DATMA USER MANUAL

13 de mayo de 2020

Version 1.0

Contents

1. DESCRIPTION

Distributed **A**u**T**omatic **M**etagenomic **A**ssembly and **A**nnotation framework -DATMA- is a pipeline for fast metagenomic analysis that orchestrates the following: sequencing quality control, 16S rRNA-identification, reads binning, *de novo* assembly and evaluation, gene prediction, and taxonomic annotation. Its distributed computing model can use multiple computing resources to reduce the analysis time. DATMA is freely available at: https://github.com/andvides/DATMA.

2. QUICK START

DATMA is available for the BIOCONDA repository. It requires the conda package manager to be installed. You must install the **Miniconda** package before running the following commands (refer to the documentation here).

2.1. Installation

To install DATMA, download (here) and uncompress the configuration files, then type the following command.

```
$> conda env create -f env.yml
$> conda install -n DATMA_env1
https://anaconda.org/bioconda/datma/1.0/download/noarch/datma-1.0-0.tar.bz2
$> conda activate DATMA_env1
```

2.2. Sequential Execution

Execute the **runDATMA.sh** script, use as arguments: the path to the configuration file, and enable the sequential mode (seq).
Example:

```
$> runDATMA.sh configBmini.txt seq
```

Modify the configuration file to set the path to the raw reads, databases, and external tools.

2.3. Distributed Execution

It requires that **COMPSs** had been installed (see section 6). Use the BIOCONDA repository to install DATMA in all the computers that conform the grid computing. Use the configuration files to administrate the DATMA workflow, and run the **runDATMA.sh** script in distributed mode (compss).
Example:

```
$> runDATMA.sh configBmini.txt compss
```

Modify the configuration file to set the path to the raw reads, databases, and external tools.

2.4. Results

DATMA generates the following directories:

- **clean**: Quality filter sequences.
- **16sSeq**: 16S rRNA Ribosomal sequences detected.
- **bins**: All bins generated by CLAME, assembled and annotated.
- **busco**: Quantitative assessment of genome assembly reported by Busco tool.
- **checkm_out**: Quantitative assessment of genome assembly reported by CheckM tool.
- **readsForbin.fastq**: Balance reads that were not binned.
- **readsForbin.exclude**: Total of reads binned
- **round_*_b***: Report directories for the bin stages.
- **htmls**: Report files in HTML format.

2.5. Unistall DATMA

To remove DATMA codes and tools type

```
$> conda deactivate
$> conda remove --name DATMA_env1 --all
```

3. INSTALL USING AUTOMATIC SCRIPT FOR LINUX

DATMA is written in Python and has been tested in Linux with Ubuntu distribution. Essential tools and their dependencies are included in the installation script, which requires **sudo** privileges. However, a manual installation could be needed if not all the dependencies are not previously installed.

3.1. Installation

Execute the **install_datma.sh** script, with sudo properties, to install DATMA (it does not include COMPSs support) and the framework tools.

$> sudo install_datma.sh

3.2. External tools

DATMA will install the following tools:

blast 2.6.0, bowtie2 2.3.5, busco 2.0.1, bwa 0.7.17, checkm-genome 1.0.12, fastqc 0.11.8, fastx_toolkit, flash2, gmp, kaiju, kraken 1.1, krona 2.7, matplotlib 2.2.3, maven, megahit 1.1.3, pplacer 1.1.alpha19, prinseq, prodigal, quast 5.0.2, rdp_classifier, samtools 1.9, spades 3.13.0, trimmomatic, velvet, clame, rapifilt, selectFasta, mergeNotCombined, unmerge.

3.3. Execution

Execute the **runDATMA.sh** script, use as arguments: the path to the configuration file and enable the sequential mode (seq) or distributed (compss), it requires that COMPSs has been installed (see section 6).
Example:

$> runDATMA.sh configBmini.txt seq

or

$> runDATMA.sh configBmini.txt compss

Modify the configuration file to set the path to the raw reads, databases, and external tools.

4. MANUAL INSTALLATION

4.1. DATMA source codes

Download DATMA source codes and export the path directory.

```
$> git clone https://github.com/andvides/DATMA.git
$> export PATH=<UserPath>/DATMA/codes/:$PATH
```

4.2. Download Framework tools

4.2.1. Reads Quality Trimming and Filtering

DATMA receives FASTQ, FASTA, or Standard Flowgram Files (SFF). For reads' quality control, it uses Trimmomatic or RAPIFILT. Source codes can be download from here and here.

4.2.2. 16S rRNA genes sequences detection

DATMA uses the BWA tool to map the raw reads against a ribosomal database (i.e., Rfam, RDP, and SILVA) and remove ribosomal sequences from the pool of reads to improve the binning.

