Supplementary Material

Wei Zhang Lei Wei Jiaqi Huang Bixi Zhong Jiaqi Li Hanwen Xu
Shuying He Yu liu Juhong Liu Hairong Lv Xiaowo Wang

2021-01-11

Contents

1 Introduction

cfDNApipe is an integrated pipeline for analyzing cell-free DNA WGBS/WGS data. It contains many cfDNA quality control and feature extraction algorithms. Also we collected some useful cell free DNA references and provide them here.

The whole pipeline is designed based on flow graph principle. Users can use the build-in pipelines for single/paired end WGBS/WGS data or build their own analysis pipeline from any intermediate data like BAM/BED files. The main functions are shown in Figure 1.

Figures 2 and 3 shows the up- and down-stream relationship between functions in WGS and WGBS data processing.

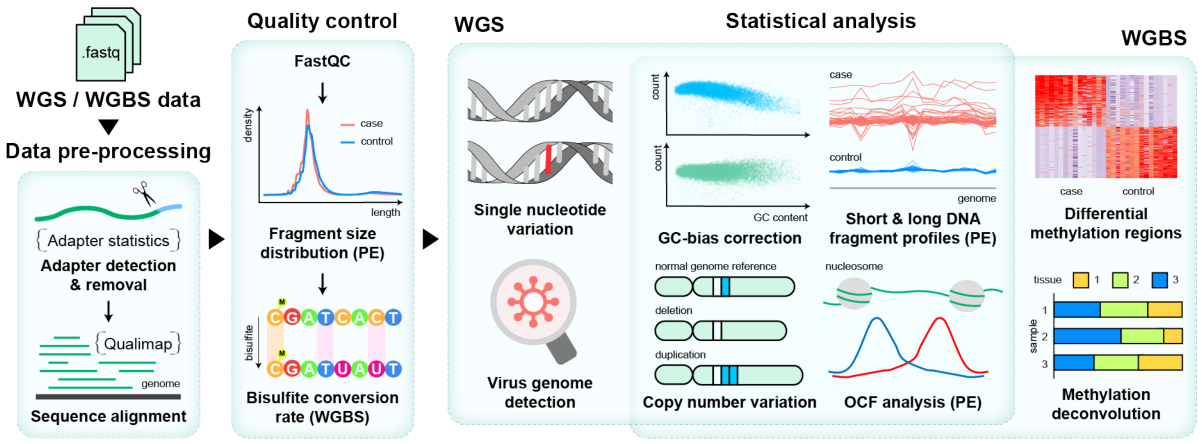

Figure 1: cfDNApipe Functions

2 Package Installation and Loading

2.1 Download and Installation

The popular WGBS/WGS analysis software are released on Unix/Linux system, based on different program language, like Bowtie2 (Langmead and Salzberg, 2012) and Bismark (Krueger and Andrews, 2011). Therefore, it's very difficult to rewrite all the software in one language. Fortunately, conda/bioconda program has collected many prevalent bioinformatics related python modules and software, therefore we can install all the dependencies through conda/bioconda.If you did not install conda before, please follow this tutorial to install conda first.

After installation, you can create a new virtual environment for cfDNA analysis. Virtual environment management means that you can install all the dependencies in this virtual environment and delete them easily by removing the whole virtual environment.

2.2 Create Environment and Install Dependencies

We tested our pipeline using different version of software and provide an environment yml file for users. Users can download this file and create the environment in one command line without any software conflict.

First, please download the yml file.

```
wget https://raw.githubusercontent.com/Honchkrow/cfDNApipe/master/environment.yml
```

Then, run the following command to create an virtual environment named cfDNApipe. The environment will be created and all the dependencies as well as the latest cfDNApipe will be installed.

```
conda env create -n cfDNApipe -f environment.yml
```

Note: The environment name can be changed by replacing "-n cfDNApipe" to "-n customized_name".

2.3 Enter Environment and Use cfDNApipe

Once the environment is created, user can enter environment using the following command.

```
conda activate cfDNApipe
```

Now, just open python and process **cell free DNA WGBS/WGS paired/single end** data.

