miEAA

Release 0.3.0

Feb 24, 2021

Contents

The miRNA Enrichment Analysis and Annotation Tool (miEAA) facilitates the functional analysis of miRNA sets. This package provides a miEAA REST API wrapper and command line interface. As such, a stable internet connection is required to utilize these tools.

To learn more about miEAA or to utilize our online interface, please visit our web server.
All miEAA tools are provided and hosted by the Chair for Clinical Bioinformatics at Saarland University.
Source code is available on GitHub.
Documentation is available on Read the Docs.

Users can execute miEAA commands directly from the command line:

```
$ mieaa -h
```

A REST API is also provided for scripting purposes:

```
from mieaa import API

mieaa_api = API()
```

Complete examples for both Python and R (using reticulate) are available under *Examples*.

CHAPTER 1

Installation

Dependencies:

- Python >= 3.5
- Requests >= 2.19

1.1 Python Package Index

```
$ pip install mieaa
```

1.2 Conda

```
$ conda install -c ccb-sb mieaa
```

CHAPTER 2

API Wrapper

2.1 mieaa

API()	miEAA api wrapper class.
API.convert_mirbase(mirnas, Iterable[str], ...)	Convert a set of either miRNAs/precursors from one miRbase version to another
API.to_precursor(mirnas, Iterable[str], IO], ...)	Convert from mirna->precursor
API.to_mirna(mirnas, Iterable[str], IO], ...)	Convert from precursor->mirna
API.get_enrichment_categories(mirna_type, ...)	Get possible enrichment categories
API.get_enrichment_parameters()	Retrieve parameters used during enrichment analysis
API.get_gui_url(page[, job_id])	Get specific url to page in web tool
API.get_progress()	Retrieve enrichment analysis progress
API.get_results(results_format, ... [, retries])	Return results in json or csv format
API.new_session()	Start a new session, clearing all job results
API.open_gui([page, job_id])	Open specific mieaa web tool page in browser
API.run_gsea(test_set, Iterable[T_co], IO], ...)	Start miRNA Set Enrichment Analysis
API.run_ora(test_set, Iterable[T_co], IO], ...)	Start Over Enrichment Analysis
API.save_enrichment_results(save_file, IO], ...)	Save results in specified format

2.1.1 mieaa.API

class mieaa.**API**

miEAA api wrapper class. Each instance is tied to a Job ID after starting an enrichment analysis. Instance must be reset with *new_session()* before starting a new analysis.

root_url [class attribute]

API root url

Type str

api_version [class attribute]
Current API version

Type str

endpoints [class attribute]
API endpoints suffix

Type dict

wait_between_requests [class attribute]
How many seconds to wait between API requests (due to throttling)

Type float

default [class attribute]
Default settings to pass to converter or analysis apis

Type dict

jobs [class attribute]
Access jobs run by any API instances, keys are job_id and values are enrichment parameters

Type dict

session [instance attribute]
Session information necessary to retrieve results

Type API_Session

job_id [instance attribute]
Unique identifier for enrichment analysis job of current session

Type uuid

__init__()
Initialize self. See help(type(self)) for accurate signature.

Methods

__init__()	Initialize self.
convert_mirbase(mirnas, Iterable[str], IO], ...)	Convert a set of either miRNAs/precursors from one miRbase version to another
get_enrichment_categories(mirna_type, species)	Get possible enrichment categories
get_enrichment_parameters()	Retrieve parameters used during enrichment analysis
get_gui_url(page[, job_id])	Get specific url to page in web tool
get_gui_urls([job_id])	Retrieve important mieaa webtool urls
get_progress()	Retrieve enrichment analysis progress
get_results(results_format, ... [, retries])	Return results in json or csv format
invalidate()	Invalidate current session.
load_job(job_id)	
new_session()	Start a new session, clearing all job results
open_gui([page, job_id])	Open specific mieaa web tool page in browser
run_gsea(test_set, Iterable[T_co], IO], ...)	Start miRNA Set Enrichment Analysis
run_ora(test_set, Iterable[T_co], IO], ...)	Start Over Enrichment Analysis
save_enrichment_results(save_file, IO], ...)	Save results in specified format

Continued on next page

Table 2 – continued from previous page

to_mirna(mirnas, Iterable[str], IO], ...)	Convert from precursor->mirna
to_precursor(mirnas, Iterable[str], IO], ...)	Convert from mirna->precursor

Attributes

api_version
default_params
endpoints
jobs
root_url
wait_between_requests

2.1.2 mieaa.API.convert_mirbase

API.**convert_mirbase**(*mirnas: Union[str, Iterable[str], IO], from_version: float, to_version: float, mirna_type: str, to_file: Union[str, IO] = '', **kwargs*) → List[str]

Convert a set of either miRNAs/precursors from one miRbase version to another

Parameters

- **mirnas** (*str or iterable*) – Iterable or delimited string of miRNAs, e.g. 'hsa-miR-199a-5p,hsa-mir-550b-1;'
- **from_version** (*float*) – MiRbase version to convert 'mirnas' from.
- **to_version** (*float*) – MiRbase version to update 'mirnas' to.
- **mirna_type** (*str*) – miRNAs/precursors to convert * *precursor* - Precursor to a mature miRNA, e.g. hsa-mir-550b-1 * *mirna* - Mature miRNA, e.g. hsa-miR-199a-5p Mixed input is not currently supported.
- **to_file** (*str or file-type, optional*) – if non-empty, save results to provided file name/path
- ****kwargs** –

  output_format (str, default='oneline')