4.2.3. CLAME binning

DATMA uses the CLAME tool to bin DNA sequences. It can be downloaded from: here.

4.2.4. Assembly and contigs' evaluation

DATMA assembles *de novo* all bins produced by CLAME. The user can select among different assembly tools: Velvet, SPAdes, or MegaHit. After assembly, the Quast tool evaluates the contigs and reports their metrics. Finally, DATMA uses the CheckM program to assess the quality and contamination of the bins.

4.2.5. ORF detection and taxonomic analysis

DATMA uses the assembled contigs to predict protein-coding-genes; the user can select between Prodigal or GeneMark for this task. Next, the contigs are annotated using BLAST and a local NT-database. It also provides the Kaiju tool for sensitive taxonomic classification.

4.2.6. Final report

DATMA reports the statistics of each workflow stage into an HTML file. It uses Krona to represent the taxonomic classification into an interactive plot.

4.3. Install Framework tools

First, install all the tools according to the author's instruction and recommendations. Then add each one to the execution **PATH**. Finally, ensure that each program can be executed from the command line.

```
$> export PATH=<UserPath>/<tool>/bin/:$PATH
```

5. DATABASES CONFIGURATION

5.1. 16S rRNA genes

Download the 16S rRNA database and generate a database index.
Example:

```
$> cd <UserPath>/DATMA/16sDatabases/rfam
$> bwa index rfam.fna
```

Update the the path into the configuration file, use the option **-database_16s_fasta** (see section 7).

5.2. NT-database

Download the Nucleotide database and specify the path into the configuration file. For the BLAST tool, use the option **-database_nt**. For the Kaiju tool, use the option **-database_kaiju** (see section 7).

5.3. CheckM databases

Download (here) and decompress the file to an appropriate folder and run the following to inform CheckM of where the files have been placed.

```
$> mkdir checkmDIR
$> cd checkmDIR
$> tar xzf checkm_data_2015_01_16.tar.gz
$> checkm data setRoot <checkmDIR PATH>
```

5.4. BUSCO datasets

Download the bacteria datasets and specify the path into the configuration file. Use the option **-lineage** (see section 7).

6. DISTRIBUTED MODE

6.1. COMP Superscalar (COMPSs)

It requires that the COMPSs version 2.0 tool be installed on the computers. The source codes and installation process can be consulted here.

6.2. Workers configuration

Replicate the DATMA installation process for all the workers.

6.3. Database configuration

When the distributed mode is selected, DATMA requires that the **database_nt** and **database_kaiju** options (see section 7) have identical locations in all the workers. If you are using different paths, you can provide a directory with the same path as the master directory and generate symbolic links of each database to this path.