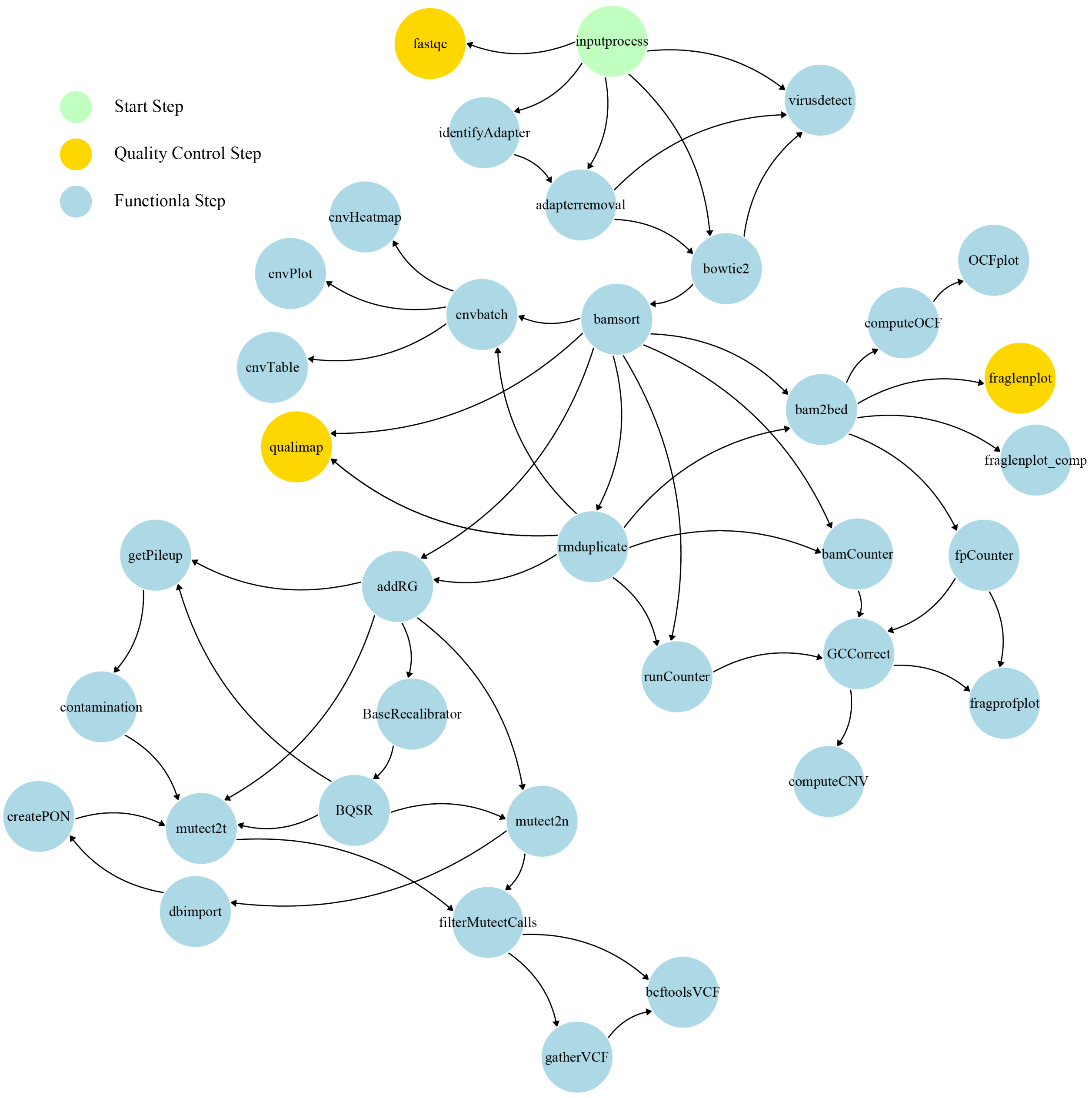

Figure 2: WGS Dataflow Overview

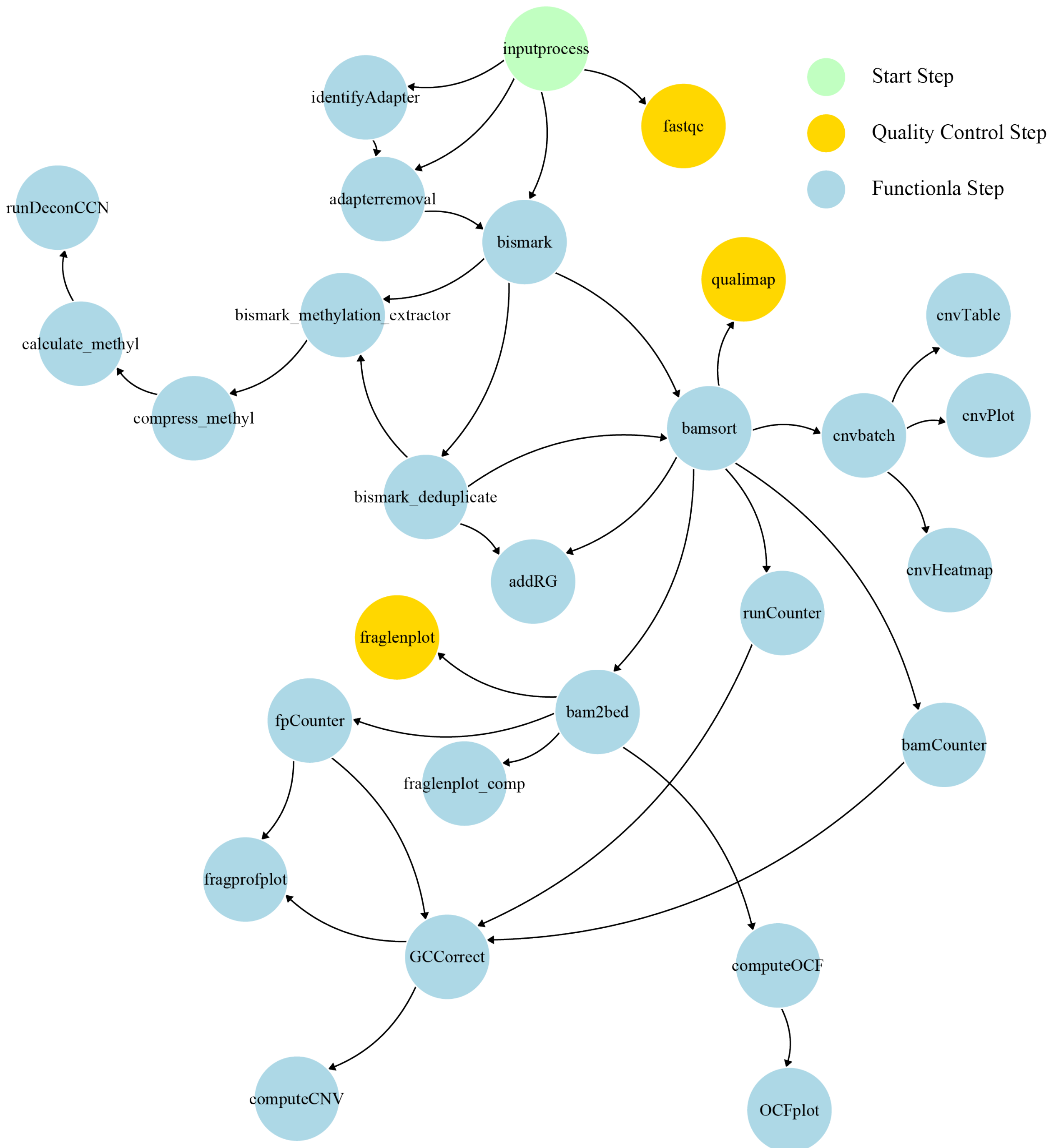

Figure 3: WGBS Dataflow Overview

3 Functional Summary

The whole workflow for cfDNApipe can be divided into two parts, raw data processing and statistical analysis.

3.1 Raw Data Processing and Quality Control

In this part, cfDNApipe offers functions to process raw sequencing data saved in FASTQ format to aligned BAM or BED files. It wraps FASTQC (Andrews and others, 2010) for pre-alignment raw sequencing data quality control. AdapterRemoval (Schubert *et al.*, 2016) is used for adapter detection and removal. Bowtie2 (Langmead and Salzberg, 2012) and Bismark (Krueger and Andrews, 2011) are adopted for read alignment and BAM format output will be generated. In addition, cfDNApipe calls different duplication removal tools for WGS and WGBS data (Krueger and Andrews, 2011; McKenna *et al.*, 2010). The BAM format output can be converted to BED format if needed. Post-alignment quality control will be executed by Qualimap (Okonechnikov *et al.*, 2016), which reports basic information and statistics for the alignment data such as genome coverage and GC content. What's more, cfDNApipe can sort, index, group the aligned read for the downstream statistical analysis (Li *et al.*, 2009; McKenna *et al.*, 2010).

3.2 Statistical Analysis

In this part, cfDNApipe provides multiple state-of-the-art statistical analysis modules for cfDNA data. Methylation level can be culculated for genome regions and methylation signal will be deconvoluted to reveal the component changes in cfDNA (Feng *et al.*, 2019). The differential methylated regions will be identified by cfDNApipe for further analysis (Kang *et al.*, 2017; Li *et al.*, 2018). Large-scale CNV (arm-level) and small-scale CNV (bin-level) are calculated to reveal genomic gains and losses (Jiang *et al.*, 2015; Supernat *et al.*, 2018; Talevich *et al.*, 2016; Huang *et al.*, 2019; Chen *et al.*, 2019; Chan *et al.*, 2013; Li *et al.*, 2017). cfDNA fragmentation analysis aims to demonstrate alters of DNA fragment length and disorder of fragmentation pattern (Jiang *et al.