

User documentation to type OTUS

November, 23, 2023

Typing an OTU consists in defining whether it is (i) composed or single and(ii) for single OTUs only, whether it is with or without noise. See the article ' Simple approaches for evaluation of OTUs quality based on dissimilarities arrays' (<https://hal.inrae.fr/hal-03824588v1>) for the definition of these types, the description of the method for typing, and the presentation of the study on diatoms of the bassin of the Arcachon Bay (France), referred to as the diatoms study here. This study is based on data provided by the Malabar project (Auby, I., Méteigner, C., Rumebe, M., Chancerel, E., Salin, F., Aluome, C., Barraquand, F., Carassou, L., Del Amo, Y., Meleder, V., Petit, A., Picoche, C., Frigerio, J.-M., and Franc, A. (2022). Malabar datasets used in study "OTU quality from dissimilarity arrays". Recherche Data Gouv, V1, DOI: 10.57745/7T2UCB).

The zip files of the scripts (SRC.zip containing SRC directory), the data for the diatoms study (DATA_DIATOMS.zip containing DATA_DIATOMS directory), and the results of the diatoms study (OUTPUT_DIATOMS.zip containing OUTPUT_DIATOMS directory) are available on FigShare at <https://doi.org/10.6084/m9.figshare.20764690>. SRC directory contains scripts (both Python and R). DATA_DIATOMS directory contains 32 samples directories each containing OTUs dissimilarity arrays. OUTPUT_DIATOMS directory contains the characteristics of the noise classifier and for each sample, the resulting OTUs type files.

This documentation explains step by step (i) how to install the two generic Python scripts LEARN_NOISE_CLASSIFIER.py and TYPE_OTU.py, (ii) how to re-run the diatoms study or just look at results, (iii) how to structure the data directory (containing dissimilarity arrays of OTUs) for a new study, (iv) how to learn the parameters of a noise classifier, and (v) how to type the OTUs of a sample.

1/ Install

To install the two generic Python scripts, LEARN_NOISE_CLASSIFIER.py and TYPE_OTU.py, load SRC.zip file on the FigShare project

Unzip the file:

```
> unzip SRC.zip
```

The created SRC directory contains the two Python scripts LEARN_NOISE_CLASSIFIER.py and TYPE_OTU.py, the R script RUN_SBM.R to learn Stochastic Block Models and a shell script RUN_DIATOMS_STUDY.py to launch the diatoms study.

Some software have to be installed on the computing platform:

- Python 3.8 or more (not tested with previous versions, the diatoms study was run with 3.9),
- Python library scikit-learn (only to learn noise classifier when running LEARN_NOISE_CLASSIFIER.py),
- R 4.2 or more with the libraries blockmodels (not necessary if the user just wants to type OTUs as composed or single by running TYPE_OTU.py), and lapack 3.10 or more.

Initialize the SRC path in both Python scripts LEARN_NOISE_CLASSIFIER.py (line 40) and TYPE_OTU.py (line 43) to be able to call them from any directory.

For example:

```
src_dir = "/home/user_id/TYPE_OTU/SRC"
```

2/ Diatoms study

To rerun the diatoms study, load DATA_DIATOMS.zip file (7.3Gb) on the Figshare project

Unzip the file in the same directory that the one where SRC has been created..

Create an OUTPUT directory to store created files.

Go to the SRC directory and run the script RUN_DIATOMS_STUDY.sh .

Beware that some OTUs are bigs (several thousands of sequences), use a computer with enough RAM (on our computer up to 80 Gb of RAM were used).

```
> unzip DATA_DIATOMS.zip
> mkdir OUTPUT
> cd SRC
> RUN_DIATOMS_STUDY.sh
```

To just examine the results of the diatoms study and not rerun the study, load OUPUT_DIATOMS.zip file on the Figshare project and unzip the file.

```
> unzip OUTPUT_DIATOMS.zip
```

3/ Structure data directory for a new study

The directory containing the dissimilarity array of all OTUs of all samples will be called the data directory. For a new study, in this documentation, it will be named <data_dir>.

This directory contains directories, one per sample. If there are k samples, in this documentation, they will be named <sample_name_1> ... <sample_name_k> .

The directory of a sample contains OTUs dissimilarity array text files of this sample. No constraint on names files. Beware that the sequence name of each row and column of the array must be provided in the text file.

For example, for the diatoms study (see DATA_DIATOMS.zip file), the data directory <data_dir> is DATA_DIATOMS. This directory contains 32 samples directories including TEY_BEN_Autumn directory. This later contains a dissimilarity array file per OTU, including otu_359.dis file that is the following text file:

```
seq_id 181113_BM_BEN_A_Tey_rbcL-30163;size=1 181113_BM_BEN_A_Tey_rbcL-32387;size=1
181113_BM_BEN_A_Tey_rbcL-30163;size=1 1.5 3.0
181113_BM_BEN_A_Tey_rbcL-32387;size=1 3.0 1.5
```

Note that as dissimilarities between reads are computed from alignment score, and as 'N' are accepted, the crude computation of dissimilarities between a read and itself may artificially generate a non zero value on the diagonal. The diagonal of all dissimilarity arrays is set to 0 in our script.

