**SOP# 04.002.01**

**Taxonomic identification of isolates (Sanger 16S) – RDP**

**Author: Allison Perrotta**

**Date of Rev.: September 30, 2016**

**Page 1 of 3**

**Purpose**

This SOP describes the procedure used to identify the taxonomy of isolates using Sanger 16S sequences and data from the Ribosomal Database Project.

**Scope**

For exploratory purposes.

**Regulatory References**

NA

**Responsibility**

* Responsibility of experimentalist: understanding and performing this procedure as described; reporting any deviations or problems to area supervisor; adequately documenting the procedures and results.
* Responsibility of area manager or supervisor: ensuring that the analyst performing this procedure is qualified; ensuring that the procedure is followed, and updating the procedure as necessary.

**Definitions/Abbreviations**

RDP – Ribosomal Database Projects

**Related Documents:**

SOP# 02.281.01 - Bacterial Isolate Revival

SOP# 02.304.01 – DNA extraction from bacterial isolates – Alkaline PEG

SOP# 02.351.01 - Full Length 16S PCR Prep 27F-1495R

SOP# 02.355.01 – Sample preparation for GeneWiz Sanger sequencing – U515

**Required Equipment and Materials / Reagents**

* Computer capable of running python (version 2.7.10) commands from the command line
* Computer with java installed
* RDP classifier, available for download at https://rdp.cme.msu.edu/

**Precautions**

N/A

**Procedure**

1. Once GeneWiz data is released, usually within 24 hours of submission:
   1. Download data from the GeneWiz website

Make sure to check any odd traces and exclude any samples that are true failures

* 1. Copy data to desired directory on local computer
  2. Each reaction submitted, representing an isolate, will be provided as a separate sequence file.
  3. The vendor should provide information about the quality of these sequence files. The most common quality errors with this type of analysis are:
     1. Non-priming errors – This means that the primer did not hybridize to the template. This will result in very short sequence reads of less than 500 bases. In a successful Sanger sequencing reaction you can expect about 1,000 bases of sequence. To address this error, check the DNA concentrations of your extracted DNA and the PCR amplified 16S material you provided for sequencing, as one or both of these are most likely too dilute. Refer to the vendor’s submission requirements and re-submit (SOP# 02.355.01 – Sample preparation for GeneWiz Sanger sequencing – U515).
     2. Mixed peaks – This occurs when there is more than one composition of sequence in the reaction. This is usually due to contamination but can also occur with mixed colony genetics. To address this error you must re-isolate from your stock (see SOP# 02.281.01 - Bacterial Isolate Revival), extract DNA (SOP# 02.304.01 – DNA extraction from bacterial isolates – Alkaline PEG), prepare for sequencing (SOP# 02.351.01 - Full Length 16S PCR Prep 27F-1495R) and re-submit (SOP# 02.355.01 – Sample preparation for GeneWiz Sanger sequencing – U515).
  4. Combine sequence files into one file using the following command from the command line:
     1. cat \*.seq >> Name.fa
  5. Classifier sequences using RDP to identify the taxonomy of each isolate:
     1. This procedure will use a python script with the following code:

# author: scott olesen <swo@mit.edu>

import argparse, subprocess

if \_\_name\_\_ == '\_\_main\_\_':

p = argparse.ArgumentParser(description="", formatter\_class=argparse.ArgumentDefaultsHelpFormatter)

p.add\_argument('fasta', help='input fasta')

p.add\_argument('output', help='output tab-delimited file')

p.add\_argument('--format', '-f', choices=['fixrank', 'allrank'], default='fixrank', help='output format')

p.add\_argument('--jar', '-j', default='/ RDPTools/classifier.jar', metavar='/path/to/classifier.jar

', help='path to RDP classifier jar')

p.add\_argument('--java', '-x', default='/srv/pkg/java/java-1.8.0\_25/bin/java', metavar='/path/to/java', help='java binary')

p.add\_argument('--verbose', '-v', action='store\_true', help='show commands being executed?')

args = p.parse\_args()

cmd = [args.java, '-Xmx1g', '-jar', args.jar, 'classify', '-o', args.output, args.fasta]

if args.verbose:

print(" ".join(cmd))

subprocess.check\_call(cmd)

* + 1. Implement the above code by typing the following form the command line:

python sangerFile.fasta outputFile.txt

* + 1. The output file, named above as outputFile.txt, will contain the name given to each sanger sequence in the input fasta file followed by fields of taxanomic levels and the confidence level assigned to to classification for each level.

**Version History**

NA

**Worksheets**

NA

**Appendix**

NA