*, 2015; Mouliere *et al.*, 2018). Orientation-aware cfDNA fragmentation (OCF) analysis measures the differential phasing of upstream and downstream fragment ends in tissue-specific open chromatin regions (Sun *et al.*, 2019). Besides, some functions are specific to cfDNA WGS data. Virus related to some specific diseases like Hepatocecullar carcinoma and chronic hepatitis B virus (HBV) can be detected (Kim *et al.*, 2016). Somatic and germline mutations is identified by comparing the sequence of DNA with that in control samples or default reference (Cai *et al.*, 2019; Zviran *et al.*, 2020).

4 Quick Start for Preset Pipeline

cfDNApipe provides easy-to-use and friendly preset pipeline for cfDNA WGS/WGBS data. Users only need to provide the input raw sequencing files (FASTQ format), genome reference folder and output folder, then the program will do everything automatically. When the analysis is finished, an pretty HTML report will be generated for demonstrating quality control and statistical analysis results. Here, we provide 2 examples for **single group analysis** and **case-control analysis**.

4.1 Single Group Analysis

Here, we show a single group analysis procedure for single end WGBS data. A runnable small dataset can be download from ***. All the samples should be put in a folder **without any other files**.

First load cfDNApipe and set global reference parameters for pipeline.

```
from cfDNApipe import *

pipeConfigure(
    threads=60,
    genome="hg19",
    refdir=r"reference_genome/hg19",
    outdir=r"output/single_WGBS",
    data="WGBS",
    type="single",
    build=True,
    JavaMem="10g",
)
```

Note: The download procedure is always time-consuming. cfDNApipe can detect the reference files which are already existed in refdir. Therefore, users can employ already established reference without rebuilding. For instance, users can just put hg19.fa and bowtie2 related files into refdir and cfDNApipe will not download and rebuild them again. Other reference files can be got from here. Downlaoding, uncompressing and putting them into refdir will be much faster.