4/ Learning a noise classifier

To type whether a single OTUs of a particular sample is with or without noise, it is necessary to have the noise classifier adapted to this sample. If the noise classifier does not exist, it is necessary to learn it. To do so, a set of training samples has to be defined by the user and provided to the LEARN_NOISE_CLASSIFIER.py script.

The usage and options of the script are given by -h option.

```
> LEARN_NOISE_CLASSIFIER.py -h
Usage: LEARN_NOISE_CLASSIFIER.py [options] sample_name_1 [sample_name_n]

Options:
-h, --help                show this help message and exit
-d DATA_DIR, --data_dir=DATA_DIR
                          directory of samples
-o OUTPUT_DIR, --output_dir=OUTPUT_DIR
                          directory to store results
-c NOISE_CLASSIFIER_FILENAME, --noise_classifier=NOISE_CLASSIFIER_FILENAME
                          filename of the file containing the parameters of
                          the noise classifier (default value: params_noise_classifier)
-n SEQUENCES_MIN_NB, --sequences_min_nb=SEQUENCES_MIN_NB
                          required minimum number of sequences to type an OTU
                          (default value: 20)
-g GAP, --gap=GAP         distance threshold to define OTUs (default value: 9)
-b KDE_BANDWIDTH, --kde_bandwidth=KDE_BANDWIDTH
                          bandwidth parameter to estimate theta density (default
                          value: 0.05)
```

The value provided respectively to --data (or -d) option, --output option (or -o), --noise_classifier (or -c) will be called <data_dir>, <output_dir>, <params_noise_classifier>.

The script will read dissimilarity array files found in <data_dir>/<sample_name_1>...<data_dir>/<sample_name_n>. Finally, the script generates the file <output_dir>/<params_noise_classifier> which contains the parameters of the noise classifier. Two other files are written <output_dir>/<params_noise_classifier>.txt (confusion matrix and mean AUC with 10 Cross Validation) and <output_dir>/<params_noise_classifier>.png (display of decision frontier with expert classification of training examples) to check the quality of the classifier.

Input files

<data_dir>/<sample_name_1>
[<data_dir>/<sample_name_n>]

Output files

<output_dir>/<params_noise_classifier>
<output_dir>/<params_noise_classifier>.txt
<output_dir>/<params_noise_classifier>.log

At least 20 negative (without noise) and 20 positive (with noise) examples are required. If it is not the case, the script is stopped without learning parameters of a noise classifier.

Note that all OTUs of the training set may not be single OTUs. However, the typing into with or without noise applies only to single OTUs. Therefore the LEARN_NOISE_CLASSIFIER.py script first types the OTUs of the training set composed or single.

Example. For the diatoms study, the data directory <data_dir> is DATA_DIATOMS (see DATA_DIATOMS.zip file). The eight samples collected on the Teychan site (samples names beginning by TEY) define the training set.

The Linux command, when in SRC directory, to learn the noise classifier is (see RUN_DIATOMS_STUDY.sh script):

```
> LEARN_NOISE_CLASSIFIER.py -d ../DATA -o ../OUTPUT TEY_BEN_Summer TEY_BEN_Autumn
TEY_BEN_Winter TEY_BEN_Spring TEY_PEL_Summer TEY_PEL_Autumn TEY_PEL_Winter TEY_PEL_Spring
```

The result of LEARN_NOISE_CLASSIFIER.py applied to these samples is the result directory OUTPUT_DIATOMS (see OUTPUT_DIATOMS.zip file).

The file OUTPUT_DIATOMS/params_noise_classifier, which defines the noise classifier parameters, contains:

```
-9.451803069318872
0.5685743153981582
0.8761286500960326
# noise classifier equation
# y = <line 1> + <line 2> * lambda_intra_max + <line 3> * lambda_inter
```

The classifier may be evaluated looking at the file OUTPUT_DIATOMS/params_noise_classifier.txt :

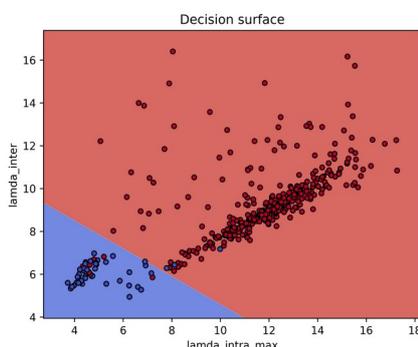
Confusion matrix:

```
[[ 42  6]
 [  6 375]]
```

Mean AUC with 10 Cross Validation:

```
0.9511296456461243
```

and the visualisation of the decision frontier with expert classification of training examples, in file OUTPUT_DIATOMS/params_noise_classifier.png:



The running time is about 24 min on a processeur Intel Xeon 2680.