7. CONFIGURATION FILES

7.1. DATMA configuration file

In this file, the user specifies the input-sequences file, the output directory, the workflow stages, the database directories, the number of threads to use, CLAME's bases parameter, etc.

```
############################################################
############################################################
##DATMA CONFIGURATION-FILE
##USE THIS FILE TO PASS THE ARGUMENTS
##TO EACH TOOL USED IN THE DATMA WORKFLOW
##SEE DATMA MANUAL FOR DETAILS
##VERSION 1 20-04-2020
############################################################
#STAGES
############################################################
#1.Sequencing quality control
#2.16S rRNA-identification
#3.CLAME reads binning
#4.Assembly, ORF prediction, and taxonomic annotation
#5.Final report
#-start_in: Flag to start in a particular stage
#           (INCLUDE the stage, [default: 1])
#-end_in:   Flag to end in a particular stage
#           (EXCLUDE the stage, [default: 6])
############################################################
-start_in 1
#-end_in 4
############################################################
#GENERAL PARAMETERS
#inputFile: DNA input file, the files can be gz compressed
#outputDir: Directory to store all the resulting files
#cpus:      Number of threads. [default: 1]
#typeReads: Use fastq, fasta, sff for unpaired reads or
```

```
#           Use Illumina for paired-end reads
#           (pass forward and reverse by a comma separated)
############################################################
-inputFile /DATMA/examples/bmini/Bancomini.fastq
-outputDir /DATMA/examples/bmini/bmini_out
-cpus 30
-typeReads fastq
############################################################
#Quality Control Configuration
#cleanTool:         It can be trimmomatic or rapifilt
#                   [default: rapifilt].
#trimmomatic_path:  Alternative path for  trimmomatic tool
#-lq:               Set the lef-cut value for quality scores
#                   [default: 0]
#-rq:               Set the right-cut value for quality scores
#                   [default: 0]
#-m:                Filter sequence shorter than min_len
#                   [default: 1]
#-wq:               Set windows size to check on the quality scores
#                   [default: 1]
#-quality_aux:      Use this flag to set other options supported
#                   by rapifilt or trimmomatic tools
############################################################
#-cleanTool trimmomatic
#-trimmomatic_path /home/users/DATMA/tools/Trimmomatic-0.38/
-lq 30
-rq 30
-m  60
-wq: 2
############################################################
#MERGE ILLUMINA FILES USIGN FLASH TOOL
#-fb:           Fragment length to merge reads [default 5]
#-flash_aux:    Set other options supported by FLASH tool
#-forceMerge:   Enable force merge of reads
#               that were not combined by FLASH
```

```
#############################################################
-fb 5
#############################################################
#16S rRNA detection
#-database_16s_fasta:   Path to the 16S rRNA database
#-useBWA:               Enable BWA to map raw reads
#                       against a 16S rRNA database
#-RDP_path:             Alternative PATH to RDP_CLASSIFIER tool
#-aux_16s:              Use this flag to set other options
#                       supported by BWA tool
#############################################################
-useBWA
-database_16s_fasta /DATMA/16sDatabase/rfam/RFAM_db.fasta
#############################################################
#CLAME
#-bases:        Set number of bases comma separator
#               to bin the reads [default: 70]
#-sizeBin:      Minimal reads to report a bin
#-ld:           Discard reads with least than this number
#               of matched reads
#-fm9:          PATH to previously generated fm9 file
#-block:        Process the dataset by blocks of this size n
#               [default: all]
#-clame_aux:    Use this flag to set other options
#               supported by CLAME tool
#############################################################
-bases 20,10
-sizeBin 80
-ld 10
-block 500
#-fm9 /DATMA/examples/bmini/Bancomini.fm9
#-clame_aux -tol 0.1
#############################################################
#ASSEMBLY OPTIONS
#-assembly: Select assembly tool: velvet, spades, or megahit
```

```
#           [default: spades]
#-asm_aux:  Use this flag to set other options
#           supported by the assembly tool
############################################################
-assembly spades
-asm_aux --only-assembler
############################################################
#ASSEMBLY QUALITY WITH QUAST TOOL OPTIONS
#-use_ref:              Enable a reference genome to asses
#                       the bin completeness
#-ref_name:             PATH to a reference genome
#-quast_aux --mgm:      Use MetaGeneMark for gene prediction
#-quast_aux --glimmer:  Use GlimmerHMM for gene prediction
#-quast_aux:            Use this flag to set other options supported
#                       by the QUAST tool
############################################################
#ORF DETECTION AND TAXONOMIC ANALYSIS
#-ORF_aux -p meta:  Enable metagenomic analysis for Prodigal tool
#-ORF_aux:          Use this flag to set other options
#                   supported by the Prodigal tool
#-checkm_aux:       Use this flag to set other options
#                   supported by the CheckM tool
############################################################
-ORF_aux -p meta
############################################################
#BLAST
#-database_nt:  PATH to NT database
#-blast_aux:    Use this flag to set other options
#               supported by the BLAST tool
############################################################
-database_nt /DATMA/nt_atabase/nt_dec_2019/nt
############################################################
#Kaiju
#-database_kaiju:   PATH to Kaiju database
#-kaiju_aux:        Use this flag to set other options
```

```
#                    supported by the Kaiju tool
##############################################################
-database_kaiju /DATMA/kaijudb/
##############################################################
#Krona
#-combine: Combine data from each bin within the Krona chart
##############################################################
-combine
##############################################################
#BUSCO
#-lineage: PATH to bacteria dataset for BUSCO tool
##############################################################
-lineage /DATMA/busco_bacteria/
##############################################################
#DELETE intermediate files
#If you set this flag, DATMA will delete most intermediate
#files to save disk space. #However, it would prevent DATMA
#from restarting from an intermediate point
##############################################################
#-delete
```

7.2. COMPSs configuration files

- **resources.xml**: provides information about the available execution resources. The basic configuration requires setting the ComputeNode, Processor Name, and ComputingUnits parameters (see the COMPSs_Installation_Manual file).

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ResourcesList>
    <ComputeNode Name="172.1.2.3">
        <Processor Name="master">
            <ComputingUnits>20</ComputingUnits>
        </Processor>
        <Adaptors>
            <Adaptor Name="integratedtoolkit.nio.master.NIOAdaptor">
                <SubmissionSystem>
                    <Interactive/>
                </SubmissionSystem>
                <Ports>
                    <MinPort>43001</MinPort>
                    <MaxPort>43002</MaxPort>
                </Ports>
            </Adaptor>
            <Adaptor Name="integratedtoolkit.gat.master.GATAdaptor">
                <SubmissionSystem>
                    <Batch>
                        <Queue>sequential</Queue>
                    </Batch>
                    <Interactive/>
                </SubmissionSystem>
                <BrokerAdaptor>sshtrilead</BrokerAdaptor>
            </Adaptor>
        </Adaptors>
    </ComputeNode>
    <ComputeNode Name="172.1.2.4">
        <Processor Name="worker1">
            <ComputingUnits>20</ComputingUnits>
        </Processor>
        <Adaptors>
            <Adaptor Name="integratedtoolkit.nio.master.NIOAdaptor">
                <SubmissionSystem>
                    <Interactive/>
                </SubmissionSystem>
                <Ports>
                    <MinPort>43001</MinPort>
                    <MaxPort>43002</MaxPort>
                </Ports>
            </Adaptor>
            <Adaptor Name="integratedtoolkit.gat.master.GATAdaptor">
                <SubmissionSystem>
                    <Batch>
```

