

Author: N.Lapalu
Date: 25/10/2015
Title: MetaGPipe manual

Principles:

MetaGPipe is a wrapper of the QIIME framework and FungalITSExtractor tool. It was especially designed to analyze PCR amplicons of fungal ITS regions sequenced with 454 technology. It was then successfully used with Illumina reads, but the analysis report exported in .xls format could be laborious to obtain. MetaGPipe extracts ITS regions from input sequences and clusterizes them with uclust by using 97% similarity as a cutoff. Each cluster is designated by its most representative sequence. Taxonomic assignation is then performed by Blast against a filtered databank of ITS regions. A reclustering step could be launched to study intra-cluster variability and highlight sequencing errors. In this case, we recommend to set the threshold to 1 (100% of similarity).

For impatient:

Download and install Virtualbox.
Download the virtual machine :
<https://urgi.versailles.inra.fr/download/fungigrapemetag/MetaGPipe.ova>
Import the virtual machine in virtualbox, and launch it

login: metagpipe
password: metagpipe

if needed:
root password: metagpipe

The metagpipe user environment is set to execute the pipeline (see the .bashrc), no configuration are required. The virtual machine contains a testing dataset in the `data` directory. We recommend to run analyses in the `analysis` directory. The sequence reference databanks are in `banks` and HMM profiles for FungalITSExtraction are in `profiles`. The pipeline source code is in `MetaGPipe-1.0`.

If you want to test the pipeline go to the `analysis` directory. The pipeline requires a configuration file (`metag.conf`), already filled.

To test the pipeline:

```
cd /home/metagpipe/analysis
```

Launch the pipeline:

```
MetaGPipe.py -i /home/metagpipe/data/data81.fasta -r  
/home/metagpipe/banks/FungalITS1Genbank_28012015_taxonMappingID.fasta -t  
/home/metagpipe/banks/FungalITS1Genbank_28012015_taxonMappingID.txt -w data81 -C  
metag.conf -s ITS1 -c 1
```

Generate the spreadsheet:

```
CreateAnalysisCardForMetaGAnalysis.py -i /home/metagpipe/data/data81.fasta -p  
/home/metagpipe/data/data81.fasta -e data81/FungalITSExtractor/ITS1.fasta -t  
data81/ITS1_otu_table.txt -r data81/ITS1.fasta_rep_set.fasta -o
```

```
data81/uclust_picked_otus/ITS1_otus.txt -c data81/reclustering
```

Install:

Dependencies

Python 2.6+
R
Mysql server
SGE
blastall
formatdb
uclust (available here : http://www.drive5.com/uclust/downloads1_2_22q.html)
HMMER 2.X
FungalITSExtractor (), already provide with the MetaGPipe, due to some additions in the code.

Python modules:

- setuptools
- numpy
- cogent
- pillow
- QIIME

1] Install all dependencies
Create a mysql database.

2] Download and install the source code

Download the archive

```
wget https://urqi.versailles.inra.fr/download/fungigrapemetag/MetaGPipe-1.0.tar.gz
```

untar the archive in your \$HOME dir and install

```
cd $HOME  
tar -xvf MetaGPipe-1.0.tar.gz  
cd MetaGPipe-1.0  
python setup_MetapPipe install
```

edit your .bashrc and add environment variables

```
cd $HOME  
edit .bashrc
```

add:

```
export REPET_PATH=$HOME/MetaGPipe-1.0  
export PYTHONPATH=$HOME/MetaGPipe-1.0  
export PATH=$HOME/MetaGPipe-1.0/bin:$PATH  
source .bashrc
```

3] Download databanks and data

```
cd $HOME  
wget https://urqi.versailles.inra.fr/download/fungigrapemetag/banks.tar.gz  
tar -xzf banks.tar.gz  
wget https://urqi.versailles.inra.fr/download/fungigrapemetag/data.tar.gz  
tar -xzf data.tar.gz
```

4] Set your working directory and configuration file

```

cd $HOME
mkdir analysis
cd analysis
touch metag.conf
edit metag.conf and add:
[MetaG_env]
repet_host:metagpipe
repet_user:metagpipe
repet_pw:metagpipe
repet_db:metagpipe
repet_port:3306
repet_job_manager:sge
metagpipe:/home/metagpipe/MetaGPIPE-1.0/Gnome_tools
profile_path:/home/metagpipe/profiles/HMMs
tmpDir:/home/metagpipe/tmpdir

