You must follow the steps below if you want to use these programs.

1. Transform the sequence file into a “fasta” format file.

Command: perl Fasta.pl {input file} {output file}.

1. Count the number of reads with correct length or no more than N base errors containing both substitution errors and indel errors.

Command: perl Valid\_Reads.pl {input file} {output file} {L} {N} {N}

# L: correct length; N: the maximum number of base indels.

1. Obtain the coverage and number of M0G0(mismatch base pair number equal to 0, and gap number equal to 0) and M1G1(mismatch base pair number equal to 1, and gap number equal to 1) and corresponding sequences of M0G0 and M1G1 separately.

Command: perl M0G0\_M1G1.pl {input file} {output file} {L} {M} {G}

# L: correct length; M: the number of reads with a mismatch; G: the number of reads with a indel.

1. Obtain the coverage and number of valid reads and corresponding sequences.

Command: perl Valid\_Coverage\_Number.pl {input file} {output file} {L}

# L: correct length.

1. Obtain payload (the DNA sequence with digital data)

Command: perl Obtain\_Payload.pl {input file} {output file}

1. Cluster the sequences with the same repeat

Command: perl Cluster.pl {input file} {output file}

1. Sort the sequences’ repeat number from biggest to smallest

Command: perl Sort.pl {input file} {output file}

1. Random select n1 percent of valid reads.

Command: perl Random\_Selection.pl {input file} {n1} {n2} {output file}

# n1: percentage of valid reads, n2: times of random selection.

1. Countthe number of missing sequences.

Command: perl Dropout.pl {input file}.

Download the code including every folder. Users must execute the command in a fixed path under the specific folders.

Require: Python 3, Perl 5.12, G++ compiler.(Win 10/Mac os/Linux)

1. **Encode (change the work-dir to encode folder)**
2. Compile C code under the ./lib.

Command: g++ ./lib/encoder-normal.cpp –o ./lib/encode

1. Python (Version 3.0 and above)

Command: python encode.py {input file} {output file}

1. The encode\_file\_meta.txt under the temp file folder will be used in the process of decoding and persevered well. Meanwhile, three files containing random\_bits\_2.csv, random\_bits\_rs.csv, and rs\_file.csv will also be used in the process of decoding and stored well.
2. **Cluster (change the work-dir to cluster folder)**
3. Random selection.

Command: perl 1.extract.pl {n} {n1} {input file}

# n: the number of sequenced reads; n1: the number of random selected reads.

1. The total files under the temp file folder are copied to the encode file folder of decode file folder.
2. **Decode (change the work-dir to decode folder)**
3. Compile C code under the ./lib
4. Command: ./lib/decoder; ./lib/repair
   1. Command 1: g++ ./lib/9\_repair.cpp –o ./lib/repair
   2. Command 2: g++ ./lib/decoder.cpp –o ./lib/decode
5. The encode\_file\_meta.txt in the temp file folder generated by the process of encoding is pasted in the encode file folder of decode file folder. Meanwhile, three files containing random\_bits\_2.csv, random\_bits\_rs.csv, and rs\_file.csv generated by the process of encoding are also pasted in the encode file folder of decode file folder.
6. Command 3: python decode.py ./encode {output file}
7. Remove the repeated binary file.

Command 4: perl ./lib/repeat\_remove\_binary\_2.pl

1. DNA sequences can be obtained for perfect decoding by random selection with multiple times.

Command 5: perl ./lib/decode\_random.pl ./temp n1 n2

# n1: the number of DNA sequences we encoded; n2: the number of times of random selection.

1. Decoding.

Command 6: absolute path\decode\lib/decode.exe ./temp(j)/decode\_file

# j = (0, 1, …, n2)

Note: Cannot contain Chinese characters