NanoAmpli-Seq - Bioinformatics Workflow

Szymon T Calus[1], Umer Zeeshan Ijaz[1], Ameet Pinto[2]
[1]University of Glasgow, [2]Northeastern University

dx.doi.org/10.17504/protocols.io.u25eyg6

Pinto Lab

Szymon T Calus
University of Glasgow

ABSTRACT

TAGS

amplicon

nanopore

Show tags

EXTERNAL LINK

https://www.biorxiv.org/content/early/2018/07/04/244517

PROTOCOL STATUS

Working

We use this protocol in our group and it is working very well.

GUIDELINES

Test data is available on the European Nucleotide Archive (ENA) website
https://www.ebi.ac.uk/ena/data/view/PRJEB21005

SAFETY WARNINGS

The highest accuracy of the data is being achieved when only 1D2 reads are used with INC-Seq, chopSEQ and nanoCLUST algorithms.

We tested 1D data as well with NA-S bioinformatics workflow, however, noticed increase in overall error rates and presence of false positive OTUs - validated on mock samples.

We do not recommend using 1D data for high accuracy profiling i.e. clinical samples. However, 1D reads can be used for research purposes e.g. development of correction algorithms etc.

BEFORE STARTING

Make sure all the necessary programs and dependencies are installed on your PC or server and work correctly.

Download and install all the required software.

1

SOFTWARE

Albacore v2.3.3

Linux
source by Oxford Nanopore Tech.

SOFTWARE
INC-Seq

Linux
source by Genome Institute of Singapore

SOFTWARE
chopSeq v0.3

Linux
source by University of Glasgow

SOFTWARE
nanoCLUST v0.4

Linux
source by University of Glasgow

Basecalling of raw nanopore data with Albacore software.

2

COMMAND

```
# Program requires input data (-i), version of the flow cell (-f),
# version of the sequencing kit (-k), output file (-o),
# amount of cores used for analysis (-t) and saving directory (-s).

/home/opt/.pyenv/versions/3.5.0/bin/full_1dsq_basecaller.py -i data/ -f FLO-MIN107 -k SQK-LSK308 -o fasta -t 20 -s .
```

Raw data (HDF5) generated with MinKNOW has to be basecalled with Albacore v2.3.3 (or newer) software. The output of the basecalling should be in FASTA format. Further analysis requires 1D2 data only so, full_1dsq_basecaller.py algorithm must be used.

Consensus calling of long 16S rRNA concatemerized reads with use of the INC-Seq algorithm.

3

COMMAND

```
# Export all necessary PATH's for the required programs.
# These PATH's are specific to our cluster and may differ
# to yours, depending on where you have installed these programs.

export PYENV_ROOT="/home/opt/.pyenv"
export PATH="$PYENV_ROOT/bin:$PATH"
eval "$(pyenv init -)"
export PYTHONPATH=/home/opt/INC-Seq/utils:$PYTHONPATH
export PATH=/home/opt/pacb/bin:$PATH
export PATH=/home/opt/pbdagcon/src/cpp:$PATH
export PATH=/home/opt/ncbi-blast-2.2.28+/bin:$PATH
export PATH=/home/opt/INC-Seq:$PATH
export PATH=/home/opt/.pyenv/versions/3.4.0/bin:$PATH


# INC-Seq consensus calling requires input data (-i),
# aligner (-a) e.g. poa, output file name (-o),
# minimum number of concatemers (--copy_num_thre) and --iterative.

inc-seq.py -i input.fasta -a poa -o incseq.fasta --iterative --copy_num_thre 3
```

The INC-Seq software requires basecalled data (e.g. Albacore) from Step 2. Correction of the data with INC-Seq algorithm uses only 1D2 data and is divided into two main steps:
1) Identification of segments made of 16S rRNA genes.
2) Anchor alignment of concatamerised amplicons and consensus calling with PBDAGCon.
Corrected reads have got ~98% accuracy and can be directly used as an input for chopSEQ software.

Linux

Correction of wrongly oriented reads and size filtration with a chopSeq algorithm.

4

COMMAND

```
# Algorithm requires input data (-i) from previous step,
# forward (-f) and reverse (-r) primer sequence,
# lower (-l) and maximum (-m) size filtration range,
# and new file destination (> new_file.fasta),
# while verbosity (-v) mode is optional.

chopSEQ.py -i incseq.fasta -f "AGRGTTTGATCMTGGCTCAG" -r "GGGCGGWGTGTACAAGRC" -l 1250 -m 1500 -v > chopseq.fasta
```

The chopSeq requires INC-Seq corrected data from Step 3.
Correction of the data is divided into multiple steps:
1) Identification of forward and reverse primers (e.g. 8F and 1387R) with pairwise2 aligner.
2) Re-orientation of incorrectly concatamerised reads and removal of tandem repeats recognised with use of etandem (EMBOSS) and subsequent merging of reads.
3) Size filtration with Biopython.
Now reads are qualified for nanoClust OTU binning and consensus calling.

Linux

Read binning and generation of OTUs with a nanoCLUST algorithm.

5

COMMAND

```
# Export all necessary PATH's for the required programs.
# These PATH's are specific to our cluster and may differ
# to yours, depending on where you have installed these programs.

export PATH=/home/opt/vsearch/bin:$PATH
export PATH=/home/opt/mafft-7.273-without-extensions/core/bin:$PATH
export MAFFT_BINARIES=/home/opt/mafft-7.273-without-extensions/core/libexec/mafft


# Provide chopSeq corrcted data (-i) and window split
# range (-s) for read partitioning and output folder (-o).

nanoCLUST.py -i chopSEQ.fasta -s 0,450,451,900,901,1300,-1 -o nanoclust_output/
```

The nanoCLUST requires chopSEQ corrected data from Step 4.
Correction of the data is divided into multiple steps:
1) Data is partitioned (i.e. 1-450,451-900, 901-1300bp).
2) Reads from each partition are grouped according to 97% similarity with VSEARCH.
3) VSEARCH partition dereplication, singleton removal and binning are performed on split data.
4) Optimal read sections are used for clustering.
5) MAFFT G-INS-i is used for within OTU alignment and consensus calling of data.
6) Consensus sequences are generated (~99.5% accuracy).
7) Abundance table is generated.

Linux