```

```

repet_host: database server name for mysql
repet_user: mysql database user
repet_pw: mysql database user password
repet_db: mysql database name
repet_port: mysql port
repet_job_manager: cluster job manager
metagpipe: full path to the code
profile_path: path to the profile directory
tmpDir: path to the temporary directory (need to be created by user)

```

If you want to test the install, go to

```

cd $HOME
mkdir analysis
cd analysis
MetaGPIPE.py -i ../data/data81.fasta -r
../banks/FungalITS1Genbank_28012015_taxonMappingID.fasta -t
../banks/FungalITS1Genbank_28012015_taxonMappingID.txt -w ../banks/data81 -C
metag.conf -s ITS1 -c 1

```

```

CreateAnalysisCardForMetaGAnalysis.py -i ../data/data81.fasta -p
../data/data81.fasta -e data81/FungalITSExtractor/ITS1.fasta -t
data81/ITS1_otu_table.txt -r data81/ITS1.fasta_rep_set.fasta -o
data81/uclust_picked_otus/ITS1_otus.txt -c data81/reclustering

```

Options:

Launch MetaGPIPE.py -h, to get all available options

```

Usage: MetaGPIPE.py -i <input file> -r <reference file> -t <taxonomy file> -w
<directory> -C <configuration file> -s <sequence type to analyze> [--noextraction]

```

Description: MetaGPIPE is a pipeline for metagenomic data.

Options:

```

-h, --help          show this help message and exit
-i INPUTFILENAME, --input=INPUTFILENAME
                    input file (fasta type)
-r REFERENCE, --reference=REFERENCE

```

```

reference file
-t TAXONOMY, --taxonomy=TAXONOMY
taxonomy file (optional)
-w WORKINGDIR, --workingDir=WORKINGDIR
output directory
-C CONFIGFILENAME, --configFileName=CONFIGFILENAME
configuration file
-s SEQUENCETYPE, --sequenceType=SEQUENCETYPE
sequence type (ITS1/ITS2/Both/ITS1Complete)
-c RECLUSTERING, --reclustering=RECLUSTERING
% similarity for reclustering accepted values between
[0.97-1]
--noextraction ITS Extraction

```

Launch CreateAnalysisCardForMetaGAnalysis.py -h to get all available options

Usage: CreateAnalysisCardForMetaGAnalysis.py [options]

Options:

```

-h, --help show this help message and exit
-i INPUT, --input=INPUT
file with raw data
-p PYROCLEANERFILENAME, --pyrocleaner=PYROCLEANERFILENAME
file with cleaned data (software like pyrocleaner)
-e ITSFILENAME, --extractITS=ITSFILENAME
file with extracted ITS sequences, in
analysis_directory/FungalITSExtractor directory
-t OTUTABLEFILENAME, --OTUTableFile=OTUTABLEFILENAME
file with OTUs, in
analysis_directory/ITSX_otu_table.txt
-o OTUFILENAME, --OTUFile=OTUFILENAME
file with picked OTUs, in analysis_directory/uclust_pi
cked_otus/ITS1_otus/ITS1_otus.txt
-r REPFILENAME, --repFile=REPFILENAME
file with rep seq, in
analysis_directory/ITSX_X_rep_set.fasta
-c RECLUSTERINGDIR, --reclusteringDir=RECLUSTERINGDIR
reclustering directory

```

Remarks: if you do not have the raw data file or cleaned data file, set the same file to -i and -p options.

Outputs:

List of files in the result directory:

```

data81.fasta (link to the input fasta file)
data81.fasta_QIIME_analysis.xls (Analysis card)
data81_ITS1_tax_assignments.txt
data81_newH.fasta (reformated input fasta file)
FungalITS1Genbank_28012015_taxonMappingID.fasta (reference databank: fasta file)
FungalITS1Genbank_28012015_taxonMappingID.txt (reference databank: mapping file)
FungalITSExtractor (extraction directory)
ITS1.fasta_rep_set.fasta (representative sequence)
ITS1_otu_table.txt (otu table)
metag.conf (configuration file)

```

reclustering (reclustering directory)
uclust_picked_otu (uclust directory)

Analysis card

File: data81.fasta_QIIME_analysis.xls

The spreadsheet contains a sheet ANALYSIS RESUME, with 3 graphics and the list of clusters (index, representative sequence, nb of members) and one sheet/cluster, with reclustering results, if performed.