  - *oneline* - Text containing only converted ids
  - *tabsep* - Tab-separated input and output id

Returns Converted miRNAs

Return type list

2.1.3 mieaa.API.to_precursor

API.**to_precursor**(*mirnas: Union[str, Iterable[str], IO], to_file: Union[str, IO] = '', **kwargs*) → List[str]

Convert from mirna->precursor

Parameters

- **mirnas** (*str or iterable*) – Iterable or delimited string of miRNAs, e.g. 'hsa-miR-199a-5p,hsa-mir-550b-1'
- **to_file** (*str, optional*) – if non-empty, save results to provided file name/path

- ****kwargs** –

  output_format (str, default='oneline')

  - *oneline* - Text containing only converted ids, multi-mapped are semicolon separated
  - *newline* - Text containing only converted ids, multi-mapped are newline separated
  - *tabsep* - Tab-separated input and output id

  conversion_type (str, default='all')

  - *all* - Output all mappings
  - *unique* - Only output unique mappings

Returns Converted miRNAs

Return type list

2.1.4 mieaa.API.to_mirna

API.**to_mirna**(*mirnas: Union[str, Iterable[str], IO], to_file: Union[str, IO] = '', **kwargs*) → List[str]

Convert from precursor->mirna

Parameters

- **mirnas** (*str or iterable*) – Iterable or delimited string of miRNAs, e.g. 'hsa-miR-199a-5p,hsa-mir-550b-1'
- **to_file** (*str, optional*) – if non-empty, save results to provided file name/path
- ****kwargs** –

  output_format (str, default='oneline')

  - *oneline* - Text containing only converted ids, multi-mapped are semicolon separated
  - *newline* - Text containing only converted ids, multi-mapped are newline separated
  - *tabsep* - Tab-separated input and output id

  conversion_type (str, default='all')

  - *all* - Output all mappings
  - *unique* - Only output unique mappings

Returns Converted miRNAs

Return type list

2.1.5 mieaa.API.get_enrichment_categories

API.**get_enrichment_categories**(*mirna_type: str, species: str, mode='all', with_suffix=False*) → dict

Get possible enrichment categories

Parameters

- **mirna_type** (*str*) –
  - *precursor* - Precursor to a mature miRNA, e.g. hsa-mir-550b-1
  - *mirna* - Mature miRNA, e.g. hsa-miR-199a-5p

- **species** (*str*) –
  - *hsa* - Homo sapiens
  - *mmu* - Mus musculus
  - *rno* - Rattus norvegicus
  - *ath* - Arabidopsis thaliana
  - *bta* - Bos taurus
  - *cel* - Caenorhabditis elegans
  - *dme* - Drosophila melanogaster
  - *dre* - Danio rerio
  - *gga* - Gallus gallus
  - *ssc* - Sus scrofa
- **mode** (*str*) –
  - *all* - include both default and expert categories
  - *default* - only show default (non-expert) categories
  - *expert* - only show expert categories
- **with_suffix** (*bool, default=False*) – whether to include '_precursor' or '_mature' at end of category name

Returns Keys are categories and values are their descriptions

Return type dict

2.1.6 mieaa.API.get_enrichment_parameters

API.**get_enrichment_parameters**()
Retrieve parameters used during enrichment analysis

2.1.7 mieaa.API.get_gui_url

API.**get_gui_url**(*page*, *job_id=None*)
Get specific url to page in web tool

Parameters

- **page** (*str*) –
  - *input* - user input wizard
  - *progress* - job progress
  - *results* - job results
- **job_id** (*str, default=None*) – Use job id in url if applicable. Will try current job_id if none provided.

Returns URL to specified mieaa webtool page

Return type str

2.1.8 mieaa.API.get_progress

API.**get_progress**()
Retrieve enrichment analysis progress

2.1.9 mieaa.API.get_results

API.**get_results**(*results_format: str = 'json'*, *check_progress_interval: float = 5.0*, *retries=5*) → Union[str, list]
Return results in json or csv format

Parameters

- **results_format** (*str, default='json'*) –
  - *json* - retrieve results in json format
  - *csv* - retrieve results in csv format
- **check_progress_interval** (*float*) – How many seconds to wait between checking if results have been computed

Returns Response

Return type requests.Response

2.1.10 mieaa.API.new_session

API.**new_session**()
Start a new session, clearing all job results

2.1.11 mieaa.API.open_gui

API.**open_gui**(*page='input'*, *job_id=None*)
Open specific mieaa web tool page in browser

Parameters

- **page** (*str, default='input'*) –
- **input - user input wizard** (***) –
- **progress - job progress** (***) –
- **results - job results** (***) –

2.1.12 mieaa.API.run_gsea

API.**run_gsea**(*test_set: Union[str, Iterable[T_co], IO]*, *categories: Iterable[T_co]*, *mirna_type: str*, *species: str*, ***kwargs*)
Start miRNA Set Enrichment Analysis

