CfBamSim: A simulation tool for cell-free DNA methylation sequencing data

A.1 Stimulation details

In the following we detail the simulation steps embedded in the CfDNA methylation data simulator.

S.1 Provide the genomic region corresponding to the simulated data: In the course of cfDNA methylation research, numerous researchers are often engaged in simulating cfDNA methylation BAM files. Depending on their specific research requirements, some researchers have the necessity to pre - screen genomic regions that match the characteristics of the sequencing data ahead of time and present them in the following format (default region file: bin_common_position.csv).

S.2 Provide the original BAM file: (i) Simulation of BAM files for normal plasma samples: It is required to provide at least two BAM files obtained by aligning the original methylation sequencing data of normal plasma. (ii) Simulation of BAM files for plasma samples of diseased patients: It is required to provide at least two BAM files obtained by aligning the original methylation sequencing data of normal plasma and at least two BAM files obtained by aligning the original methylation sequencing data of plasma samples from diseased patients. The format of the BAM file is as follows:

S.3 Sampling methods: The cfDNA simulation tool provides two distinct sampling methods, uniform sampling and non-uniform sampling, to mimic different biological scenarios.

(i)Uniform Sampling

In uniform sampling, the proportion of reads drawn from the normal and diseased BAM files is consistent across all genomic bins. This approach ensures that the mixing ratios remain stable throughout the genome, providing a controlled environment for data simulation. The mixing ratios are defined within user-specified boundaries (disease_ratio_min and disease_ratio_max). Reads are sampled proportionally from the base (normal) and mix (diseased) samples, creating a uniform distribution of reads across all bins. Uniform sampling is ideal for generating benchmark datasets with predictable characteristics, making it suitable for testing computational tools and establishing a baseline for analysis.

(ii)Non-Uniform Sampling

In non-uniform sampling, the mixing ratio varies dynamically across genomic bins, introducing heterogeneity into the simulated data. For each bin, the proportion of diseased reads is determined randomly, often with weighted probabilities, to reflect realistic cfDNA scenarios where diseased cell contributions vary across regions. This variability allows for a more biologically relevant dataset, capturing complexities observed in real-world cfDNA samples. Non-uniform sampling better models biological variability, providing more realistic data for evaluating tools designed to analyze heterogeneous cfDNA samples or detect tissue-of-origin signals.

By allowing users to choose between these two sampling methods, the tool ensures flexibility in simulating datasets for diverse research applications, from controlled experiments to complex biological studies.

S.4 Stimulation methods: (i) Simulate diseased plasma samples: Replace the fragments by uniform sampling or non-uniform sampling at a ratio of 0.5% - 10% and rewrite them into the Bam file. (ii) Simulation of normal plasma samples: Replace the fragments by uniform sampling at a ratio of 15% - 50% and rewrite them into the Bam file (Figure S1).