Second, run the processing pipeline.

```
res = cfDNAWGBS(
    inputFolder=r"path_to_WGBS_SE",
    idAdapter=True,
    rmAdapter=True,
    rmAdOP={"--gzip": True},
    dudup=True,
    CNV=True,
    armCNV=True,
    fragProfile=True,
    verbose=True,
    report=True,
)
```

Now, you can drink a cup of ceffe and wait for finishing.

4.2 Case Control Analysis

Here, we show a case-control analysis procedure for paired end WGS data. A runnable small dataset can be download from ***. Each group samples should be put in each folder without any other files.

First load cfDNApipe and set global reference parameters for pipeline.

```
from cfDNApipe import *

pipeConfigure2(
    threads=60,
    genome="hg19",
    refdir="reference_genome/hg19",
    outdir="output/paired_WGS",
    data="WGS",
    type="paired",
    JavaMem="8G",
    case="cancer",
    ctrl="normal",
    build=True,
)
```

Second, run the processing pipeline.

```
a, b = cfDNAWGS2(
    caseFolder="path_to_data/case",
    ctrlFolder="path_to_data/ctrl",
    caseName="case",
    ctrlName="ctrl",
    idAdapter=True,
    rmAdapter=True,
    rmAdOP={"--gzip": True},
    bowtie2OP={"-q": True, "-N": 1, "--time": True},
    dudup=True,
    CNV=True,
    armCNV=True,
    fragProfile=True,
    OCF=True,
    report=True,
    verbose=False,
)
```

Now, you can wait for finish running.

5 Customized Pipeline For Pipeline Function Verification

cfDNApipe integrates several state-of-the-art statistical models. Here, we illustrate how to use customized pipeline in cfDNApipe to perform these analysis.

The analyses performed below are start with aligned BAM or BED files. The dataset is from paper “Lengthening and shortening of plasma DNA in hepatocellular carcinoma patients” with accession number EGAS00001001024. Users can process the raw dataset following section 4.2 and take the intermediate results from step “rmduplicate” and “bam2bed” to reproduce results in section 5.1~5.4.

5.1 Large-Scale CNV in HCC patient

CNV is a common phenomenon appears in kinds of cancer types. CNV is also detected from cfDNA WGS data reported by (Jiang *et al.*, 2015). However, due to the low circulating tumor DNA (ctDNA) concentration, the aggregated signal in arm-level reveals more significant copy number changes.

Here, using the dataset from (Jiang *et al.*, 2015), we will perform the Large-scale (arm-level) CNV to reveal the difference between HCC patients and the healthy.

First, set global parameters and get the aligned read files.

```
from cfDNApipe import *
import glob

pipeConfigure2(
    threads=20,
    genome="hg19",
    refdir=r"reference_genome/hg19",
    outdir=r"output/pcs_armCNV",
    data="WGS",
    type="paired",
    JavaMem="8G",
    case="cancer",
    ctrl="normal",
    build=True,
)

verbose = False

case_bam = glob.glob("path_to_data/HCC/*.bam")
ctrl_bam = glob.glob("path_to_data/CTR/*.bam")
```

Code above will generate the output folders for case and control based on the flag set in function "pipeConfigure2". The output folder structure is introduced in cfDNApipe homepage.

Second, correct GC bias for case and control sample.

```
# case
switchConfigure("cancer")
case_bamCounter = bamCounter(
    bamInput=case_bam, upstream=True, verbose=verbose, stepNum="case01"
)
case_gcCounter = runCounter(
    filetype=0, upstream=True, verbose=verbose, stepNum="case02"
)
case_GCCorrect = GCCorrect(
    readupstream=case_bamCounter,
    gcupstream=case_gcCounter,
    verbose=verbose,
    stepNum="case03",
)

# ctrl
switchConfigure("normal")
ctrl_bamCounter = bamCounter(
    bamInput=ctrl_bam, upstream=True, verbose=verbose, stepNum="ctrl01"
)
ctrl_gcCounter = runCounter(
    filetype=0, upstream=True, verbose=verbose, stepNum="ctrl02"
)
ctrl_GCCorrect = GCCorrect(
    readupstream=ctrl_bamCounter,
    gcupstream=ctrl_gcCounter,
    verbose=verbose,
    stepNum="ctrl03",
)
```

Users can also check the GC corrected results in each folder with suffix "GCCorrect". Figure of HCC patient sample "H228" is shown in Figure 2.