```
                        <Queue>sequential</Queue>
                    </Batch>
                    <Interactive/>
                </SubmissionSystem>
                <BrokerAdaptor>sshtrilead</BrokerAdaptor>
            </Adaptor>
        </Adaptors>
    </ComputeNode>
</ResourcesList>
```

- **project.xml**: provides information about the resources used in a specific execution. The basic configuration requires setting the ComputeNode, InstallDir, User, AppDir, and Pythonpath parameters (see the COMPSs_Installation_Manual file).

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Project>
    <MasterNode/>
        <ComputeNode Name="172.1.2.3">
        <InstallDir>/opt/COMPSs/</InstallDir>
        <WorkingDir>/tmp/COMPSsWorker/</WorkingDir>
        <User>datma_user</User>
        <LimitOfTasks>1</LimitOfTasks>
    <Application>
        <AppDir>/DATMA/codes/</AppDir>
        <Pythonpath>/ DATMA/codes/</Pythonpath>
    </Application>
    </ComputeNode>
        <ComputeNode Name="172.1.2.4">
        <InstallDir>/opt/COMPSs/</InstallDir>
        <WorkingDir>/tmp/COMPSsWorker/</WorkingDir>
        <User>andresb</User>
        <LimitOfTasks>1</LimitOfTasks>
    <Application>
        <AppDir>/DATMA /codes/</AppDir>
        <Pythonpath>/DATMA /codes/</Pythonpath>
        </Application>
    </ComputeNode>
</Project>
```

8. RUNNING

8.1. Automatic mode

To easy the execution, DATMA provides a run script to execute the complete workflow. Just type the **runDATMA.sh** script use as arguments: the path to the configuration file and the running mode sequential (seq) or distributed (compss).

```
$> runDATMA.sh configBmini.txt seq
```

or

```
$> runDATMA.sh configBmini.txt compss
```

8.2. Sequential mode

DATMA can be executed using the Python command.
Example:

```
$> python datma_seq.py -f configBmini.txt seq
$> python finalReport.py -f configBmini.txt seq
```

8.3. Distributed mode

DATMA can be executed using the runcompss script of COMPSs.
Example:

$\$> runcompss --project=<project.xml> --resources=<resources.xml> --summary -d --lang=python datma.py -f <configurationFile.txt>$

```
$> python finalReport.py -f configBmini.txt seq
```

8.4. Delete intermediate files

DATMA can delete most intermediate files, which would save disk space. However, it would prevent DATMA from restarting from an intermediate point. You can set the **delete** flag into the configuration file (see section 7) or use the **deleteDATMAfiles.py** to remove no essential files.
Example:

```
$> python deleteDATMAfiles.py -f configBmini.txt seq
```

9. FAQ

- **Input sequences**
  DATMA receives unpaired reads in FASTQ, FASTA, or SFF format. Illumina paired-end (separated forward and reverse files) is also supported. Interlaced forward and reverse are not directly supported, but our unmerge program program can prepossess them

- **What Python version is required?**
  DATMA requires 2.7.17<Python version <3.6.5.

- **Error: ResolvePackageNotFound for _openmp_mutex**
  Install the packages that use this module individually.
  $> conda install -n busco -c bioconda -c conda-forge clame=1.0=he1b5a44_1

- **Error: BUSCO analysis failed**
  DATMA requires BUSCO version 2.0.1. Moreover, the environment variables AUGUSTUS_CONFIG_PATH and PATH must be set to AUGUSTUS directory.
  Example:
  $> export AUGUSTUS_CONFIG_PATH=<minicondaPATH>/pkgs/augustus/config/
  $> export PATH=<minicondaPATH>/pkgs/augustus/scripts/:$PATH

- **Error: COMPSS Unable to Launch JVM**
  DATMA requires COMPPSs version <2.4. Moreover the environment variable JAVA_HOME must be set to the JRE directory.
  Example:
  $> export JAVA_HOME=<minicondaPATH>/jre/

Please contact us:
bernardo.benavides@udea.edu.co
felipe.cabarcas@udea.edu.co