Tutorial of aavMPRA

@mengm5@github.com

1 Graphic abstract

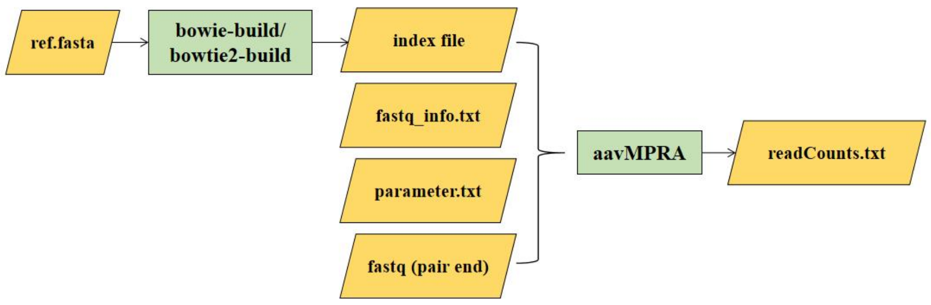

Fig1. Overview of aavMPRA

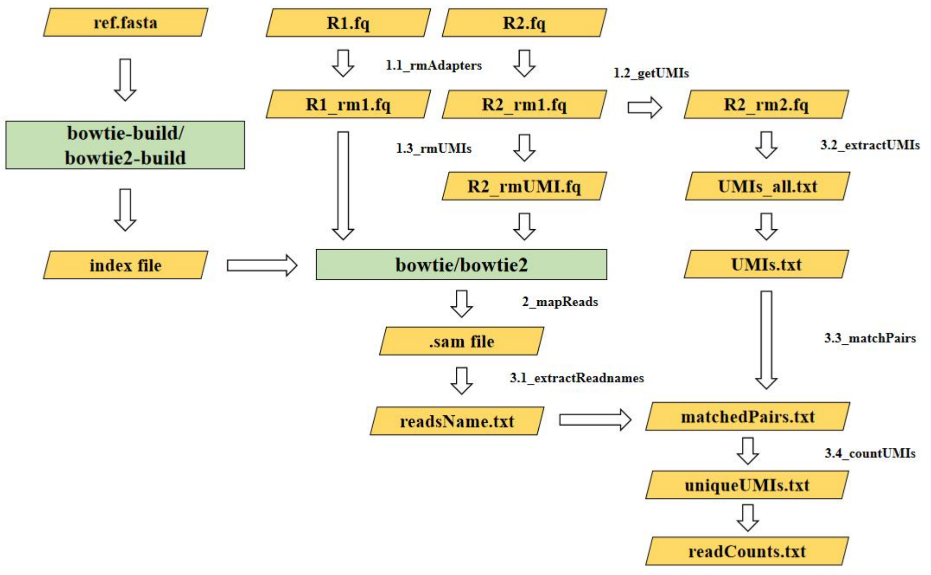

Fig2. Workflow of aavMPRA

2 Reference sequence

2.1 Overall design of library:

P5 + adapter1 + enhancer + adapter2 + UMI + adapter3 + P7[index]P7

AATGATA...CGATCTGCAGATGGCTCTTTGTCCTACACCCTCGGG...CCACAGCCCGGAGCGT

CTTCTGACCGTCGCTTTTTTTTTTTGGGCACAGGAAAGATACTTAGATCG......AGTCACTGA

CCAATCTCG...TGCTTG

2.2 Structure of read1(R1) and read2(R2):

R1 structure: adapter1 + enhancer_part

CAGATGGCTCTTTGTCCTACACCCTCGGG...

R2 structure: adapter3(reversed) + UMI(reversed) + adapter2(reversed) + enhancer_part(reversed)

AAGTATCTTTCCTGTGCCCAAAAAAAAAAAGCGACGGTCAGAAGACGCTCCGGGCTGTG...