Parameters

- **test_set** (*str, iterable or file-like*) – set of miRNAs/precursors we want to test

- **categories** (*str, iterable or file-like*) – Categories we want to run analysis on
- **mirna_type** (*str*) –
  - *precursor* - Precursor to a mature miRNA, e.g. hsa-mir-550b-1
  - *mirna* - Mature miRNA, e.g. hsa-miR-199a-5p
- **species** (*str*) –
  - *hsa* - Homo sapiens
  - *mmu* - Mus musculus
  - *rno* - Rattus norvegicus
  - *ath* - Arabidopsis thaliana
  - *bta* - Bos taurus
  - *cel* - Caenorhabditis elegans
  - *dme* - Drosophila melanogaster
  - *dre* - Danio rerio
  - *gga* - Gallus gallus
  - *ssc* - Sus scrofa
- ****kwargs** –

  p_value_adjustment (str, default='fdr')
  - *none* - No adjustment
  - *fdr* - FDR (Benjamini-Hochberg) adjustment
  - *bonferroni* - Bonferroni adjustment
  - *BY* - Benjamini-Yekutieli adjustment
  - *hochberg* - Hochberg adjustment
  - *holm* - Holm adjustment
  - *hommel* - Hommel adjustment

  independent_p_adjust (bool, default=True)
  - *True* - Adjust p-values for each category independently
  - *False* - Adjust p-values for all categories collectively

  significance_level (float, default=0.05) Filter out p-values above significance level

  threshold_level (int, default=2) Filter out subcategories that contain less than this many miRNAs

Returns Response

Return type requests.Response

2.1.13 mieaa.API.run_ora

API.**run_ora**(*test_set: Union[str, Iterable[T_co], IO], categories: Iterable[T_co], mirna_type: str, species: str, reference_set: Union[str, io.IOBase] = '', **kwargs*)

Start Over Enrichment Analysis

Parameters

- **test_set** (*str, iterable or file-like*) – set of miRNAs/precursors we want to test
- **categories** (*str, iterable or file-like*) – Categories we want to run analysis on
- **mirna_type** (*str*) –
  - *precursor* - Precursor to a mature miRNA, e.g. hsa-mir-550b-1
  - *mirna* - Mature miRNA, e.g. hsa-miR-199a-5p
- **species** (*str*) –
  - *hsa* - Homo sapiens
  - *mmu* - Mus musculus
  - *rno* - Rattus norvegicus
  - *ath* - Arabidopsis thaliana
  - *bta* - Bos taurus
  - *cel* - Caenorhabditis elegans
  - *dme* - Drosophila melanogaster
  - *dre* - Danio rerio
  - *gga* - Gallus gallus
  - *ssc* - Sus scrofa
- **reference_set** (*str or file-like, default=''*) – ORA specific, background reference set of miRNAs/precursors
- ****kwargs** –

  p_value_adjustment (str, default='fdr')
  - *none* - No adjustment
  - *fdr* - FDR (Benjamini-Hochberg) adjustment
  - *bonferroni* - Bonferroni adjustment
  - *BY* - Benjamini-Yekutieli adjustment
  - *hochberg* - Hochberg adjustment
  - *holm* - Holm adjustment
  - *hommel* - Hommel adjustment

  independent_p_adjust (bool, default=True)
  - *True* - Adjust p-values for each category independently
  - *False* - Adjust p-values for all categories collectively

  significance_level (float, default=0.05) Filter out p-values above significance level

threshold_level (int, default=2) Filter out subcategories that contain less than this many miRNAs

Returns Response

Return type requests.Response

2.1.14 mieaa.API.save_enrichment_results

`API.`**`save_enrichment_results`**(*save_file: Union[str, IO], file_type: str = 'csv', check_progress_interval: float = 5.0*) → str

Save results in specified format

Parameters

- **`save_file`** (*`str or file-like`*) – File to save results in
- **`file_type`** (*`str, default='csv'`*) – Type of file to write results to. Options are *json* or *csv*
- **`check_progress_interval`** (*`float, default=5`*) – How many seconds to wait between checking if results have been computed

2.2 Examples

2.2.1 Python

Python API Example

A barebones example script can be found on Github.

```
from mieaa import API

mieaa_api = API()
```

Target sets, categories, and reference sets support strings, iterables, and file objects.

Mixed delimiters should still function, but are not recommended.

```
#initial_mirnas = ['hsa-miR-374c', 'hsa-miR-642b', 'hsa-miR-550b', 'hsa-miR-107',
↪'hsa-miR-125b']
initial_mirnas = 'hsa-miR-374c hsa-miR-642b,hsa-miR-550b;hsa-miR-107;hsa-miR-125b'
```

Convert between miRBase versions

Results can be optionally saved to a file by specifying the `to_file` argument.

```
# mieaa_api.convert_mirbase(initial_precursors, 9.1, 22, 'precursor', to_file='mirnas.
↪txt')
updated_mirnas = mieaa_api.convert_mirbase(initial_mirnas, 16, 22, 'mirna')
updated_mirnas
```

```
['hsa-miR-374c-5p',
 'hsa-miR-642b-3p',
 'hsa-miR-550b-3p',
 'hsa-miR-107',
 'hsa-miR-125b-5p']
```

Convert between miRNAs <-> precursors

Results can be optionally saved to a file by specifying the `to_file` argument.