5/ Typing OTUs

With the Python script TYPE_OTU.py, it is possible to type the OTUs of a sample (i) as composed or single and (ii) if the noise classifier has been learned, as with or without noise.

The usage and options of the script are given by -h option.

```
> TYPE_OTU.py -h
Usage: TYPE_OTU.py [options] sample_name

Options:
-h, --help                show this help message and exit
-d DATA_DIR, --data_dir=DATA_DIR
                           directory of samples
-o OUTPUT_DIR, --output_dir=OUTPUT_DIR
                           directory to store results
-S, --type_only_single    Type only composed/single and no noise (default value:
```

```

False)
-c NOISE_CLASSIFIER_FILENAME, --noise_classifier=NOISE_CLASSIFIER_FILENAME
  filename of the file containing the parameters of
  the noise classifier (default value: params_noise_classifier.txt)
-n SEQUENCES_MIN_NB, --sequences_min_nb=SEQUENCES_MIN_NB
  required minimum number of sequences to type an OTU
  (default value: 20)
-g GAP, --gap=GAP
  distance threshold to define OTUs (default value: 9)
-b KDE_BANDWIDTH, --kde_bandwidth=KDE_BANDWIDTH
  bandwidth parameter to estimate theta density (default
  value: 0.05)

```

The value provided respectively to --data (or -d) option, --output option (or -o), --noise_classifier (or -c) will be called <data_dir>, <output_dir>, <params_noise_classifier>.

The script read dissimilarity arrays files found in <data_dir>/<sample_name>. If it is not specified to only type OTUs as composed or single (option --type_only_single or -S), the parameters of the noise classifier are also read in <output_dir>/<params_noise_classifier>.

Two files are created as result:

- <output_dir>/<sample>_type.txt

A text file with the following columns: OTU name, OTU size (number of sequences), is_composed (1 : yes, 0 : no, -1 : OTU not considered), is_noise (1 : yes, 0 : no, -1 : OTU not considered), theta (ratio between the number of missing edges in the graph associated to an OTU over the total number of possible edges), y (distance to the frontier of the noise classifier), degree20 (proportion of sequence with a degree less than 0.20).

- <output_dir>/<sample>_theta.png (the estimated density of theta with identification of the threshold differencing composed and single OTUs).

Input files

<data_dir>/<sample_name>
 <output_dir>/<params_noise_classifier>

Output files

<output_dir>/<sample_name>_type.txt
 <output_dir>/<sample_name>_theta.png

If argument -S is given, only the composed / single typing is done. No noise typing means (i) no necessity to learn noise classifier, (ii) no evaluation of a SBM (Stochastic Bloc Model) with the R script RUN_SBM.R. This results in a very reduced time execution.

It is necessary that the user checks that the estimated density of theta is comparable at the one shows in the reference of the approach (curve enough smoothed but not too much). If it is not the case, the option --kde_bandwidth (or -b) has to be adjusted.

Note that it is possible to launch OTU typing of several samples in parallel on a computer.

Example. For the diatoms study, the sample TEY_BEN_Winter is typed with the following Linux command line (see RUN_DIATOMS_STUDY.sh):

```
> TYPE_OTU.py -d ../DATA_DIATOMS -o ../OUTPUT_DIATOMS TEY_BEN_Winter
```

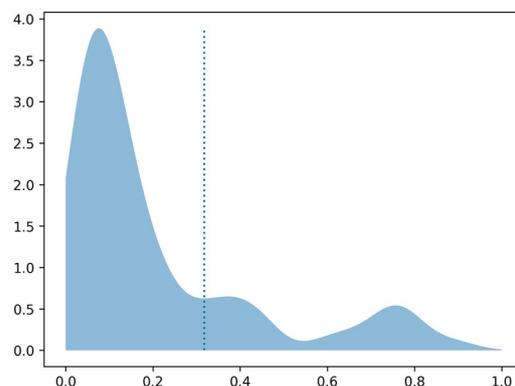
Here is the beginning of OUTPUT_DIATOMS/TEY_BEN_Winter_type.txt:

```

otu_0.dis 7068 1 -1 0.6220 -1.0000 0.1429
otu_1.dis 4629 1 -1 0.8805 -1.0000 0.7699
otu_2.dis 3434 0 1 0.1067 3.8711 0.0102

```

Here is the image of the file OUTPUT_DIATOMS/TEY_PEL_Winter_theta.png.



The running time is 1 min on a processeur Intel Xeon 2680. For all the 32 samples, the running time is about 2,5 hours.