - Start of alignment in subject
    - End of alignment in subject
    - Expect (e) value
    - Bit score
    - Query Coverage
    - Subject Title (pulled from database)
- blastp_Species_database_std.err and .out
  - These files are will contain any error or general information produced from DIAMOND

8.6.2 EnTAP Files: */processed*

Files within the */processed* are generated by EnTAP and will contain information based on the hits returned from similarity searching against each database. This information contains the *best hits* (discussed previously) from each database based on e-value, coverage, informativeness, phylogenetic closeness, and contaminant status.

The files below represent a run with the same parameters as the section above:

- All the TSV files mentioned in this section will have the same header as follows (from left to right):
  - Query sequence ID
  - Subject sequence ID
  - Percentage of identical matches
  - Alignment length
  - Number of mismatches
  - Number of gap openings
  - Start of alignment in query
  - End of alignment in query
  - Start of alignment in subject

  - End of alignment in subject
  - Expect (e) value
  - Query coverage
  - Subject title
  - Species (pulled from hit)
  - Origin Database
  - ORF (taken from frame selection stage)
  - Contaminant (yes/no the hit was flagged as a contaminant)
- database/best_hits.faa and .fnn and .tsv
  - Best hits (protein and nucleotide) that were selected from this database
  - This contains ALL best hits, including any contaminants that were found as well as uninformative hits
  - The .tsv file contains the header information mentioned above of these same sequences
  - Note: Protein or nucleotide information may not be available to report depending on your type of run (these files will be empty)
- database/best_hits_contam.faa/.fnn/.tsv
  - Contaminants (protein/nucleotide) separated from the best hits file. As such, these contaminants will also be in the _best_hits.faa/.fnn.tsv files
- database/best_hits_no_contam.faa/.fnn/.tsv
  - Sequences (protein/nucleotide) that were selected as best hits and not flagged as contaminants
  - With this in mind: best_hits = best_hits_no_contam + best_hits_contam
  - These sequences are separated from the rest for convenience if you would like to examine them differently
- database/no_hits.faa/.fnn/.tsv
  - Sequences (protein/nucleotide) from the transcriptome that did not hit against this particular database.
  - This does not include sequences that were lost during expression filtering or frame selection
- database/unselected.tsv
  - Similarity searching can result in several hits for each query sequence. With only one best hit being selected, the rest are unselected and end up here
  - Unselected hits can be due to a low e-value, coverage, or other properties EnTAP takes into account when selecting hits

8.6.3 EnTAP Files: */overall_results*

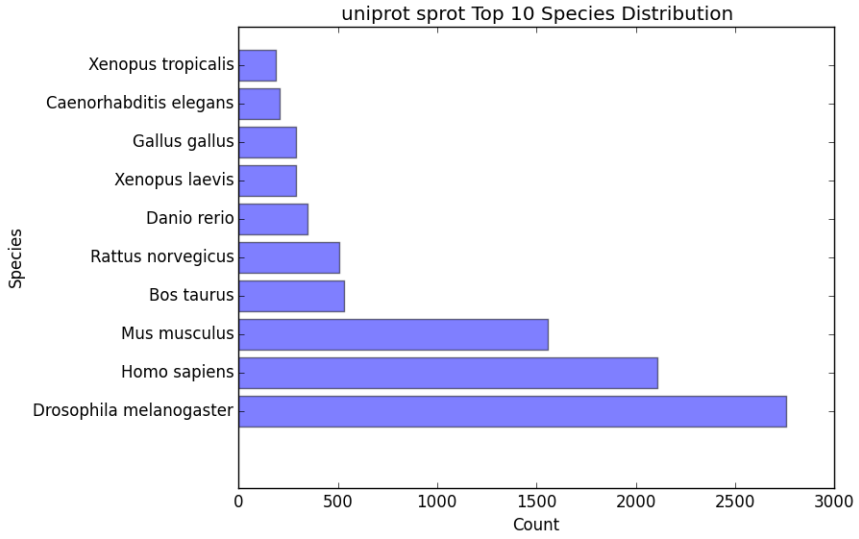
While the */processed* directory contains the best hit information from each database, the */overall_results* directory contains the overall best hits combining the hits from each database.