Some names are not uniquely converted. We can specify conversion type as either `all` (default) or `unique`.

We can also decide whether we want our output to only include converted results with multiple-mapped values separated by a semicolon (default, `oneline`), on their own individual lines (`newline`), or a tab separated input - output (`tabsep`).

```
precursors = mieaa_api.to_precursor(updated_mirnas, to_file='./precursors.txt',␣
↪conversion_type='all')
precursors
```

```
['hsa-mir-374c',
 'hsa-mir-642b',
 'hsa-mir-550b-1;hsa-mir-550b-2',
 'hsa-mir-107',
 'hsa-mir-125b-1;hsa-mir-125b-2']
```

```
with open('./precursors.txt') as prec_file:
    mirnas = mieaa_api.to_mirna(prec_file, output_format='tabsep')
mirnas
```

```
['hsa-mir-374c\thsa-miR-374c-5p;hsa-miR-374c-3p',
 'hsa-mir-642b\thsa-miR-642b-5p;hsa-miR-642b-3p',
 'hsa-mir-550b-1\thsa-miR-550b-3p;hsa-miR-550b-2-5p',
 'hsa-mir-550b-2\thsa-miR-550b-3p;hsa-miR-550b-2-5p',
 'hsa-mir-107\thsa-miR-107',
 'hsa-mir-125b-1\thsa-miR-125b-5p;hsa-miR-125b-1-3p',
 'hsa-mir-125b-2\thsa-miR-125b-5p;hsa-miR-125b-2-3p']
```

Enrichment Analysis

Starting Enrichment Analysis

Run Gene Set Enrichment Analysis (GSEA) or Over-representation Analysis (ORA).

Please refer to documentation for possible keyword arguments.

For ORA, if `reference_set` is not specified or is left empty, default to using miEAA reference sets for specified categories.

```
# mieaa_api.run_gsea(precursors, ['HMDD, mndr'], 'precursor', 'hsa')
with open('./precursors.txt', 'r') as test_set_file:
    mieaa_api.run_ora(test_set_file, ['HMDD, mndr'], 'precursor', 'hsa', reference_
↪set='')
```

Viewing computation progress

```
mieaa_api.get_progress()
```

```
0.7
```

Retrieving Enrichment Results

Get results after enrichment analysis has been completed, determining how often to check progress via `check_progress_interval` (default is 5 seconds).

```
json = mieaa_api.get_results(check_progress_interval=5)
```

The returned data can be easily turned into a pandas dataframe.

```
import pandas as pd
cols = ['category', 'subcategory', 'enrichment', 'p-value', 'p-adjusted', 'q-value',
↪'expected', 'observed', 'mirnas/precursors']
df = pd.DataFrame(json, columns=cols)
df.head()
```

category	subcategory	enrichment	p-value	p-adjusted	q-value	expected	observed	mirnas/precursors
Diseases (HMDD)	Alopecia	over-represented	0.0017138	0.0478120	0.0478120	1.0678879	2	hsa-mir-125b-1; hsa-mir-125b-2
Diseases (HMDD)	Atopic Dermatitis	over-represented	0.0021345	0.0478120	0.0478120	1.0754312	2	hsa-mir-125b-1; hsa-mir-125b-2
Diseases (HMDD)	Lichen Planus	over-represented	0.0017138	0.0478120	0.0478120	1.0678879	2	hsa-mir-125b-1; hsa-mir-125b-2
Diseases (HMDD)	Myotonic Muscular Dystrophy	over-represented	0.0006357	0.0356009	0.0356009	0.9196121	3	hsa-mir-107; hsa-mir-125b-1; hsa-mir-125b-2
Diseases (HMDD)	Nevus	over-represented	0.0001459	0.0163450	0.0163450	1.0226292	3	hsa-mir-125b-1; hsa-mir-125b-2

Results can also be obtained as a csv string.

```
csv_string = mieaa_api.get_results('csv')
```

Saving Enrichment Results

Results can be automatically saved to a json or csv (default) file.

```
# mieaa_api.save_enrichment_results('./example.json', file_type='json')
file_contents = mieaa_api.save_enrichment_results('./results.csv')
```

Alternatively, we can write the csv results to a file.

```
with open('results_2.csv', 'w+') as outfile:
    outfile.write(csv_string)
```

Miscellaneous

After running an analysis, we may wish to view the parameters we used for our analysis.

```
mieaa_api.get_enrichment_parameters()
```

```
{'enrichment_analysis': 'ORA',
 'p_value_adjustment': 'fdr',
 'independent_p_adjust': True,
 'significance_level': 0.05,
 'threshold_level': 2,
 'categories': ['HMDD_precursor', 'MNDR_precursor'],
 'reference_set': '',
 'testset_file': <_io.TextIOWrapper name='./precursors.txt' mode='r' encoding='UTF-8'>
↪}
```

Upon running an analysis, our API instance is assigned a unique Job ID.

If we wish to reuse the same instance to run a new analysis, we must create a new session.

```
mieaa_api.new_session()
```

2.2.2 R

R API Example

The reticulate library allows us to utilize the wrapper class in R, assuming Python is also installed.