2.3 Steps to correct reads:

1.1_rmAdapters: remove adapter1 and adapter3 (reversed)

R1_rm1 structure: enhancer_part

R2_rm1 structure: UMI(reversed) + adapter2(reversed) + enhancer_part(reversed)

1.2_getUMIs: extract UMI

R2_rm2 structure: UMI(reversed)

1.3_rmUMIs: remove UMI

R2_rmUMI structure: adapter2(reversed) + enhancer_part(reversed)

2.4 In the step 1.3_rmUMIs, to minimize the loss of sequence information during searching adapters by cutadapt, the adapter2 in R2 will not be trimmed. We map reads from R1_rm1 and R2_rmUMI to the reference. Hence, the reference sequence for building bowtie/bowtie2 index should contain sequence of adapters for mapping reads. The structure of reference sequence is:

adapter1 + enhancer + adapter2

To build bowtie/bowtie2 index, you should generate .fasta file consist of reference sequences.

3 Data processing

Spending time depending on read depth.

3.1 Ensure that your server runs Linux, we test it on CentOS Linux 7.

3.2 Clone the repository, then enter the aavMPRA directory and export the path of aavMPRA program to main path for execution by the following command:

```
git clone https://github.com/mengm5/aavMPRA.git
cd /your_path_to_aavMPRA/aavMPRA
export PATH=/your_path/aavMPRA/aavMPRA:$PATH
```

3.3 If you want to use conda for installation of dependencies, please install conda first if it is not on your server (https://docs.anaconda.com/miniconda/). Once conda is installed, create aavMPRA environment by the following command:

```
conda env create -n aavMPRA -f aavMPRA_environment.yml
```

It is a simple and quick way to install dependencies.

3.4 If users don't use conda to install dependencies, you can also install them locally. The list of dependencies is:

```
python=3.12.2
pip:
  - altgraph==0.17.4
  - numpy==2.0.0
  - packaging==24.1
  - pandas==2.2.2
  - pyinstaller==6.8.0
  - pyinstaller-hooks-contrib==2024.7
  - python-dateutil==2.9.0.post0
  - pytz==2024.1
  - six==1.16.0
  - tzdata==2024.1
bowtie=1.3.1
bowtie2=2.5.4
```

Make sure those dependencies can be executed directly.

3.5 Build the bowtie/bowtie2 index by the following command if it is not been built:

```
bowtie-build ref.fa /your_path_to_save_index/index_name
bowtie2-build ref.fa /your_path_to_save_index/index_name
```

The pre-build index is at /your_path/aavMPRA/index for testing.

3.6 Create a fastq_info.txt file. This file contains 4 columns:

1) the first column is paths where fastqs exist,
2) the second is the original names of fastqs,
3) the third is names that you want to change,
4) the forth is used to distinguish read1 (R1) and read2 (R2).

The example file is at /your_path/aavMPRA/data.

```
vim /your_path/aavMPRA/data/fastq_info.txt
Path   Fastq  Rename Read
/aavMPRA/data test1_1.fastq.gz  Test1  R1
/aavMPRA/data test1_2.fastq.gz  Test1  R2
/aavMPRA/data test2_1.fastq.gz  Test2  R1
/aavMPRA/data test2_2.fastq.gz  Test2  R2
```

3.7 Create a parameter.txt file. This file contains 6 columns:

1) the first column is names of steps,
2) the second is names of parameters,
3) the third is the assignment of parameters,
4) the forth is the description of parameters,
5) the fifth is the tools to which the parameters belong,
6) the sixth is the execution mode, the mutagenesis mode uses bowtie to map reads and the common mode uses bowtie2.

The common mode uses bowtie2 because the length of the common library sequence is 400bp, and the length of read1 and read2 can not completely cover the entire sequence. Therefore the bowtie2 can provide better mapping results when mapping longer reads. Meanwhile the mutagenesis mode uses bowtie due to the shorter sequence length of mutagenesis library (230bp).

The example file is at /your_path/aavMPRA/data. You can change the parameters manually for your requirement.

```
vim /your_path/aavMPRA/data/parameter.txt
Steps  Name   Parameter  Explanation   Tool   Mode
rmAdapters "-g"   GCAGATGGCTCTTTGTCCTA "5'  adapter  to  be  removed
from R1"  cutadapt  generic
rmAdapters "-G"   AAGTATCTTTCCTGTGCCCA "5'  adapter  to  be  removed
from R2"  cutadapt  generic
......
```

In our study, we primarily utilized the default parameters. Detailed descriptions of these parameters can be found on the official websites of cutadapt, bowtie, and bowtie2. Users may query these sites as needed to add, delete, or modify entries in the parameter.txt file according to our specified format.