8.6.4 EnTAP Files: */figures*

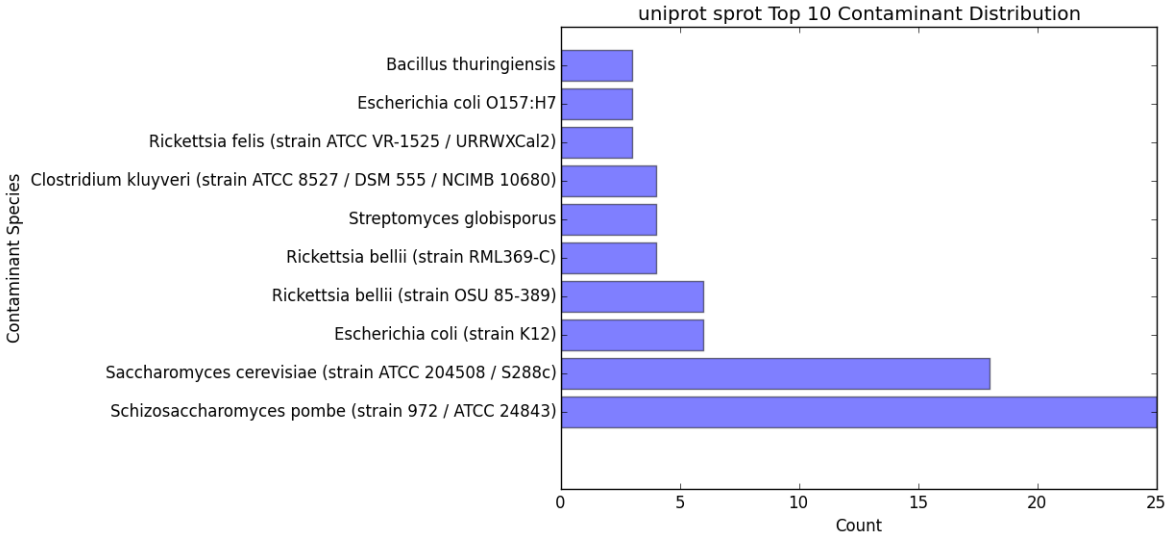
In addition to files, EnTAP will generate figures within the */figures* directory for each database. These are some useful visualizations of the information provided by similarity searching.

Here, there will be several figures:

- species_bar.png / species_bar.txt
  - Bar graph representing the top 10 species that were hit within a database
  - Text file representing the data being displayed



- contam_bar.png / contam_bar.txt
  - Bar graph representing the top 10 contaminants (within best hits) that were hit against the databast
  - Text file representing the data being displayed



8.7 Orthologous Groups/Ontology (EggNOG)

The */ontology/EggNOG* directory will contain all of the relevant information for the EggNOG stage of the pipeline. This folder will contain the *EggNOG files*, *files* analyzing the annotation from EggNOG, and *figures* generated from EnTAP.

8.7.1 EggNOG Files: */ontology/EggNOG*

Files within the */ontology/EggNOG* are generated through DIAMOND alignment against the EggNOG orthologous database and will contain information based on the hits returned. More information can be found at EggNOG.

- blastp_transcriptome_eggnog_proteins.out
  - EggNOG results for sequences from the final transcriptome being used (post-processing)