Extracted ITS regions

File: FungalITSExtractor/ITS1.fasta

```
>fasta_55HQG576205F501H
TCGAGTCAGGGTCC
>fasta_108HQG576205FXV09
GTAGTGATCAGGGTTGCGCTCGTTGCGGGACTTAACCCAACATCTCACGACACGAGCTGACGACAGCCATGCAGCACCTGTG
CACTAGCCAGCC
>fasta_141HQG576205FZ9DR
TGCTGTCTCTCGGGACGGCGCCACCGCCGGTGGACTACTAACTCTGTTAATTTTTGTTCAATCTGAATCAAATAAGAAA
TAAGTTAA
>fasta_286HQG576205F4AHE
GCTCCCGGTAAAAACGGGACGGCCGCCAGAGGACCCCTAAACTCTGTTTCTATATGTAACTTCTGAGTAAAAAACCAT
AAAATAAAATCAAAA
> accession
extracted sequence (ITS1, ITS2 or Both)
```

Clusters

File: uclust_picked_otus/ITS1_otus.txt

```
0   fasta_517HQG576205FXEAL fasta_605HQG576205F1IYE
1   fasta_398HQG576205F35LF
2   fasta_312HQG576205F3Z09 fasta_375HQG576205FXQJG
3   fasta_611HQG576205GL1J1
4   fasta_461HQG576205GD85Y fasta_435HQG576205F45UX
5   fasta_444HQG576205FV2M5 fasta_400HQG576205F71C4
6   fasta_608HQG576205GH5UW
```

c1: cluster index

c2: members (accessions) of the cluster

Cluster taxonomic assignation

File: data81_ITS1_tax_assignments.txt

1	Eukaryota;Fungi;Dikarya;Ascomycota;Pezizomycotina;....	5e-60	252396
0	Eukaryota;Fungi;environmental samples;uncultured fungus	4e-25	690542
3	Eukaryota;Fungi;Dikarya;Ascomycota;Pezizomycotina;....	2e-19	382973
2	Eukaryota;Fungi;Dikarya;Ascomycota;Pezizomycotina;....	2e-65	77306
5	Eukaryota;Fungi;environmental samples;uncultured fungus	2e-30	681695
4	Eukaryota;Fungi;environmental samples;uncultured fungus	3e-42	642922
6	Eukaryota;Fungi;environmental samples;uncultured fungus	1e-35	138704
9	No blast hit None None		
8	Eukaryota;Fungi;environmental samples;uncultured fungus	8e-78	599067
39	Eukaryota;Fungi;environmental samples;uncultured fungus	2e-17	119600
12	No blast hit None None		
11	No blast hit None None		

```
10    No blast hit      None  None
```

c1: cluster index

c2: lineage

c3: evaluate

c4: lineage index in databank

Cluster representative sequence

File: ITS1.fasta_rep_set.fasta

```
>0 fasta_517HQG576205FXEAL
CTGAGTTTTTTTAAACTCTCCAAACCATGTGAACTTACCACTGTTGCCTCGGTGGATGGTGCTGTCTCTC
>1 fasta_398HQG576205F35LF
GTGAACATACCTTATGTTGCCTCGGCGGATCAGCCCGGACCCCGTAAAAAAGGGACGGCCCGCCGAGGAACCCTAAACTC
TGTTTTTAGTGGAACCTCTGAGTATAAAAAACAAATAAATCAA
>10 fasta_508HQG576205F0CV9
TAAAAATCAAAA
>11 fasta_55HQG576205F501H
TCGAGTCAGGGTCC
>12 fasta_500HQG576205GB5TB
CTGAGTTTTTAAAC
>cluster index representative accession
representative sequence
```