Finally, compute z-score for CNV signal and generate overall illustration for autosomes.

```
switchConfigure("cancer")
res_computeCNV = computeCNV(
    caseupstream=case_GCCorrect,
```

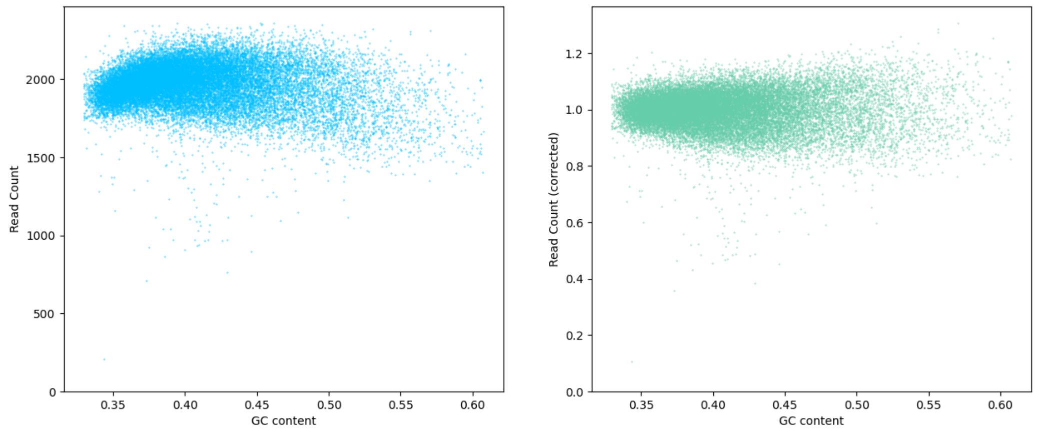

Figure 4: GC content correction for sample H228

```
    ctrlupstream=ctrl_GCCorrect,
    stepNum="ARMCNV",
    verbose=verbose,
)
```

Users can find the final Z-score file and the results are shown in Figure 3.

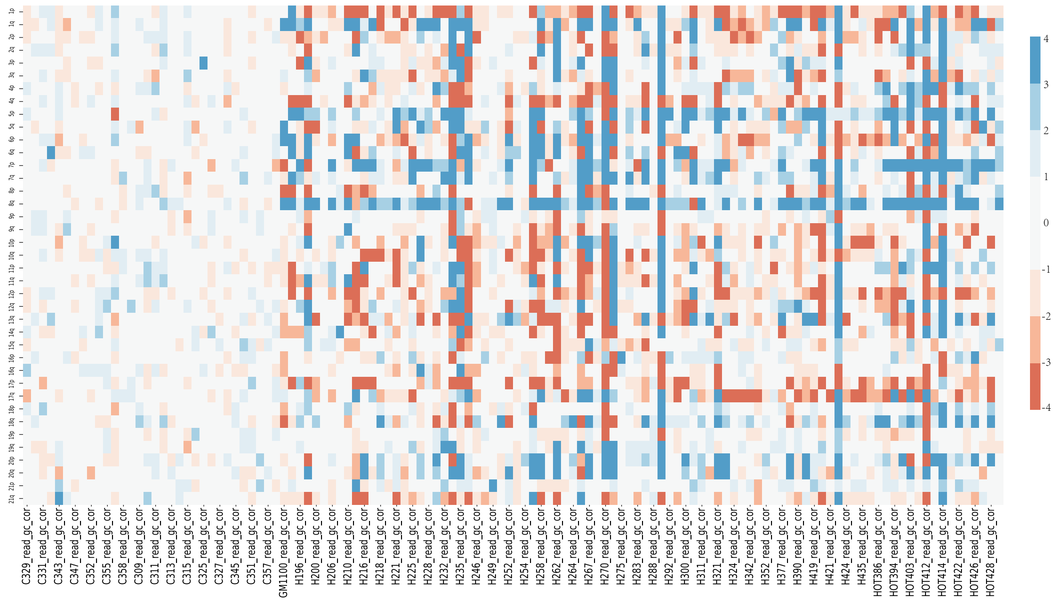

Figure 5: CNV Z-score for all samples

For HCC patients, the former work has reported that p,q arm in chromatin 1 and 8 shows significant gain and loss. Plotting p,q arm in chromatin 1 and 8, we can see the similar results between Figure 4 and Fig.2 of former work (Jiang *et al.*, 2015).

Note: The standardization method adopted by cfDNApipe is different from the original paper, but this does not influence the results.