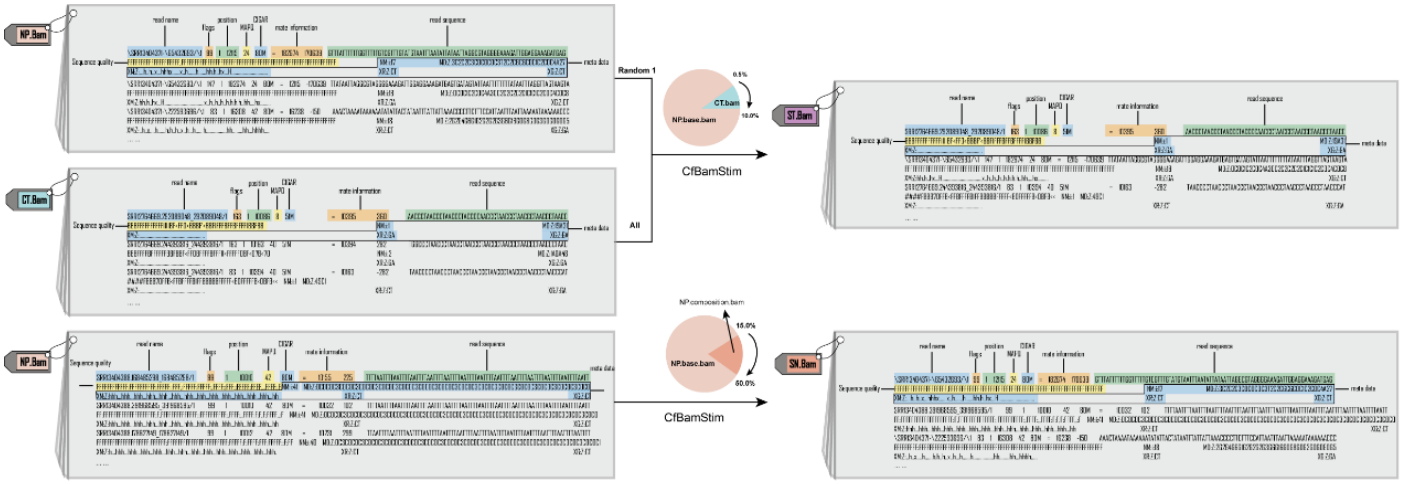

Fig.S1. BAM file rewrite process

S.5 Other Functions: (i) Annotation File: The annotation file specifies genomic regions such as exons, introns, and regulatory elements. It is used to enhance the biological realism of the simulation by applying different sampling or mixing ratios to specific functional regions. (ii) Hotspot File: The hotspot file highlights regions of the genome with a higher likelihood of mutations or disease-associated changes, such as oncogenic mutations or specific methylation hotspots. (iii) Length Distribution File: The length distribution file provides the empirical fragment length distribution of cfDNA. It is used to filter or sample reads according to realistic fragment sizes, ensuring that the simulated BAM files match real-world cfDNA data.

A.2 Help Documentation

This tool generates simulated BAM files for cfDNA methylation sequencing. It supports normal and diseased sample simulation with features like uniform and non-uniform sampling, hotspot region prioritization, length distribution filtering, and multithreading.

H.1 Required Input Files

(i) Input Interval File:

- CSV format.
- Contains genomic intervals for sampling.
- Format:

```
chrom,start,end
chr1,10000,20000
chr1,20001,30000
chr2,5000,15000
chr2,15001,25000
```

(ii) BAM Files:

- Ensure that all BAM files are indexed (.bai files required).
- Format:

```
\SRR13404371-\1/\1  99      4       60194894  42      80M     =       60194977  163
        ATTTAAATTATTAATTTATTTAGATTTTGGTTTAAGGAGAAGTGATTTAAATATTAGTTTTTTTTATTATTAGTTTAAAG
        FFFFFFFFFFFFFFFFFF:FFFFFFFFFFFFFFFFFFFFFFFF:F:FFFFFFFFFFFFFFFFFFFFFFFFFFFFF      NM:i:15
        MD:Z:8C7C2C7C0A1C0C15C5C0C2C3C1C5C5C4
        XM:Z:........h.......h..h.......x..hh..............h.....hx..h...h.h.....h.....h....   XR:Z:CT
        XG:Z:CT
\SRR13404371-\1/\1  147     4       60194977  42      80M     =       60194894  -163
        GAAAAAAATTTAGTGATTTTTGAGTTTTATTTTTTTATTTTTAAAAGGAGTTTGATATAATTGAGTTTATTTAGTGTTTA
        FFFFFFFFFFFFFFF,FFFFFFFFFFF:FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF:FFFFFFFFFFFF      NM:i:12
        MD:Z:10C6C6C2C1C3C1C1C0C26C5C6C1
        XM:Z:..........x......h......h..h.h...h.h.hh.........................h.....x......h.   XR:Z:GA
        XG:Z:CT
\SRR13404371-\2/\1  99      6       35867429  42      80M     =       35867525  176
        TTGAATTATGTTTTATTTAAAAGTATATGGTTTTTTGAGTATTGTTATTTATGTGTGAATTATTTTATTATAAATTTTAA
        FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF:FFFFFFFFFFF:FFFFFFF:FFFFFFFFFFFFFFFFFFF      NM:i:13
        MD:Z:0C4C0C4C0C0C9C6C1C8C3C14C3C15
        XM:Z:x....hh....hhh.........h......h.h........x...h..............h...h..............   XR:Z:CT
        XG:Z:CT
```