8.7.2 EnTAP Files: */processed*

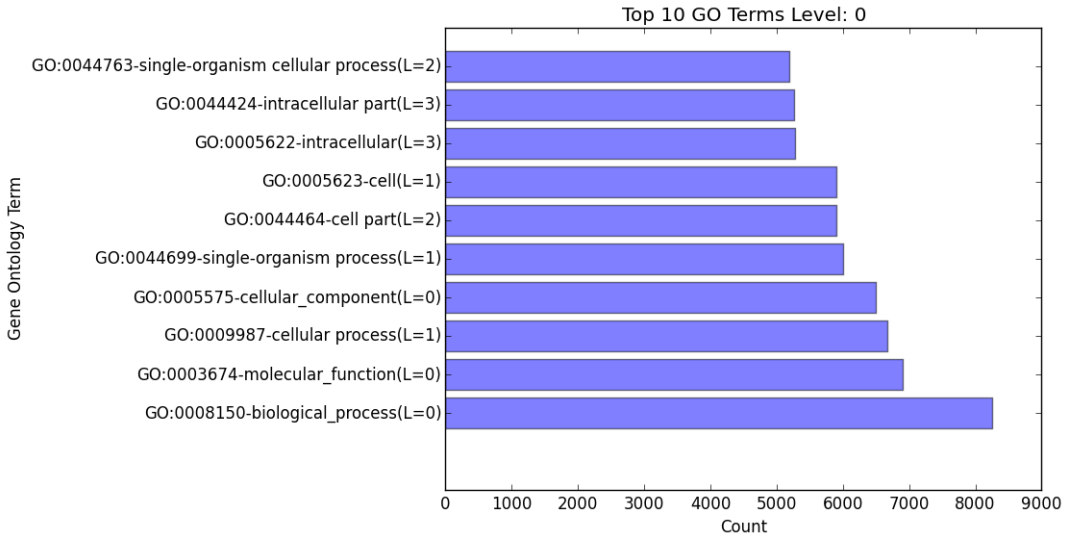
Files within the */processed* are generated by EnTAP and contain information on what sequences were annotated and which were not.

- unannotated_sequences.fnn/faa
  - Sequences where no gene family could be assigned (nucleotide/protein)
- annotated_sequences.fnn/faa
  - Sequences where a gene family could be assigned (nucleotide/protein)

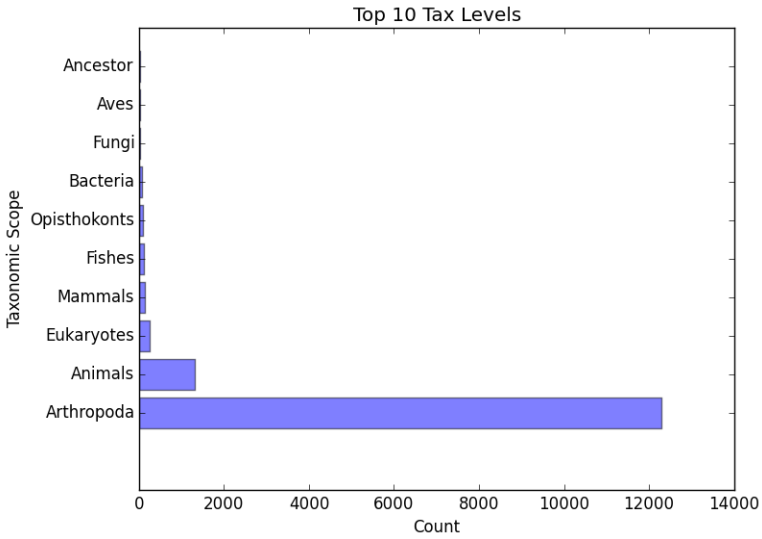
8.7.3 EnTAP Files: */figures*

The */figures* will contain figures generated by EnTAP of Gene Ontology and Taxonomic distribution of the results

- (overall/molecular_function/cellular_component/biological_process)#_go_bar_graph.png/.txt
  - Bar graph of each category of Gene Ontology terms of a specific level # (remember, level 0 signifies all levels!)



- eggnog_tax_scope.png/.txt
  - A bar graph representation of the taxonomic scope of the gene families assigned through EggNOG



8.8 Protein Families (InterProScan)

The */ontology/InterProScan* directory will contain all of the relevant information for the optional InterProScan stage of the pipeline. This folder will contain the *InterProScan files* and *files* generated from EnTAP analyzing the annotation from InterProScan.

