MyTaxa Manual

Author: Chengwei Luo, Luis M. Rodriguez-R., and Konstantinos T. Konstantinidis

Citation: Chengwei Luo, Luis M. Rodriguez-R., and Konstantinos T. Konstantinidis, MyTaxa: an advanced taxonomic classifier for genomic and metagenomic sequences, *Nucleic Acids Research*, *in press*

[Installation, standalone only]

Note: if you don't intend to have standalone MyTaxa installed on your local machine but just use the webserver instead, you can skip this section.

To start with, you will need a C++ compiler and GNU make installed. Check if you have them by typing 'g++' and 'make' in your terminal. If you have them installed, then you can start moving onto install MyTaxa. Otherwise, please go to http://gcc.gnu.org for g++, and http://www.gnu.org/software/make/ for GNU make.

There are two venues to install MyTaxa. You can either download from the webserver page from Georgia Tech: http://enve-omics.ce.gatech.edu/mytaxa

Or, directly from github if you have git installed, go to terminal and run:

```
$ git clone git://github.com/luo-chengwei/MyTaxa
```

This will create a MyTaxa directory on your local machine, and then you need to 'cd' into this directory and run:

```
$ make
```

This will create the binary 'MyTaxa' with which you will run MyTaxa analysis from.

There is still one more step to complete the installation: the DBs. You can either download it following this address:

http://enve-omics.ce.gatech.edu/mytaxa/download/db.latest.tar.gz

or, you can run this utility script:

```
$ python utils/download_db.py
```

Then you need to decompress the db archive by running:

```
$ tar xvzf db.latest.tar.gz
```

Note that you should have all the db files stores at ./MyTaxa/db/

You should be all set. To verify it, you can run:

```
$ ./MyTaxa
```

This should print out a help/usage menu, and you are ready to go.

[How to run it]

Note: there is a toy example you can follow in the ./MyTaxa/example/ directory.

To run MyTaxa analysis, the only input is a tabular blast-like file with the results from a search against a reference database; but there are a few things you need to make sure before running it:

- The input file for the search should be genes carried by the query sequences. You can run ab initio gene prediction software such as MetaGeneMark, Prodigal, FragGeneScan *etc* (1-3) to get to this fastA file on your own; or, you can use utils/metaxa_prep.py for this purpose, which essentially uses Prodigal to call all the genes. **You can use the gene prediction results in .gff format from MetaGeneMark/GeneMark as a start point with the webserver, to follow this route, you can now jump to the webserver subsection.** (**Note:** since you will rely on the webserver for the db search, your priority will be low and your input size will be limited; also you cannot customize your reference database.)

- The naming of the entries of the input gene multi-fastA file. In general, you want to make every gene's name something looks like this:

```
>contig_xyz|gene_id
```

for instance, by running **metaxa_prep.py,** the genes predicted have names like:

```
>contig1|1915-2331|1
```

which means this gene is from "contig1" (contigID) and the gene name is "1915-2331|1" (gene ID); you can vary contigID and gene ID in anyway you wish, as long as the contig ID does not contain '|' sign, and they are combined by '|' sign.

- The reference database should be NCBI gi number referenced. For example, in the NR database, an entry looks like:

>gi|21305377|gb|AAM45611.1|AF384285_1 (AF384285) envelope protein [Human immunodeficiency virus type 1]

The numeric id "21305377" is the gi number you need in the results, and that's how MyTaxa will recognize the matches.

In the toy example folder, you can find "example.fa" as an example input for the search.

After you have the fasta file and reference database ready, you can run the blast search or any search engine you like, as long as you can produce a blast tabular format-like file (check the "example.blat" as an example). **With the output you have reached another entry point of the webserver, you can skip the rest and jump to the webserver subsection below.**

Then you should modify the output file from the search by adding three columns to the end of each match (i.e., each row), and they are:

- contigID
- geneID
- matched gene's gi number

You can use **utils/infile_convert.pl** to modify the tabular blast output into the MyTaxa input file (you can run "perl utils/infile_convert.pl" to get it print out the usage menu). You can check "example.mytaxa.input.txt" as an example of this file.

From this point, you can either use the standalone or the webserver to generate the output of MyTaxa.

Webserver:

Webserver address: http://enve-omics.ce.gatech.edu/mytaxa/submit

There are two entry points for the webserver as described above:

GFF file of gene prediction;
Tabular blast output like file;

You'll need to fill out a simple form (name, email, job name), and you can customize some of the parameters (e.g., bit-score, alignment length), then you can hit the “submit” button to get the job in queue. When it is finished, you will be notified by email.

Standalone:

In the standalone version, you can run in terminal this command:

```
$ ./MyTaxa <input_file> <output_file> <score_cutoff> <num_of_matches_to_use>
```

where the input file would be the modified blast output-like tabular file, the output file is where the MyTaxa prediction will be find. There are two parameters you need to set: the score cutoff (recommend value: 0.5) and number of matches to use in the analysis (recommend value: 5). It will generate an output that will be explained below.

[The output format and visualization]

There is a toy example available: ./MyTaxa/example/example.mytaxa

Each query sequence will have two lines in the output, and they are:

- Line 1: <query name> <lowest_level> <score> <lowest_level_taxonomy_id>
- Line 2: the taxonomy path from (super)-kingdom down to the lowest taxon predicted.

You can parse it into relative abundance at phylum/genus/species levels by running **utils/MyTaxa.distribution.pl**. (run “perl utils/MyTaxa.distribution.pl” to print out detailed usage information).

This will generate three output files corresponding to phylum, genus, and species level relative abundance of different taxa. In the example folder, you can find toy examples: “example.mytaxa.Phylum.txt”, “example.mytaxa.Genus.txt”, and “example.mytaxa.Species.txt”.

The output from MyTaxa could also be used to visualize in Krona (http://sourceforge.net/p/krona/home/krona/) by using the utility script **utils/mytaxa2krona.py:**

```
$ python utils/mytaxa2krona.py <mytaxa_output> <krona_input>
```

the krona_input would a file generated by this script that would be used to generate the interactive pie-chart by Krona (4). A toy example for this file is ./MyTaxa/example/example.krona_input.txt.

Then you can run Krona as:

```
$ ktImportText –o <output_html> <krona_input>
```

the output_html would be the output from Krona that you can open in the mainstream browsers to visualize the composition of the MyTaxa output. A toy example is ./MyTaxa/example/example.krona_input.txt

[Trouble shooting]

We have a FAQ page at: http://enve-omics.ce.gatech.edu/mytaxa/faq

You can refer to the wiki page where some of the issues/fix would be reported:

https://github.com/luo-chengwei/MyTaxa/wiki

References

1. Rho M, Tang H, & Ye Y (2010) FragGeneScan: predicting genes in short and error-prone reads. *Nucleic acids research* 38(20):e191.
2. Zhu W, Lomsadze A, & Borodovsky M (2010) Ab initio gene identification in metagenomic sequences. *Nucleic acids research* 38(12):e132.
3. Hyatt D*, et al.* (2010) Prodigal: prokaryotic gene recognition and translation initiation site identification. *BMC bioinformatics* 11:119.
4. Ondov BD, Bergman NH, & Phillippy AM (2011) Interactive metagenomic visualization in a Web browser. *BMC bioinformatics* 12:385.