Three essential parameters must be adjusted:

-g: Refers to adapter1 as mentioned previously.

-G: Refers to the reverse sequence of adapter3.

-A: Refers to the reverse sequence of adapter2.

Additionally, users can modify the number of CPU cores allocated to the process (e.g., -j for cutadapt, -p for bowtie/bowtie2) to reduce processing time.

3.8 Run the pipeline by the following command:

```
aavMPRA -o /output_path \
-f /path_to_fastq_info/fastq_info.txt \
-p /path_to_parameter/parameter.txt \
-m [mutagenesis/common] \
-i /path_to_index/index_name \
[--gz]
# output_path: absolute path is recommended.
# -m input mutagenesis or common.
# if the fastq file saved as .gz file, please add --gz.
```

3.9 The aavMPRA will generate .sh file sequentially to execute analysis:

1) 0_softlink.sh: read the fastq_info.txt and generate a file path that contains softlinks of fastqs with standard names.
2) 1.1_rmAdapters.sh, 1.2_getUMIs.sh, 1.3_rmUMIs.sh: correct reads structure by cutadapt.
3) 2_mapReads.sh: map reads to the bowtie/bowtie2 index.
4) 3.1_extractReadnames.sh, 3.2_extractUMIs.sh: extract readnames from .sam file and UMIs from fastqs.
5) 3.3_matchPairs.sh: combine readnames and UMIs.
6) 3.4_countUMIs.sh, 3.5_mergeSamples.sh: generate readcounts file and merge all readcounts of samples into one.

If you want to reanalyze from the middle of all steps and do not want to start from the beginning, it is recommended to modify these .sh files.

3.10 The aavMPRA will generate a list of directories of result:

1) 0_softlinks: softlinks of all samples (renamed).
2) 1_correctReads: fastqs with corrected reads of all samples.
3) 2_mapReads: sam files with mapped reads of all samples.
4) 3_readCounts: unique UMIs, readCounts files.

	input	output
1.1_rmAdapters.sh	xxx_R1.fastq xxx_R2.fastq	xxx_R1_rm1.fastq xxx_R2_rm1.fastq xxx_untrimed_1_rm1.fastq xxx_untrimed_2_rm1.fastq xxx_rmAdapters.log
1.2_getUMIs.sh	xxx_R1_rm1.fastq xxx_R2_rm1.fastq	xxx_R1_rm2.fastq xxx_R2_rm2.fastq xxx_untrimed_1_rm2.fastq xxx_untrimed_2_rm2.fastq xxx_getUMIs.log
1.3_rmUMIs.sh	xxx_R2_rm1.fastq	xxx_R2_rmUMI.fastq xxx_rmUMIs.log
2_mapReads.sh	xxx_R1_rm1.fastq xxx_R2_rmUMI.fastq	xxx.sam xxx_mapReads.log
3.1_extractReadnames.sh	xxx.sam	xxx_readsName.txt
3.2_extractUMI.sh	xxx_R2_rm2.fastq	xxx_UMIs_all.txt xxx_UMIs_rm.txt xxx_UMIs.txt
3.3_matchPairs.sh	xxx_readsName.txt xxx_UMIs.txt	xxx_matchedPairs.txt
3.4_countUMIs.sh	xxx__matchedPairs.txt	xxx_uniqueUMIs.txt xxx_readCounts.txt
3.5_mergeSamples.sh	xxx,yyy_readCounts.txt	mergedSamplesCount.txt

xxx_UMIs_all.txt contains all length of short sequences cut from xxx_R2_rm2.fastq, because the length of UMI in our design is 10bp. Hence we remove the ‘not-10bp-UMIs’ and save them into xxx_UMIs_rm.txt while we save the ‘10bp-UMIs’ into xxx_UMIs.txt for next step.