8.8.1 InterProScan Files: */ontology/InterProScan*

Files within the */ontology/InterProScan* are generated through InterProScan and will contain information based on the results from the InterPro databases. More information can be found at InterProScan.

- interproscan.tsv/xml
  - Tab delimited or XML file containing information on the sequences with domain matches. Information such as signature accession/description information and GO/Pathway alignments.

8.8.2 EnTAP Files: */processed*

Files within the */processed* are generated by EnTAP and contain information on what sequences had domain matches, and which did not.

- unannotated_sequences.fnn/faa
  - Sequences where no domain could be assigned (nucleotide/protein)
- annotated_sequences.fnn/faa
  - Sequences where a no domain could be assigned (nucleotide/protein)

CHAPTER 9

Troubleshooting

This page contains several useful ways to troubleshoot whenever you run into an issue with EnTAP execution or installation. If you do not come to a solution through checking inputs, paths, or the configuration file I'd review these in the order I am laying them out.

1. *Debug File* : EnTAP will pipe any errors to the Debug File printed after every execution as well as the 'standard' output. Depending on your system, the 'standard' output may be placed into separate files or printed to the terminal
2. *Error Codes* : Provide specific information on where your run failed within the EnTAP pipeline. This could be during execution or installation. These can be as general as 'DIAMOND has failed,' or much more detailed. They will be more detailed if the failure was detected by EnTAP rather than accompanying software.
3. *.err and .out Files* : Provide information from the software that has failed. These are essentially what the software utilized by EnTAP (ex: DIAMOND) would print out to the user if an error occurs. Thus, they are out of EnTAP's control and may provide some further insight into the issue.
4. *FAQs* : If you're still running into trouble, check out these common issues!

9.1 Debug File

The Debug File contains detailed information of the EnTAP run. The more relevant information will reside at towards the bottom of the file. This will explicitly tell you what went wrong with the execution. As mentioned above, the pertinent error information will be printed to the terminal as well as to this file.

9.2 Error Codes

EnTAP has quite a few error codes that are generally going to have messages attached to them when your execution fails. I'll list the error codes and possible messages/troubleshooting ideas here. This is a continually evolving section where additional codes will be added as they are created (and I find the time to update). However, pre-existing codes will NOT be changed!

0. No error!
10. There is some error in your inputs to EnTAP. This is most likely caused by your command arguments. Make sure to check these out as well as any relevant paths. EnTAP should provide the specific error attached to this code as to why the run failed.
12. Some error in parsing the configuration file attached to EnTAP. Make sure it is following the proper format as when it was first downloaded from the GitLab page. There should be no spaces or extra line breaks after the '=' for each parameter.
13. This error is called when the configuration file could not be generated. If EnTAP does not recognize a configuration file, it will attempt to generate one for you.
14. This error is called when a configuration file could not be found in either your input, or the execution directory for EnTAP. It will merely generate the file and exit execution to allow you to fill in any parameters you may want to include.
15. There was an issue in download the EnTAP Taxonomic Database. This will only be called when you are running the - -config command. Some solutions can be: check your internet connection, ensure you have the required Python libraries installed (it currently uses Python as a means of download), and try running this outside of EnTAP (with the commmand python tax_download.py -o output_file.txt). If you decide to download the text version separately, just be sure to include that in the entap_config.txt file and it will convert it to a binary when you run - - config again.

9.3 .err and .out Files

The .err and .out files contain all of the standard output from software utilized by EnTAP. Consider DIAMOND... EnTAP will utilize DIAMOND for Similarity Searching. This software has it's own set of error and output information. EnTAP will print these to the relevant .err and .out files within the proper directory (in this case similarity_search/DIAMOND). Again, this information will also be printed to the Debug File.

9.4 FAQs

This list will be continually updated with common problems experienced by users.

