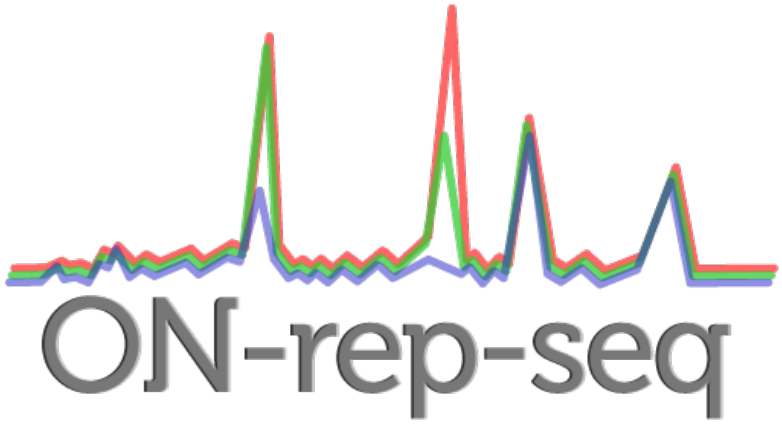
On-rep-seq Documentation

Release 1.0

Laura M Forero

Jul 17, 2020

Getting started


ON-rep-seq

CHAPTER 1

ON-rep-seq analysis toolbox

ON-rep-seq is a molecular method where bacterial (or yeast) selective intragenomic fragments generated with Rep-PCR are sequenced using Oxford Nanopore Technologies. This apporoch allows for species and sub-species level identification but also often strain level discrimination of bacterial and yeast isolates at very low cost. Current version of ON-rep-seq allows for analysis of up to 192 isolates in one R9 flow cell but will give most cost effective results by using flongle for which it wass initially designed.

CHAPTER 2

Requirements

- Anaconda

You can follow the installation guide .

CHAPTER 3

Installation

Clone github repo and enter directory:

```
git clone https://github.com/lauramilena3/On-rep-seq
cd On-rep-seq
```

Create On-rep-seq virtual environment and activate it:

```
conda env create -n On-rep-seq -f On-rep-seq.yaml
source activate On-rep-seq
```

Go into On-rep-seq directory and create variables to your basecalled data and the results directory of your choice:

```
fastqDir="/path/to/your/basecalled/data"
reusultDir="/path/to/your/desired/results/dir"
```

3.1 Note to macOS users (Canu)

If you are using os then you need to edit the config file to set a new directory for canu:

```
sed -i'.bak' -e 's/Linux-amd64/Darwin-amd64/g' config.yaml
```

3.2 Download kraken database

View the number of avaliable cores in your machine and set a number:

```
nproc
nCores="n"
```

If you are using your laptop we suggest you to leave 2 free cores for other system tasks.

Download kraken database. Notice this step can take up to 48 hours (!needs to be done only once):

```
kraken2-build --download-taxonomy --db db/NCBI-bacteria --threads $nCores #4h
kraken2-build --download-library bacteria --db db/NCBI-bacteria --threads $nCores #33h
kraken2-build --build --db db/NCBI-bacteria --threads $nCores #4h
```

CHAPTER 4

Running On-rep-seq analysis

4.1 Note to all users

ON-rep-seq is under regular updates. For better results, please keep your local installation up to date:

```
cd On-rep-seq
git pull
```

4.2 Input data

The input data is basecalled fastq files. Please check Guppy basecaller For best performance we strongly recommend basecalling on GPU (tested on GTX 1080Ti and RTX 2080).

4.3 Running

Run the snakemake pipeline with the desired number of cores:

```
snakemake -j $nCores --use-conda --config basecalled_dir=$fastqDir results_dir=
↪$reusultDir
```

4.3.1 Limiting memory

You can limit the memory resources (in Megabytes) used per core by using the resources directive as follows:

```
snakemake -j $nCores --use-conda --config basecalled_dir=$fastqDir results_dir=
↪$reusultDir --resources mem_mb=$max_mem
```

View dag of jobs to visualize the workflow

To view the dag run:

```
snakemake --dag | dot -Tpdf > dag.pdf
```

CHAPTER 5

Results structure

All results are stored in the Results folder as follows:

```
Results
├── 01_porechopped_data
│   └── {barcode}_demultiplexed.fastq     # Demultiplexed fastq per barcode
├── 02_LCPs
│   ├── LCP_clustering_heatmaps.ipynb     # Clustering jupyter notebook
│   ├── LCP_plots.pdf                     # Plots
│   ├── {barcode}.txt                     # All LCPs
│   └── LCPsClusteringData
│       └── {barcode}.txt                 # LCPs used for clustering
├── 03_LCPs_peaks
│   ├── 00_peak_consensus
│   │   └── fixed_{barcode}_{peak}.fasta  # Corrected consensus fasta of peaks
│   ├── 01_taxonomic_assignments
│   │   ├── taxonomy_assignments.txt      # Taxonomy of all barcodes
│   │   └── taxonomy_{barcode}.txt        # Taxonomy per Barcode
│   └── peaks_{barcode}.txt               # File with the peaks of each barcode
└── check.txt                             # Final file "On-rep-seq succesfuly executed
↪"
```

CHAPTER 6

Publications & citing

bioRxiv