OTU table

File: ITS1_otu_table.txt

```
# QIIME v1.4.0 OTU table
#OTU ID      fasta Consensus Lineage
0      2      Eukaryota;Fungi;environmental samples;uncultured fungus
1      1      Eukaryota;Fungi;Dikarya;Ascomycota;Pezizomycotina;Sordariomycetes;...
2      2      Eukaryota;Fungi;Dikarya;Ascomycota;Pezizomycotina;Sordariomycetes;...
3      1      Eukaryota;Fungi;Dikarya;Ascomycota;Pezizomycotina;Sordariomycetes;...
4      2      Eukaryota;Fungi;environmental samples;uncultured fungus
5      2      Eukaryota;Fungi;environmental samples;uncultured fungus
```

c1: cluster index

c2: number of read in the cluster

c3 : lineage

Build your own fungal ITS databank

We provide working databanks with the pipeline, but you can build your own databanks if you want an up-to-date version. You will find below a protocol to download data, extract ITS and build the data to the required format. You will get 2 files for each databank (one per type of sequence: ITS1, ITS2, Both).

```
cd $HOME
mkdir mybanks
cd mybanks
DownloadEntriesFromGenbank.py -e your.email@domain.com -b 1000
cat *.gb > mybank.gb
GetFastaFileFromGenBankFile.py -i mybank.gb -o mybank.fasta
dbSplit.py -i mybank.fasta -n 10000 -d
ITSExtractorDataBanks.sh batches/ ../profiles/HMMs/
```

Concatenate the results in 3 databanks (ITS1, ITS2 and both), DD=day, MM=month, YYYY=year

```
cat batches/*/ITS1.fasta > FungalITS1Genbank_DDMMYYYY.fasta
cat batches/*/ITS2.fasta > FungalITS2Genbank_DDMMYYYY.fasta
cat batches/*/Both.fasta > FungalITS1andITS2Genbank_DDMMYYYY.fasta
```

Create the indexed taxonomy and fasta file

```
WriteTaxonomyMappingAndFastaFiles.py -f FungalITS1Genbank_DDMMYYYY.fasta -g mybank.gb
WriteTaxonomyMappingAndFastaFiles.py -f FungalITS2Genbank_DDMMYYYY.fasta -g mybank.gb
WriteTaxonomyMappingAndFastaFiles.py -f FungalITS1andITS2Genbank_DDMMYYYY.fasta -g mybank.gb
```

Your databanks are now ready to use. For each type, you get a taxonomy and fasta file(e.g., ITS1: FungalITS1Genbank_DDMMYYYY_taxonMappingID.txt and FungalITS1Genbank_DDMMYYYY_taxonMappingID.fasta).

Virtual machine: set the number of slots for job submission

MetaGPIPE uses the SGE scheduler to submit jobs. MetaGPIPE was initially designed to split input data and launch jobs on cluster. This virtual machine is configured with a number of slots equal to 1. If you run your virtual machine on a higher number of threads, you can increase this number.

The number of cpus allocated to the virtual machine is available in the virtualbox configuration >System >processors.

If you don't know the number of cpus, check with :

```
grep processor /proc/cpuinfo
```

you get a line per cpu:

```
processor      : 0
processor      : 1
.....
```

With CentOS, you can also use the command `lscpu`.

Now, to check the queue configuration:

```
qstat -f
```

queue name	qtype	resv/used/tot.	load_avg	arch	states
all.q@metagpipe	BIP	0/0/1	0.00	lx26-amd64	

The number of allocated slots is 1. Now you want to increase the number of slots : as root

```
su
```

```
qconf -mq all.q
```

the config is displayed

```
qname          all.q
hostlist       @allhosts
seq_no        0
load_thresholds np_load_avg=1.75
suspend_thresholds NONE
nsuspend      1
suspend_interval 00:05:00
priority       0
```

```
min_cpu_interval    00:05:00
processors          UNDEFINED
qtype               BATCH INTERACTIVE
ckpt_list           NONE
pe_list             make
rerun               FALSE
slots               1,[metagpipe=1]
tmpdir              /tmp
shell               /bin/csh
prolog              NONE
epilog              NONE
shell_start_mode    posix_compliant
starter_method      NONE
```

Edit `[metagpipe=1]` and replace the number with the number of desired slots (here 7 slots) `[metagpipe=7]`. Keep at least one cpu for the system. If you overload the scheduler with an inappropriate number of slots, the system will crash. Save the file with `esc` then `wq`
root@metagpipe modified "all.q" in cluster queue list

Checking:

```
qstat -f
```

queue	name	qtype	resv/used/tot.	load_avg	arch	states
all.q	@metagpipe	BIP	0/0/7	0.00	lx26-amd64	

Now if you launch metagpipe, you will be able to run 7 jobs in parallel.