1. I am getting an error that a sequence header could not be found during Expression Analysis (RSEM).
    - EnTAP tries to eliminate any inconsistencies between software packages by trimming your sequence headers (i.e. ">trinity_3122_1") to the first space. Versions before 0.9.1 will not do this by default. However sometimes, especially when using BAM/SAM files, EnTAP may not be able to match one of the headers to your input transcriptome. If this is the case, try trimming the excess information beyond the first space then re-running EnTAP. An example sequence, ">trinity_3122_1 extra info", should be input as ">trinity_3122_1".

The following issues remain for historical purposes. They are not relevant after the release of EnTAP v0.9.0.

1. I'm having issues reading the EnTAP database!
    - First, ensure you are specifying the correct paths in the configuration file. Next, attempt to generate the database locally (through - - data-generate flag during configuration). This is likely the result of incompatibility in Boost versions. The SQL version should always be compatible with your system as well, although slower.
2. Eggnog-mapper is failing when running DIAMOND.

- Eggnog-mapper leverages DIAMOND to search against the EggNOG databases. In order to do this, it uses a global call to DIAMOND. This may not work if you do not have it installed globally. Additionally, the DIAMOND EggNOG database may not be compatible with your local DIAMOND version. If so, you'll have to re-index the fasta version of this database (found from the EggNOG FTP) with your version of DIAMOND.

3. **Eggnog-mapper is failing during parsing of the data due to "too_many_columns"**

    - This is due to an incompatibility between EnTAP and the version of Eggnog-mapper you are using. Please use the version provided in the EnTAP repository (0.7.4.1-beta). Additionally, replace the eggnog.db file you are using with the database at the following address: http://eggnogdb.embl.de/download/emapperdb-4.5.0/eggnog.db.gz. This incompability will be resolved shortly (if not already)

CHAPTER 10

Changelog

This page contains (mostly) all of the changes that were made between each version of EnTAP. The current latest version is EnTAP Beta v0.10.8-beta

10.1 EnTAP Beta v0.10.9-beta (June 29, 2023)

- **Optimizations made to similarity searching using the EnTAP SQL database. Should improve speed**
  - Added 'api-taxon' command that will verify whether an input taxon can be found in the taxonomy database. It will return json formatted text
  - Added additional messaging throughout EnTAP execution to stdout
  - Removed Test Data from repository, it is no longer compatible with latest version of software within EnTAP. Will add back an updated dataset next version
  - Changed DIAMOND command to use '–max-target-seqs' instead of '–top' command
  - Fixed an issue where duplicate sequences were printed to the final_annotations files
- Fixed an issue where the taxanomic species may not have been found when searching against the SQL EnTAP database

10.2 EnTAP Beta v0.10.8-beta (March 21, 2021)

- This version requires a new version of the EnTAP database to be downloaded
- Added Gene Enrichment files as an output option(gene ID + effective length and geneID + GO term). These can be seen with the output-type flag in the ini file
- Changed Gene Ontology level printing. 0 will continue to print every term. Other levels will now print that level AND higher. So a level of 1 will print 1, 2, 3, etc. Previous a level of 1 would only print GO Terms with a level of 1

- Changed 'uninformative' input from a file to a list of terms in the ini file. Much more straightforward this way
- If no alignments are found against a database during DIAMOND, the pipeline will no longer exit, it will continue to the next database. If no alignments are found against any databases, it will stop at that point
- Fixed a bug where TransDecoder output may not have been parsed correctly for some users. This presented itself as a parsing error and halted EnTAP at that stage of the pipeline
- Fixed bug where InterProScan Mobidlite database was giving an error for some users (and halting execution)

10.3 EnTAP Beta v0.10.7-beta (October 6, 2020)

- Fixed an issue where certain sequence headers may not have been parsed properly resulting in unrecognized sequence errors during Similarity Searching

10.4 EnTAP Beta v0.10.6-beta (August 26, 2020)

- Added support to pipe the TransDecoder flag '–no_refine_starts' during Execution
- Fixed an issue where error messages during EggNOG searching would not get printed (seg fault)
- Contaminant information will not be printed to the log if there are none