(iii) BAM File Information:

- `bam_file_info.txt` in the input directory.
- Tab-delimited format.
- Format:

```
filename        category
normal_sample1.bam      normal
normal_sample2.bam      normal
diseased_sample1.bam    diseased
diseased_sample2.bam    diseased
```

- ✓ `filename`: BAM file name.
- ✓ `category`: `normal` or `diseased`.

H.2 Optional Input Files

(i) Hotspot File:

- CSV format.
- Columns: `bin` (region) and `weight`.
- Format:

```
bin,weight
chr1:1-20000,2.0
chr1:20001-40000,1.5
chr2:1-20000,3.0
```

(ii) Annotation File:

- Tab-delimited format.
- Format:

```
chrom   start   end   type
chr1    1000    5000  exon
chr1    6000    10000 intron
```

- ✓ `type`: `exon`, `intron`, etc.

(iii) Fragment Length Distribution File:

- CSV format.
- Format:

```
length   probability
150      0.3
160      0.4
170      0.3
```

- ✓ `length`: Fragment length.
- ✓ `probability`: Probability for the length.

H.3 Parameters

Argument	Type	Default	Description
`--input_interval_file`	`str`	Required	Path to the input interval file.
`--input_dir`	`str`	Required	Directory containing input BAM files and `bam_file_info.txt`.
`--output_dir`	`str`	Required	Directory to save the output simulated BAM files.
`--simulated_sample_category`	`str`	`diseased`	Category of the simulated sample (`normal` or `diseased`).
`--sampling_method`	`str`	`uniform`	Sampling method (`uniform` or `non-uniform`).
`--window_size`	`int`	`20000`	Size of the genomic window for sampling.
`--disease_ratio_min`	`float`	None	Minimum ratio of disease reads in a bin.
`--disease_ratio_max`	`float`	None	Maximum ratio of disease reads in a bin.
`--target_coverage`	`float`	None	Target genome coverage for the simulated BAM files.
`--target_depth`	`int`	None	Target sequencing depth for the simulated BAM files.
`--num_simulated_samples`	`int`	`2`	Number of simulated samples to generate.
`--num_threads`	`int`	`2`	Number of threads to use for parallel processing.
`--sequencing_type`	`str`	`single-end`	Type of sequencing to simulate (`single-end` or `paired-end`).
`--hotspot_file`	`str`	`''` (Optional)	Path to the hotspot file for prioritized sampling in specified regions.
`--annotation_file`	`str`	`''` (Optional)	Path to the genome annotation file for feature-based sampling adjustments.
`--length_distribution_file`	`str`	`''` (Optional)	Path to the length distribution file for controlling fragment size sampling.

H.4 Usage Examples

(i) Example 1: Simulate 2 Normal Plasma Samples

- ✓ Command

```
python simulate_disease_samples_updated.py \
    --input_interval_file ./bin_common_position.csv \
    --input_dir ./input_bams \
    --output_dir ./output_samples \
    --simulated_sample_category normal \
    --num_simulated_samples 2 \
    --num_threads 4 \
    --window_size 20000
```

✓ Explanation:

a) Simulates 2 normal plasma samples.

b) Uses `bin_common_position.csv` for the genomic interval information.

c) Reads input BAM files from `./input_bams` directory.

d) Saves output to the `./output_samples` directory.

e) Simulates with a window size of `20,000 bp`.

f) Uses `4 threads` for parallel processing.