A barebones example script can be found on Github.

```
library(reticulate)
mieaa = import("mieaa")
mieaa_api = mieaa$API()
```

Target sets, categories, and reference sets support strings, iterables, and file objects.

Mixed delimiters should still function, but are not recommended.

```
# list('hsa-miR-374c', 'hsa-miR-642b', 'hsa-miR-550b', 'hsa-miR-107', 'hsa-miR-125b')
initial_mirnas = 'hsa-miR-374c hsa-miR-642b,hsa-miR-550b;hsa-miR-107;hsa-miR-125b'
```

Convert between miRBase versions

Results can be optionally saved to a file by specifying the `to_file` argument.

```
# mieaa_api$convert_mirbase(initial_precursors, '9.1', '22', 'precursor', to_file=
↪'mirnas.txt')
updated_mirnas = mieaa_api$convert_mirbase(initial_mirnas, '16', '22', 'mirna')
updated_mirnas
```

```
[[1]]
[1] "hsa-miR-374c-5p"

[[2]]
```

(continues on next page)

(continued from previous page)

```
[1] "hsa-miR-642b-3p"

[[3]]
[1] "hsa-miR-550b-3p"

[[4]]
[1] "hsa-miR-107"

[[5]]
[1] "hsa-miR-125b-5p"
```

Convert between miRNAs <-> precursors

Results can be optionally saved to a file by specifying the `to_file` argument.

Some names are not uniquely converted. We can specify conversion type as either `all` (default) or `unique`.

We can also decide whether we want our output to only include converted results with multiple-mapped values separated by a semicolon (default, `oneline`), on their own individual lines (`newline`), or a tab separated input - output (`tabsep`).

```
precursors = mieaa_api$to_precursor(updated_mirnas, to_file='./precursors.txt',␣
↪conversion_type='all')
precursors
```

```
[[1]]
[1] "hsa-mir-374c"

[[2]]
[1] "hsa-mir-642b"

[[3]]
[1] "hsa-mir-550b-1;hsa-mir-550b-2"

[[4]]
[1] "hsa-mir-107"

[[5]]
[1] "hsa-mir-125b-1;hsa-mir-125b-2"
```

```
py = import_builtins()  # part of 'reticulate'

# Files need to be python file objects
with(py$open("precursors.txt", 'r') %as% prec_file, {
    mirnas = mieaa_api$to_mirna(prec_file, output_format='tabsep')
})
mirnas
```

```
[[1]]
[1] "hsa-mir-374c\thsa-miR-374c-5p;hsa-miR-374c-3p"

[[2]]
[1] "hsa-mir-642b\thsa-miR-642b-5p;hsa-miR-642b-3p"
```

(continues on next page)

(continued from previous page)

```
[[3]]
[1] "hsa-mir-550b-1\thsa-miR-550b-3p;hsa-miR-550b-2-5p"

[[4]]
[1] "hsa-mir-550b-2\thsa-miR-550b-3p;hsa-miR-550b-2-5p"

[[5]]
[1] "hsa-mir-107\thsa-miR-107"

[[6]]
[1] "hsa-mir-125b-1\thsa-miR-125b-5p;hsa-miR-125b-1-3p"

[[7]]
[1] "hsa-mir-125b-2\thsa-miR-125b-5p;hsa-miR-125b-2-3p"
```

Enrichment Analysis

Starting Enrichment Analysis

Run Gene Set Enrichment Analysis (GSEA) or Over-representation Analysis (ORA).

Please refer to documentation for possible keyword arguments.

For ORA, if `reference_set` is not specified or is left empty, default to using miEAA reference sets for specified categories.

```
# mieaa_api$run_gsea(precursors, ['HMDD, mndr'], 'precursor', 'hsa')
with(py$open("precursors.txt", 'r') %as% test_set_file, {
    mieaa_api$run_ora(test_set_file, list('HMDD, mndr'), 'precursor', 'hsa',␣
↪reference_set='')
})
```

Viewing computation progress

```
mieaa_api$get_progress()
```

0.7

Retrieving Enrichment Results

Get results after enrichment analysis has been completed, determining how often to check progress via `check_progress_interval` (default is 5 seconds).

```
json = mieaa_api$get_results(check_progress_interval=5)
```

The returned data can be easily turned into a dataframe.

```
cols = c('category', 'subcategory', 'enrichment', 'p-value', 'p-adjusted', 'q-value',
↪'expected', 'observed', 'mirnas/precursors')
df = data.frame(matrix(unlist(json), nrow=length(json), byrow=T))
colnames(df) = cols
head(df)
```

category	subcategory	enrichment	p-value	p-adjusted	q-value	expected	observed	mirnas/precursors
Diseases (HMDD)	Alopecia	over-represented	0.0017138	0.0478120	0.0478120	1.0678879	2	hsa-mir-125b-1; hsa-mir-125b-2
Diseases (HMDD)	Atopic Dermatitis	over-represented	0.0021345	0.0478120	0.0478120	1.0754312	2	hsa-mir-125b-1; hsa-mir-125b-2
Diseases (HMDD)	Lichen Planus	over-represented	0.0017138	0.0478120	0.0478120	1.0678879	2	hsa-mir-125b-1; hsa-mir-125b-2
Diseases (HMDD)	Myotonic Muscular Dystrophy	over-represented	0.0006357	0.0356009	0.0356009	0.9196121	3	hsa-mir-107; hsa-mir-125b-1; hsa-mir-125b-2
Diseases (HMDD)	Nevus	over-represented	0.0001459	0.0163450	0.0163450	1.0226292	3	hsa-mir-125b-1; hsa-mir-125b-2