10.5 EnTAP Beta v0.10.5-beta (August 12, 2020)

- Added a step to remove the stop codon ('*') sometimes printed at the end of the TransDecoder FASTA output. This may have caused an issue when running TransDecoder and InterProScan together

10.6 EnTAP Beta v0.10.4-beta (July 29, 2020)

- Fixed an issue where expression analysis transcriptome generation would fail (error message presented to user as 'frame selection')

10.7 EnTAP Beta v0.10.3-beta (July 28, 2020)

- Fixed a parsing issue of user inputs for contanminants and taxon

10.8 EnTAP Beta v0.10.2-beta (July 26, 2020)

- Fixed a pathing issue when EnTAP generated frame selected transcriptomes

10.9 EnTAP Beta v0.10.1-beta (July 19, 2020)

Note: Please use v0.10.2-beta or later instead of this version

- Added support for TransDecoder for Frame Selection
- Added TPM as an additional output from Expression Filtering
- Added an .ini file and moved many commands/paths from the command line to this
- Standardized/finalized output header namings for gFACs support
- Changed the default Frame Selection software to TransDecoder. GeneMarkS-T can still be selected through the .ini file
- Changed the default Gene Ontology level to 1. This can be easily changed through the ini file
- Fixed issue where some EggNOG descriptions were not printed to the final output
- Fixed a few issues with older GCC versions
- Fixed an issue where GeneMarkS-T would write to the working directory

10.10 EnTAP Beta v0.9.2-beta (June 4, 2020)

- Updated EggNOG Database links

10.11 EnTAP Beta v0.9.1-beta (January 12, 2020)

- Changed –trim flag to –no-trim. Trimming sequence headers to the first space is the default now. If you have executions from previous versions, you may need to use the –no-trim flag as needed for backwards compatibility (picking up where you left off)
- Fixed a bug where the –single-end command was not properly recognized

10.12 EnTAP Beta v0.9.0-beta (May 12, 2019)

- This release focused on reducing installation complexity and removing dependencies
- Overhauled the configuration/execution process by removing EggNOG-mapper and replacing it with an internal EnTAP method. This will make installation and both stages much clearer for the user
- Removed Boost Libraries from dependencies further reducing installation complexity
- Added printing of error messages to the standard log from any software being used by EnTAP. This will make debugging much easier
- Added UniProt mapping to the EnTAP database. This will pull any additional mapping information from UniProt Swiss-Prot alignments
- Updated supported DIAMOND version to 0.9.9
- The EnTAP database MUST be re-configured for this release
- Resolved any incompatibility with DIAMOND and EggNOG databases as well as versioning problems
- Standardized EnTAP log entries and added additional statistics

- – -ontology flag will now use EnTAP's method of EggNOG accession (0) or InterProScan (1)
- Bug fixes

10.13 EnTAP Beta v0.8.4-beta (August 2, 2018)

- Fixed an issue when inputting already translated sequences

10.14 EnTAP Beta v0.8.3-beta (May 23, 2018)

- Minor bug fixes
- Changes to CMake to hopefully resolve issues a couple users had with linking to Boost Libraries

10.15 EnTAP Beta v0.8.2-beta (April 29, 2018)

- Revamped configuration stage of EnTAP (reduced time and hopefully made things clear/more compatible across systems)
- Removed - -database-out flag (seemed a bit redundant to me). - -outfiles flag will be the default when indexing databases
- Added - -data-generate flag. This can be specified in EnTAP config stage (no effect during execution) for whether you'd like to generate the EnTAP databases rather than downloading from FTP address
- Added - -data-type flag. This can be used in either configuration or execution. Specifies which database you'd like to download/generate or use during execution. Binary (0, default) or SQL (1). Binary is faster with more memory usage, SQL will be slower but easier compatibility.
- Combined EnTAP databases into one (entap_database.sql/entap_database.bin). WARNING: Re-download or configuration of databases is REQUIRED with this newer version.
- Removed download_tax.py script (no longer necessary)