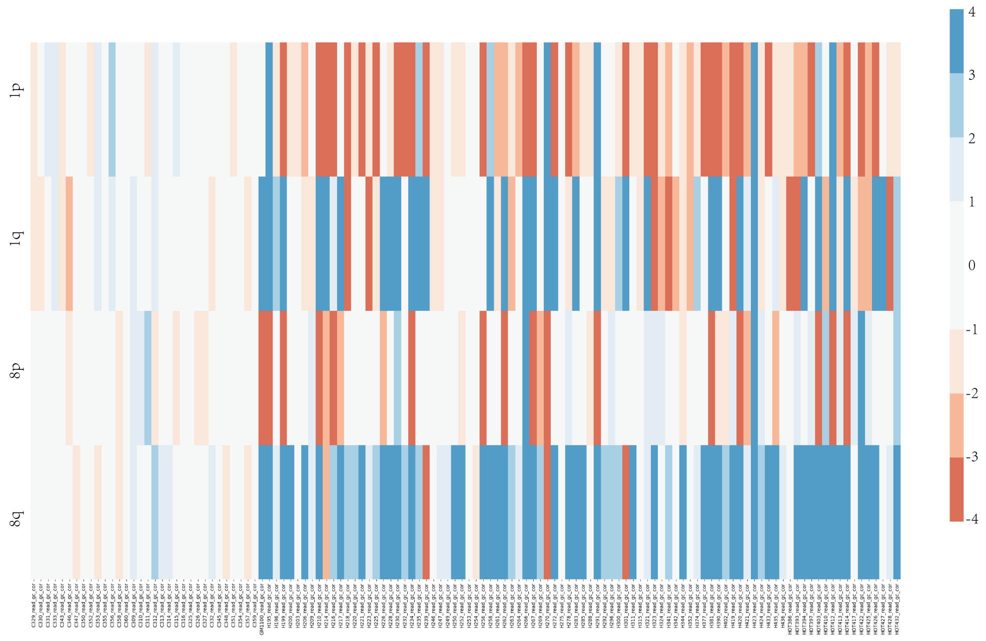

Figure 6: CNV Z-score in chromatin 1 and 8 for all samples

5.2 Fragment Length Analysis of cfDNA

The size distribution of cfDNA fragments shows some intrinsic nature in the formation of cfDNA (Jiang *et al.*, 2015). For example, periodicities corresponding to nucleosomes (~147bp) and chromatosomes (nucleosome + linker histone; ~167bp) have been noticed (Snyder *et al.*, 2016). What's more, circulating tumor DNA exists in plasma of patient with cancer. Lots of studies has shown that plasma DNA molecules released by tumors is shorter than that released by normal tissue (Jiang *et al.*, 2015; Mouliere *et al.*, 2018). cfDNApipe takes this information as an important signature of plasma DNA and can generate distribution plot for every samples as well as comparison for different groups.

Here, we shows how to generate length distribution for case control samples and compare the difference.

First, set global parameters and get the aligned read files in bed format.

```
from cfDNApipe import *
import glob

pipeConfigure(
    threads=20,
    genome="hg19",
    refdir=r"reference_genome/hg19",
    outdir=r"output/pcs_fraglen",
    data="WGS",
    type="paired",
    JavaMem="10G",
    build=True,
)
```

```
verbose = False

case_bed = glob.glob("path_to_data/HCC/*.bed")
ctrl_bed = glob.glob("path_to_data/CTR/*.bed")
```

Second, using the following one command to generate figures for every sample.

```
res_fraglenplot_comp = fraglenplot_comp(
    casebedInput=case_bed, ctrlbedInput=ctrl_bed, caseupstream=True, verbose=verbose)
```

Users can find figures for every sample (figures not show) and comparison results like below.

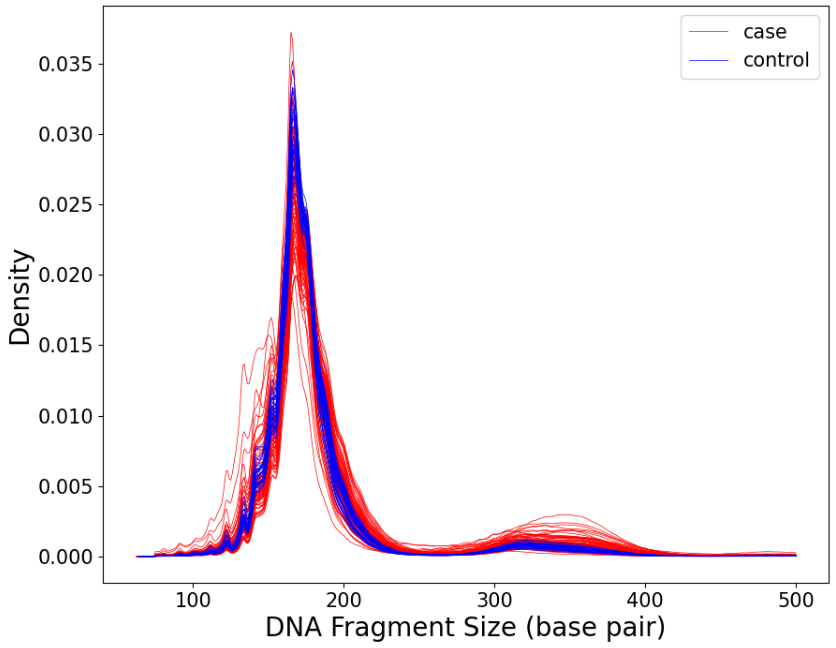

Figure 7: cfDNA fragment size distribution between HCC patients (case) and the healthy (control)

5.3 Orientation-Aware cfDNA Fragmentation Analysis

Cancerous cells and normal tissue cell shows different chromatin accessibility therefore different cells release cfDNA from different genome location. Digested by enzyme, cfDNA from different tissue shows different chromatin accessibility related signal. Orientation-aware cfDNA fragmentation (OCF) analysis is to quantify this signal and reveal the origin of cfDNA (Sun *et al.*, 2019).

Here, we shows how to perform OCF analysis for case control samples and compare the difference.

First, set global parameters and get the aligned read files in bed format.

```
from cfDNApipe import *
import glob

pipeConfigure2(
    threads=20,
    genome="hg19",
    refdir=r"reference_genome/hg19",
    outdir=r"output/pcs_armCNV",
    data="WGS",
    type="paired",
    JavaMem="8G",
    case="cancer",
    ctrl="normal",
    build=True,
)

verbose = False

case_bed = glob.glob("path_to_data/HCC/*.bed")
ctrl_bed = glob.glob("path_to_data/CTR/*.bed")
```

Second, using the following commands to calculate OCF value and compare OCF value between HCC patients (case) and the healthy (control).

```
# case
switchConfigure("cancer")

res_computeOCF = computeOCF(
    casebedInput=case_bed,
    ctrlbedInput=ctrl_bed,
    caseupstream=True,
    verbose=verbose,
)
res_OCFplot = OCFplot(upstream=res_computeOCF, verbose=verbose)
```

Users can find comparison figure for HCC patients (case) and the healthy (control). The validate results can be found in previous work Fig.4 and Fig.5 B and C (Sun *et al.*, 2019).