Results can also be obtained as a csv string.

```
csv_string = mieaa_api$get_results('csv')
```

Saving Enrichment Results

Results can be automatically saved to a json or csv (default) file.

```
# mieaa_api$save_enrichment_results('./example.json', file_type='json')
file_contents = mieaa_api$save_enrichment_results('./results.csv')
```

Alternatively, we can write the csv results to a file.

```
outfile = file("./results_2.csv")
writeLines(csv_string, outfile)
close(outfile)
```

Miscellaneous

After running an analysis, we may wish to view the parameters we used for our analysis.

```
mieaa_api$get_enrichment_parameters()
```

```
$enrichment_analysis
[1] "ORA"

$p_value_adjustment
[1] "fdr"

$independent_p_adjust
[1] TRUE

$significance_level
[1] 0.05

$threshold_level
[1] 2

$categories
```

(continues on next page)

(continued from previous page)

```
[1] "HMDD_precursor" "MNDR_precursor"

$reference_set
[1] ""

$testset_file
<_io.TextIOWrapper name='precursors.txt' mode='r' encoding='UTF-8'>
```

Upon running an analysis, our API instance is assigned a unique Job ID.

If we wish to reuse the same instance to run a new analysis, we must create a new session.

```
mieaa_api$new_session()
```

CHAPTER 3

Command Line Interface

Commands can be invoked via the command line using `mieaa SUBCOMMAND`.Help and options for all subcommands can be view with `mieaa SUBCOMMAND -h`

Supported Species

- *hsa* - Homo sapiens
- *mmu* - Mus musculus
- *rno* - Rattus norvegicus
- *ath* - Arabidopsis thaliana
- *bta* - Bos taurus
- *cel* - Caenorhabditis elegans
- *dme* - Drosophila melanogaster
- *dre* - Danio rerio
- *gga* - Gallus gallus
- *ssc* - Sus scrofa

Specifying precursors For subcommands where you need to specify precursor or mature, mature is always assumed unless the `--precursor` (`-p`) flag is set.

Mutually exclusive options Most subcommands require one of `--mirna-set` (`-m`) or `--mirna-set-file` (`-M`) to be specified.

- `mieaa SUBCOMMAND --mirna-set MIRNA [MIRNA ...]`
- `mieaa SUBCOMMAND --mirna-set MIRNAS_STRING`
- `mieaa SUBCOMMAND --mirna-set-file MIRNA_FILE`

3.1 Subcommands

3.1.1 to_precursor

Convert miRNA -> precursor

```
usage: miEAA to_precursor [-h] [-v] [-m MIRNA_SET [MIRNA_SET ...]]
                          [-M MIRNA_SET_FILE] [-p] [-o OUTFILE]
                          [--oneline | --newline | --tabsep] [-u]

optional arguments:
  -h, --help            show this help message and exit
  -v, --verbose         Always print results to stdout
  -p, --precursor, --precursors
                        Use if running on a set of precursors as opposed to
                        miRNAs
  -o OUTFILE, --outfile OUTFILE
                        Save results to provided file
  --oneline             Output style: Multi-mapped ids are separated by a
                        semicolon (default)
  --newline             Output style: Multi-mapped ids are separated by a
                        newline
  --tabsep              Output style: Tab-separated `original converted` ids
  -u, --unique          Only output ids that map uniquely

mutually exclusive required arguments:
  either a set or file must be provided

  -m MIRNA_SET [MIRNA_SET ...], --mirna-set MIRNA_SET [MIRNA_SET ...]
                        miRNA/precursor target set
  -M MIRNA_SET_FILE, --mirna-set-file MIRNA_SET_FILE
                        Specify miRNA/precursor target set via file
```

Examples:

```
$ mieaa to_precursor -m hsa-miR-20b-5p hsa-miR-144-5p --tabsep --unique
$ mieaa to_precursor -m 'hsa-miR-20b-5p,hsa-miR-144-5p' --newline -o precursors.txt
$ mieaa to_precursor -M mirnas.txt --outfile precursors.txt
```

3.1.2 to_mirna

Converting between precursor -> miRNA

```
usage: miEAA to_mirna [-h] [-v] [-m MIRNA_SET [MIRNA_SET ...]]
                      [-M MIRNA_SET_FILE] [-p] [-o OUTFILE]
                      [--oneline | --newline | --tabsep] [-u]

optional arguments:
  -h, --help            show this help message and exit
  -v, --verbose         Always print results to stdout
  -p, --precursor, --precursors
                        Use if running on a set of precursors as opposed to
                        miRNAs
  -o OUTFILE, --outfile OUTFILE
                        Save results to provided file
```

(continues on next page)