10.16 EnTAP Beta v0.8.1-beta (April 14, 2018)

- Added additional error logging to provide more information when something goes wrong
- Configuration file mandatory (default place to look is current working directory)
- Changed tax database paths in config file to avoid confusion (separate text and bin). Config file must be re-downloaded/generated!
- Defaults/output during configuration changed to config file then if not found, database-out flag
- Added deletion of empty files if a certain stage failed (preventing re-reading an empty file)
- Added errors/warnings for no alignments/hits in each stage
- entap_out directory changed to transcriptomes to be more clear (holds only transcriptomic data)
- Final EnTAP output files moved from the root outfiles directory to final_results directory
- Several filename changes to add consistency in new transcriptomes directory (final transcriptome is now _final.fasta.

- Several title changes to the log file to mitigate confusion
- EggNOG no longer broken down into separate files - those that hit and those that did not hit a database. Now entire transcriptome is pushed with one output file
- 10 species/contaminants/other in similarity searching statistics has been changed to 20 to provide more information to the user
- Best hit selection state combined with similarity search
- Added 'N' as an accepted nucleotide
- Several behind the scenes changes
- Fixed Cmake global installation issue
- Fixed incorrect error codes
- Fixed InterPro printing bug to no hits/hits files
- Fixed Frame Selection not printing new lines for certain files

10.17 EnTAP Beta v0.8.0-beta (December 16, 2017)

- Overhaul of the taxonomic/gene ontology databases
  - Faster accession/indexing
  - MUST be re-downloaded and re-indexed (or use the updated versions that come with the EnTAP distribution)
  - Taxonomic database includes thousands more entries with synonyms of many species
  - Perl is no longer a dependency, with Python being used to download the database
- Added blastx support
  - Blastx now allowed for ALL stages of annotation (similarity search + ontology)
  - –runN flag now specifies blastx (frame selection will not be ran)
  - –runP flag now specifies blastp (frame selection will be performed if nucleotide sequences are input)
- Added InterProScan support
  - Now possible to run EggNOG and/or InterProScan (with both blastx or blastp)
  - EggNOG and/or InterProScan specified with –ontology flag (0 and/or 1)
  - Full output of both will be provided in the final annotations file
- Added additional statistics to the log file for EggNOG and Expression Analysis
- Added numerous file/path/software checks to the start of an EnTAP run
  - Test runs/path checks are performed on all software that will be ran
  - Additional checks to specific flags
  - These checks can be turned off for an EnTAP run with –no-check flag (not advised!)
- –tag flag changed to –out-dir to specify output directory (not just what you'd like it named as)
  - Defaults to current directory with /outfiles folder
- –paired-end flag for Expression Filtering changed to –single-end (with paired-end being the default)

- Added contaminant and informative yes/no columns in final annotations file (among other headers)
- Added ability to input your own list of informative/uninformative terms for EnTAP to flag
- Added contaminant and none contaminant final annotation files
- Fixed a sequence id issue in Expression Filtering not mapping to BAM/SAM file
- Fixed a bug in –trim flag for sequence headers
- Fixed a bug where some systems had issues with graphing
- Debug and log files are now time stamped and not overwritten
- Fixed pathing for EnTAP configuration and made more streamlined
- Fixed several instances of older compilers complaining
- Added a lot of error messaging to help diagnose any issues easily
- Changed similarity search to have full database name, not path
- Fixed a bug in parsing input fasta file (added corrupt file checks)

10.18 EnTAP Beta v0.7.4.1-beta (September 5, 2017)

- Minor changes to taxonomic database download and indexing

10.19 EnTAP Beta v0.7.4-beta (August 26, 2017)

- Initial beta release!

CHAPTER 11

Future Features / Roadmap

This page contains future features that have been requested by users with an accompanying rough roadmap for implementation. The dates outlined below are subject to change depending on resources/time. If you would like to request support for additional features please email Alexander Hart at entap.dev@gmail.com

A roadmap for the future is in progress and will be released next version!

Note: Make sure you specify the version you are using in the bottom left before reading!