5.4 Enhanced Fragmentation Analysis Reveals Disorder of cfDNA

The former work has announced disorder in arm-level CNV, and (Mouliere *et al.*, 2018) reported another disorder in various cancer types. Next, we shows how to perform the enhanced fragmentation analysis for case control samples and compare the difference.

First, set global parameters and get the aligned read files in bed.gz format.

```
from cfDNApipe import *
import glob

pipeConfigure2(
    threads=20,
    genome="hg19",
    refdir=r"reference_genome/hg19",
    outdir=r"output/pcs_armCNV",
    data="WGS",
    type="paired",
    JavaMem="8G",
    case="cancer",
    ctrl="normal",
    build=True,
```

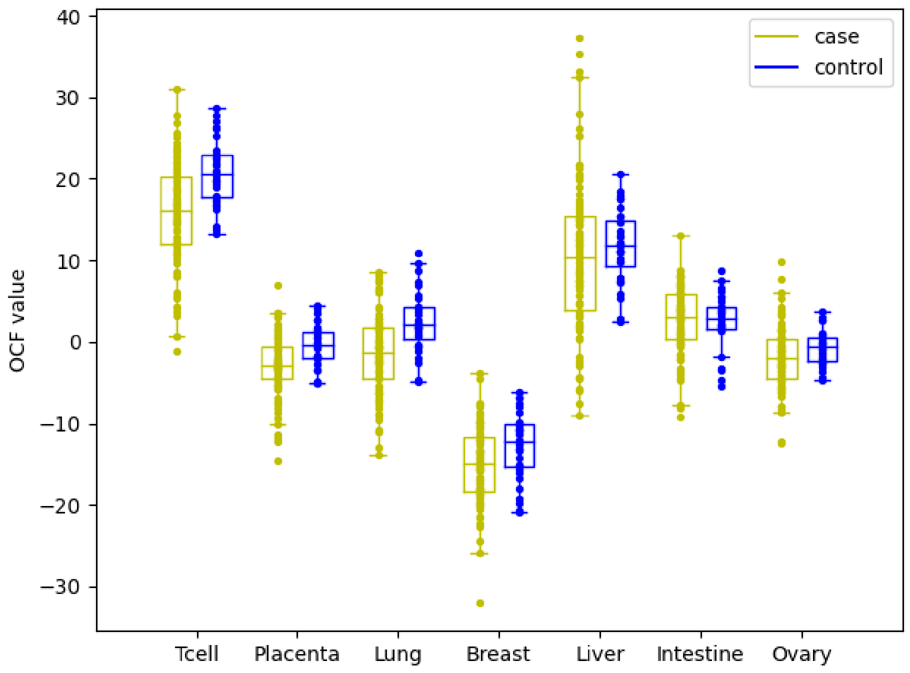

Figure 8: OCF analysis between HCC patients (case) and the healthy (control)

```
)

verbose = False

case_bedgz = glob.glob("path_to_data/HCC/*.bed.gz")
ctrl_bedgz = glob.glob("path_to_data/CTR/*.bed.gz")
```

Second, using the following commands to remove GC-bias and calculate enhanced fragmentation profiles between HCC patients (case) and the healthy (control).

```
# case
switchConfigure("cancer")
case_fragCounter = fpCounter(
    bedgzInput=case_bedgz, upstream=True, verbose=verbose, stepNum="case01", processtype=1
)
case_gcCounter = runCounter(
    filetype=0, binlen=5000000, upstream=True, verbose=verbose, stepNum="case02"
)
case_GCCorrect = GCCorrect(
    readupstream=case_fragCounter,
    gcupstream=case_gcCounter,
    readtype=2,
    corrkey="-",
    verbose=verbose,
    stepNum="case03",
)

# ctrl
```

```python
switchConfigure("normal")
ctrl_fragCounter = fpCounter(
    bedgzInput=ctrl_bedgz, upstream=True, verbose=verbose, stepNum="ctrl01", processtype=1
)
ctrl_gcCounter = runCounter(
    filetype=0, binlen=5000000, upstream=True, verbose=verbose, stepNum="ctrl02"
)
ctrl_GCCorrect = GCCorrect(
    readupstream=ctrl_fragCounter,
    gcupstream=ctrl_gcCounter,
    readtype=2,
    corrkey="-",
    verbose=verbose,
    stepNum="ctrl03",
)

switchConfigure("cancer")
res_fragprofplot = fragprofplot(
    caseupstream=case_GCCorrect,
    ctrlupstream=ctrl_GCCorrect,
    stepNum="FP",
)
```

The results below shows a disorder fragmentation profile of HCC patient compared with the healthy group.