(continued from previous page)

```
  --oneline             Output style: Multi-mapped ids are separated by a
                        semicolon (default)
  --newline             Output style: Multi-mapped ids are separated by a
                        newline
  --tabsep              Output style: Tab-separated `original converted` ids
  -u, --unique          Only output ids that map uniquely

mutually exclusive required arguments:
  either a set or file must be provided

  -m MIRNA_SET [MIRNA_SET ...], --mirna-set MIRNA_SET [MIRNA_SET ...]
                        miRNA/precursor target set
  -M MIRNA_SET_FILE, --mirna-set-file MIRNA_SET_FILE
                        Specify miRNA/precursor target set via file
```

Examples:

```
$ mieaa to_mirna -m hsa-mir-20b hsa-mir-144 --tabsep --unique
$ mieaa to_mirna -m 'hsa-mir-20b,hsa-mir-144' --newline -o mirnas.txt
$ mieaa to_mirna -M precursors.txt --outfile mirnas.txt
```

3.1.3 convert_mirbase

Converting miRBase version

```
usage: miEAA convert_mirbase [-h] [-v] [-m MIRNA_SET [MIRNA_SET ...]]
                             [-M MIRNA_SET_FILE] [-p] [-o OUTFILE]
                             [--oneline | --newline | --tabsep] [--to TO]
                             FROM

positional arguments:
  FROM                  mirBase version to convert miRNAs/precursors from

optional arguments:
  -h, --help            show this help message and exit
  -v, --verbose         Always print results to stdout
  -p, --precursor, --precursors
                        Use if running on a set of precursors as opposed to
                        miRNAs
  -o OUTFILE, --outfile OUTFILE
                        Save results to provided file
  --oneline             Output style: Multi-mapped ids are separated by a
                        semicolon (default)
  --newline             Output style: Multi-mapped ids are separated by a
                        newline
  --tabsep              Output style: Tab-separated `original converted` ids
  --to TO               mirBase version to convert miRNAs/precursors from
                        (default=22)

mutually exclusive required arguments:
  either a set or file must be provided

  -m MIRNA_SET [MIRNA_SET ...], --mirna-set MIRNA_SET [MIRNA_SET ...]
                        miRNA/precursor target set
```

(continues on next page)

(continued from previous page)

```
  -M MIRNA_SET_FILE, --mirna-set-file MIRNA_SET_FILE
                        Specify miRNA/precursor target set via file
```

Examples:

```
$ mieaa convert_mirbase 16 -m hsa-miR-642b,hsa-miR-550b
$ mieaa convert_mirbase 16 --to 22 -m hsa-miR-642b hsa-miR-550b
$ mieaa convert_mirbase 16  -M version_16.txt -o version_22.txt
```

3.1.4 gsea

Gene Set Enrichment Analysis (GSEA)

```
usage: miEAA gsea [-h] [-v] [-m MIRNA_SET [MIRNA_SET ...]] [-M MIRNA_SET_FILE]
                  [-p] [-o OUTFILE] [-x] [-c CATEGORIES [CATEGORIES ...]]
                  [-C CATEGORIES_FILE] [-t THRESHOLD] [-s SIGNIFICANCE] [-g]
                  [-a {none,fdr,bonferroni,BY,holm,hochberg,hommel}]
                  [--csv | --json]
                  {hsa,mmu,rno,ath,bta,cel,dme,dre,gga,ssc}

positional arguments:
  {hsa,mmu,rno,ath,bta,cel,dme,dre,gga,ssc}
                        Species

optional arguments:
  -h, --help            show this help message and exit
  -v, --verbose         Always print results to stdout
  -p, --precursor, --precursors
                        Use if running on a set of precursors as opposed to
                        miRNAs
  -o OUTFILE, --outfile OUTFILE
                        Save results to provided file
  -x, --no-results      Do not monitor progress or obtain results. Can
                        retrieve later using Job ID.
  -t THRESHOLD, --threshold THRESHOLD
                        Filter out subcategories that contain less than this
                        many miRNAs/precursors (default=2)
  -s SIGNIFICANCE, --significance SIGNIFICANCE, --alpha SIGNIFICANCE
                        Significance level (default=0.05)
  -g, --group-adjust    Adjust p-values over aggregated groups (By default
                        each group is adjusted independently)
  -a {none,fdr,bonferroni,BY,holm,hochberg,hommel}, --adjustment {none,fdr,bonferroni,
↪BY,holm,hochberg,hommel}
                        p-value adjustment method (default='fdr')
  --csv                 Store results in output file in csv format (default)
  --json                Store results in output file in json format (default
                        is csv)

mutually exclusive required arguments:
  either a set or file must be provided

  -m MIRNA_SET [MIRNA_SET ...], --mirna-set MIRNA_SET [MIRNA_SET ...]
                        miRNA/precursor target set
  -M MIRNA_SET_FILE, --mirna-set-file MIRNA_SET_FILE
                        Specify miRNA/precursor target set via file
```

(continues on next page)

(continued from previous page)

```
mutually exclusive optional arguments:
  either a set or file may be provided

  -c CATEGORIES [CATEGORIES ...], --categories CATEGORIES [CATEGORIES ...]
                        Set of categories to include in analysis, can include
                        `all`, `default`, `expert` or specific categories
  -C CATEGORIES_FILE, --categories-file CATEGORIES_FILE
                        File specifying categories to include in analysis
```

