Here we provided codes of MEPCAL used in this study. To better monitor the running of MEPCAL, the process of encoding and decoding were assigned to several different programs with chained input and output. These input and output files were provided along with programs. You can voluntarily run these programs and get same output files.

Table 1 and Table 2 respectively listed the programs used for encoding and decoding and corresponding input/output files.

**NOTE**: We did not provide RaptorQ codec and RS codec here. Please go to the following URL to obtain these programs, and use them in accordance with what we declared. Actually, any compliant RS and RaptorQ codec can be used to complete the MEPCAL operation.

RaptorQ codec: https://github.com/BibbyLiang/RaptorQ-Codec

RS codec: https://github.com/tomerfiliba/reedsolomon

**Table 1 Programs used for encoding and input/output files.**

|  |  |  |  |
| --- | --- | --- | --- |
| Program | Input | Output | Note |
| pri\_redun.py | binary.txt | inf\_pri.txt |  |
| RQ\_enc.m | inf\_pri.txt | sRepsym.txt | In MATLAB |
| Seq\_screening.py | inf\_pri.txt & sRepsym.txt | screenedbase.txt |  |
| mencode.py | screenedbase.txt | finalbase.txt & index.txt |  |

The description of these files:

* binary.txt: the file containing binary stream of information to be stored
* inf\_pri.txt: the file containing information symbols and primary repair symbols (in 256-ray numbers)
* sRepsym.txt: the file containing secondary repair symbols (400×)
* screenedbase.txt: the file containing DNA sequences that had passed screening
* finalbase.txt: the file containing encoding result (DNA sequence)
* index.txt: the file containing indices of secondary repair symbols

**Table 2 Programs used for decoding and input/output files.**

|  |  |  |  |
| --- | --- | --- | --- |
| Program | Input | Output | Note |
| mdecode.py | finalbase.txt | encsym.txt |  |
| enc\_index.py | encsym.txt & index.txt | rqdec.txt |  |
| RQ\_dec.m | rqdec.txt | infsym.txt | In MATLAB |
| pri\_dec.py | infsym.txt | infsym\_real.txt | Optional |
| inver\_ray.py | infsym.txt | binary\_real.txt |  |
| Source\_decoding.py | binary\_real.txt or  infsym\_real.txt | information | source coding step |

The description of these files:

* encsym.txt: the file containing DNA symbols decoded by RS code
* rqdec.txt: the file containing DNA symbols and indices that can be directly used by RaptorQ decoder
* infsym.txt: the file containing DNA symbols decoded by RaptorQ code, which was supposed to be identical with inf\_pri.txt
* infsym\_real.txt: the file containing binary stream after outer RS decoding given some of the information groups were not successfully recovered

**Example of usage for encoding:**

* Run “pri\_redun.py”, you will get “Please enter the name of file containing binary sequence:”, type in “binary.txt”, you will get “inf\_pri.txt”, which contains information symbols and primary repair symbols.
* Run “RQ\_enc.m” in MATLAB with built-in input (“pri\_rep.txt”), and you will get “sRepsym.txt”, which contains secondary repair symbols.
* Run “Seq\_screening.py” with built-in inputs (“inf\_pri.txt” and “sRepsym.txt”), and the real-time results of sequence screening will be displayed (Pass? True/False). After all screening, you will get “screenedbase.txt”, which contains screened sequences).
* Run “mencode.py” with built-in input (“screenedbase.txt”), and you will get the encoding result of MEPCAL (DNA sequences, in “finalbase.txt”) along with an index file (“index.txt”) for RaptorQ decoding.

**Example of usage for decoding:**

* Run “mdecode.py” with built-in input (“finalbase.txt”), and you will get “encsym.txt”, which contains DNA symbols after inner RS decoding.
* Run “enc\_index.py” with built-in input (“encsym.txt” and “index.txt”), and you will get “rqdec.txt”, which contains DNA symbols and corresponding indices.
* Run “RQ\_enc.m” in MATLAB with built-in input (“rqdec.txt”), and you will get “infsym.txt”, which contains DNA symbols after RaptorQ decoding.
* If not all encoding groups were recovered, run “pri\_dec.py”, and you will get “Please type in the number of unrecovered encoding groups (0-37):”. Next, type in 1 or 2 or 3, and you will get “Please type in the order of unrecovered encoding groups:”. If the 12th , 20th and 34th encoding groups were not recovered, type in “11”-Enter-“19”-Enter-“33”-Enter, and you will get “infsym\_real.txt”, which contains information symbols after outer RS decoding.
* Now you get the complete information, albeit in the form of 256-ray numbers. Run “inver\_ray.py”, and you will get the binary stream. Next, run “Source\_decoding.py”, you will get the original information.

**Note**: the file of decoding dictionary used in this study is “decdic\_new.txt”.