(ii) Example 2: Simulate 3 Diseased Plasma Samples with Custom Coverage and Depth

✓ Command

```
python simulate_disease_samples_updated.py \
    --input_interval_file ./bin_common_position.csv \
    --input_dir ./input_bams \
    --output_dir ./output_samples \
    --simulated_sample_category diseased \
    --num_simulated_samples 3 \
    --target_coverage 50 \
    --target_depth 200 \
    --num_threads 6
```

✓ Explanation:

a) Simulates 3 diseased plasma samples.

b) Sets the target genome coverage to `50x` and target depth to `200`.

c) Uses `6 threads` for parallel processing.

d) Reads from `./input_bams` and saves to `./output_samples`.

(iii) Example 3: Simulate with Custom Hotspot Regions and Annotations

✓ Command

```
python simulate_disease_samples_updated.py \
    --input_interval_file ./bin_common_position.csv \
    --input_dir ./input_bams \
    --output_dir ./output_samples \
    --simulated_sample_category diseased \
    --num_simulated_samples 2 \
    --hotspot_file ./hotspot_regions.csv \
    --annotation_file ./annotations.txt \
    --num_threads 4
```

✓ Explanation:

a) Simulates 2 diseased plasma samples.

b) Incorporates hotspot regions defined in `hotspot_regions.csv`.

c) Uses functional annotations from `annotations.txt`.

d) Processes in parallel using `4 threads`.

(iv) Example 4: Simulate with Fragment Length Distribution

✓ Command

```
python simulate_disease_samples_updated.py \
    --input_interval_file ./bin_common_position.csv \
    --input_dir ./input_bams \
    --output_dir ./output_samples \
    --simulated_sample_category diseased \
    --num_simulated_samples 1 \
    --length_distribution_file ./length_distribution.csv
```

✓ Explanation:

a) Simulates 1 diseased plasma sample.

b) Fragment lengths are sampled based on the probabilities in `length_distribution.csv`.

c) Reads input BAMs from `./input_bams` and outputs results to `./output_samples`.

(v) Example 5: Simulate Paired-End Sequencing Data

✓ Command

```
python simulate_disease_samples_updated.py \
    --input_interval_file ./bin_common_position.csv \
    --input_dir ./input_bams \
    --output_dir ./output_samples \
    --simulated_sample_category diseased \
    --num_simulated_samples 2 \
    --sequencing_type paired-end
```

✓ Explanation:

a) Simulates 2 diseased plasma samples with paired-end sequencing.

b) Processes BAM files from `./input_bams`.

c) Saves the output in `./output_samples`.

These examples provide a comprehensive overview of the tool's flexibility and utility in simulating cfDNA samples under diverse conditions.

H.5 Output Files

- Simulated BAM files: Saved in the output directory with names like:

```
Simulated_diseased_sample_1.bam
Simulated_normal_sample_2.bam
```

- Log file:

```
simulation.log
```

  - ✓ Contains details of the simulation process.

H.6 Notes

(i) Default Behavior:

- If optional files (hotspot, annotation, length distribution) are not provided, the tool simulates samples using default settings.
  - ✓ Ensure the BAM files are indexed (`.bai`).
  - ✓ Verify file paths and formats for all inputs.
  - ✓ `--hotspot_file`, `--annotation_file`, and `--length_distribution_file` are optional.

(ii) Validation:

- Ensure BAM files are indexed and listed correctly in `bam_file_info.txt`.

(iii) Dependencies:

- Python 3.x
- pysam, pandas, numpy

(iv) Performance:

- Use more threads (`--num_threads`) for faster processing on multi-core systems.
- Adjust `--window_size` to balance memory usage and processing time.

(v) Debugging:

- Check simulation.log for warnings and errors.
- Ensure all required input files (`bam_file_info.txt`, BAM files, etc.) are correctly formatted and available.

This guide should assist in effective utilization of the cfDNA simulation tool for research and analysis.

Contact

For questions or issues, please contact the tool developer (zhihui@ems.hrbmu.edu.cn).