Examples:

```
$ mieaa gsea hsa --precursors -M precursors.txt -C categories.txt -o results.csv
$ mieaa gsea hsa -p -M precursors.txt -c HMDD MNDR > results.csv
$ mieaa gsea mmu -M mirnas.txt -C categories.txt --adjustment none
$ mieaa gsea rno -M mirnas.txt -C categories.txt -a bonferroni --json -o results.json
```

3.1.5 ora

Over-representation Analysis (ORA)

```
usage: miEAA ora [-h] [-v] [-m MIRNA_SET [MIRNA_SET ...]] [-M MIRNA_SET_FILE]
                 [-p] [-o OUTFILE] [-x] [-c CATEGORIES [CATEGORIES ...]]
                 [-C CATEGORIES_FILE] [-t THRESHOLD] [-s SIGNIFICANCE] [-g]
                 [-a {none,fdr,bonferroni,BY,holm,hochberg,hommel}]
                 [--csv | --json] [-r REFERENCE_SET [REFERENCE_SET ...]]
                 [-R REFERENCE_SET_FILE]
                 {hsa,mmu,rno,ath,bta,cel,dme,dre,gga,ssc}

positional arguments:
  {hsa,mmu,rno,ath,bta,cel,dme,dre,gga,ssc}
                        Species

optional arguments:
  -h, --help            show this help message and exit
  -v, --verbose         Always print results to stdout
  -p, --precursor, --precursors
                        Use if running on a set of precursors as opposed to
                        miRNAs
  -o OUTFILE, --outfile OUTFILE
                        Save results to provided file
  -x, --no-results      Do not monitor progress or obtain results. Can
                        retrieve later using Job ID.
  -t THRESHOLD, --threshold THRESHOLD
                        Filter out subcategories that contain less than this
                        many miRNAs/precursors (default=2)
  -s SIGNIFICANCE, --significance SIGNIFICANCE, --alpha SIGNIFICANCE
                        Significance level (default=0.05)
  -g, --group-adjust    Adjust p-values over aggregated groups (By default
                        each group is adjusted independently)
  -a {none,fdr,bonferroni,BY,holm,hochberg,hommel}, --adjustment {none,fdr,bonferroni,
↪BY,holm,hochberg,hommel}
                        p-value adjustment method (default='fdr')
  --csv                 Store results in output file in csv format (default)
  --json                Store results in output file in json format (default
```

(continues on next page)

(continued from previous page)

```
                        is csv)

mutually exclusive required arguments:
  either a set or file must be provided

  -m MIRNA_SET [MIRNA_SET ...], --mirna-set MIRNA_SET [MIRNA_SET ...]
                        miRNA/precursor target set
  -M MIRNA_SET_FILE, --mirna-set-file MIRNA_SET_FILE
                        Specify miRNA/precursor target set via file

mutually exclusive optional arguments:
  either a set or file may be provided

  -c CATEGORIES [CATEGORIES ...], --categories CATEGORIES [CATEGORIES ...]
                        Set of categories to include in analysis, can include
                        `all`, `default`, `expert` or specific categories
  -C CATEGORIES_FILE, --categories-file CATEGORIES_FILE
                        File specifying categories to include in analysis

mutually exclusive optional arguments:
  either a set or file may be provided

  -r REFERENCE_SET [REFERENCE_SET ...], --reference-set REFERENCE_SET [REFERENCE_SET .
↪..]
                        (Optional) Set of background miRNAs/precursors
  -R REFERENCE_SET_FILE, --reference-set-file REFERENCE_SET_FILE
                        (Optional) File specifying background
                        miRNAs/precursors
```

Examples:

```
$ mieaa ora hsa --precursors -M precursors.txt -C categories.txt > results.csv
$ mieaa ora hsa -p -M precursors.txt -c HMDD MNDR -o results.csv
$ mieaa ora hsa -p -M precursors.txt -C categories.txt -R reference.txt
$ mieaa ora mmu -M mirnas.txt -C categories.txt --adjustment none
$ mieaa ora rno -M mirnas.txt -C categories.txt -a bonferroni --json -o results.json
```

3.1.6 open

Open the specified mieaa web tool page in browser

```
usage: miEAA open [-h] [-v] [-j JOB_ID] {input,progress,results}

positional arguments:
  {input,progress,results}
                        Open MiEAA interface in browser

optional arguments:
  -h, --help            show this help message and exit
  -v, --verbose         Always print results to stdout
  -j JOB_ID, --jobid JOB_ID
                        Job ID
```

Examples:

```
$ mieaa open input
$ mieaa open progress -j 31b41542-7856-40be-91b2-fd6afe28fa0b
$ mieaa open results --jobid 31b41542-7856-40be-91b2-fd6afe28fa0b
```

CHAPTER 4

Links

- miEAA Web Server
- miEAA on the Python Package Index
- miEAA on Anaconda
- Chair for Clinical Bioinformatics, Saarland University
- GitHub IssueTracker
- GitHub Source Code

CHAPTER 5

License

MIT License

Copyright (c) 2020 Jeffrey Solomon

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Index

Symbols

A

C

G

N

O

R

S

T