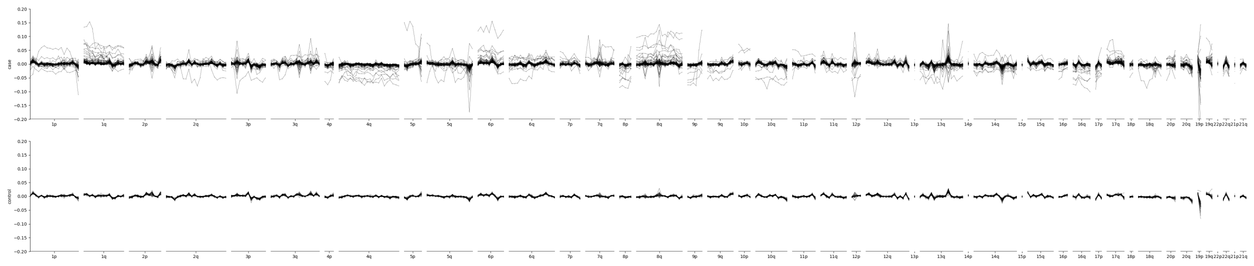

Figure 9: Fragmentation Profile Between HCC patients (case, top) and the healthy (control, bottom)

6 Others

cfDNApipe contains other useful functions such as germline and somatic mutation detection as well as virus detection. For more detailed information, please see cfDNApipe homepage.

7 Reference

Andrews,S. and others (2010) FastQC: A quality control tool for high throughput sequence data.

Cai,Z. *et al.* (2019) Comprehensive liquid profiling of circulating tumor dna and protein biomarkers in long-term follow-up patients with hepatocellular carcinoma. *Clinical Cancer Research*, **25**, 5284–5294.

Chan,K.A. *et al.* (2013) Noninvasive detection of cancer-associated genome-wide hypomethylation and copy number aberrations by plasma dna bisulfite sequencing. *Proceedings of the National Academy of Sciences*, **110**, 18761–18768.

Chen,X. *et al.* (2019) Low-pass whole-genome sequencing of circulating cell-free dna demonstrates dynamic changes in genomic copy number in a squamous lung cancer clinical cohort. *Clinical Cancer Research*, **25**, 2254–2263.

Feng,H. *et al.* (2019) Disease prediction by cell-free dna methylation. *Briefings in bioinformatics*, **20**, 585–597.

Huang,C.-C. *et al.* (2019) Bioinformatics analysis for circulating cell-free dna in cancer. *Cancers*, **11**, 805.

Jiang,P. *et al.* (2015) Lengthening and shortening of plasma dna in hepatocellular carcinoma patients. *Proceedings of the National Academy of Sciences*, **112**, E1317–E1325.

Kang,S. *et al.* (2017) CancerLocator: Non-invasive cancer diagnosis and tissue-of-origin prediction using methylation profiles of cell-free dna. *Genome biology*, **18**, 1–12.

Kim,D. *et al.* (2016) Centrifuge: Rapid and sensitive classification of metagenomic sequences. *Genome research*, **26**, 1721–1729.

Krueger,F. and Andrews,S.R. (2011) Bismark: A flexible aligner and methylation caller for bisulfite-seq applications. *bioinformatics*, **27**, 1571–1572.

Langmead,B. and Salzberg,S.L. (2012) Fast gapped-read alignment with bowtie 2. *Nature methods*, **9**, 357.

Li,H. *et al.* (2009) The sequence alignment/map format and samtools. *Bioinformatics*, **25**, 2078–2079.

Li,J. *et al.* (2017) Cell-free dna copy number variations in plasma from colorectal cancer patients. *Molecular oncology*, **11**, 1099–1111.

Li,W. *et al.* (2018) CancerDetector: Ultrasensitive and non-invasive cancer detection at the resolution of individual reads using cell-free dna methylation sequencing data. *Nucleic acids research*, **46**, e89–e89.

McKenna,A. *et al.* (2010) The genome analysis toolkit: A mapreduce framework for analyzing next-generation dna sequencing data. *Genome research*, **20**, 1297–1303.

Mouliere,F. *et al.* (2018) Enhanced detection of circulating tumor dna by fragment size analysis. *Science translational medicine*, **10**.

Okonechnikov,K. *et al.* (2016) Qualimap 2: Advanced multi-sample quality control for high-throughput sequencing data. *Bioinformatics*, **32**, 292–294.

Schubert,M. *et al.* (2016) AdapterRemoval v2: Rapid adapter trimming, identification, and read merging. *BMC research notes*, **9**, 1–7.

Snyder,M.W. *et al.* (2016) Cell-free dna comprises an in vivo nucleosome footprint that informs its tissues-of-origin. *Cell*, **164**, 57–68.

Sun,K. *et al.* (2019) Orientation-aware plasma cell-free dna fragmentation analysis in open chromatin regions informs tissue of origin. *Genome research*, **29**, 418–427.

Supernat,A. *et al.* (2018) Comparison of three variant callers for human whole genome sequencing. *Scientific reports*, **8**, 1–6.

Talevich,E. *et al.* (2016) CNVkit: Genome-wide copy number detection and visualization from targeted dna sequencing. *PLoS computational biology*, **12**, e1004873.

Zviran,A. *et al.* (2020) Genome-wide cell-free dna mutational integration enables ultra-sensitive cancer monitoring. *Nature Medicine*, 1–11.