Supplementary Material

2021-04-21

Contents

1 Introduction

cfDNApipe is an integrated pipeline for analyzing cell-free DNA WGBS/WGS data. It contains the essential steps for cfDNA pre-processing, quality control and feature extraction algorithms.

The whole pipeline is designed based on flow graph principle. Users can use the build-in pipelines for single/paired end WGS/WGBS data or build their own analysis pipeline from any intermediate data like BAM/BED files. The main functions are shown in Fig. S1.

Fig. S2 shows the whole functions in WGS and WGBS data processing.

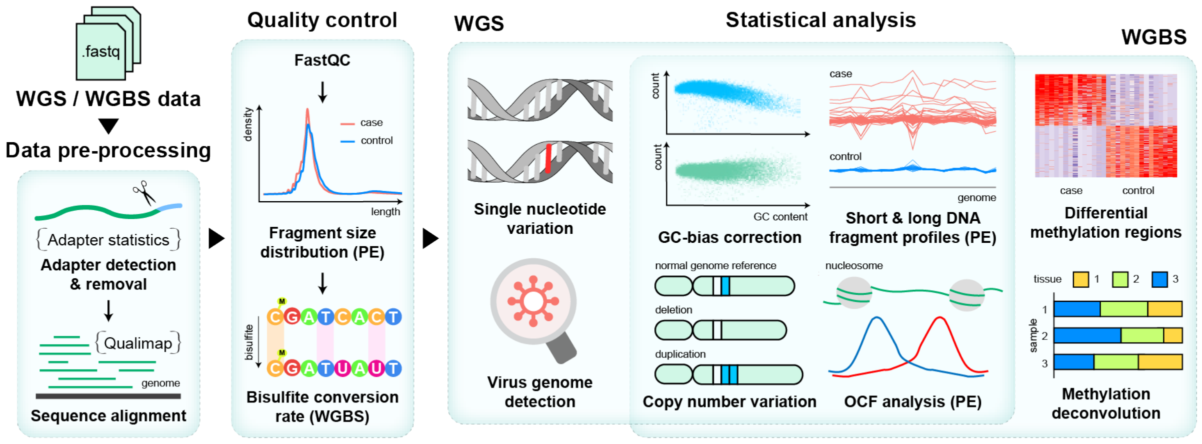

Fig. S1 cfDNApipe Functions

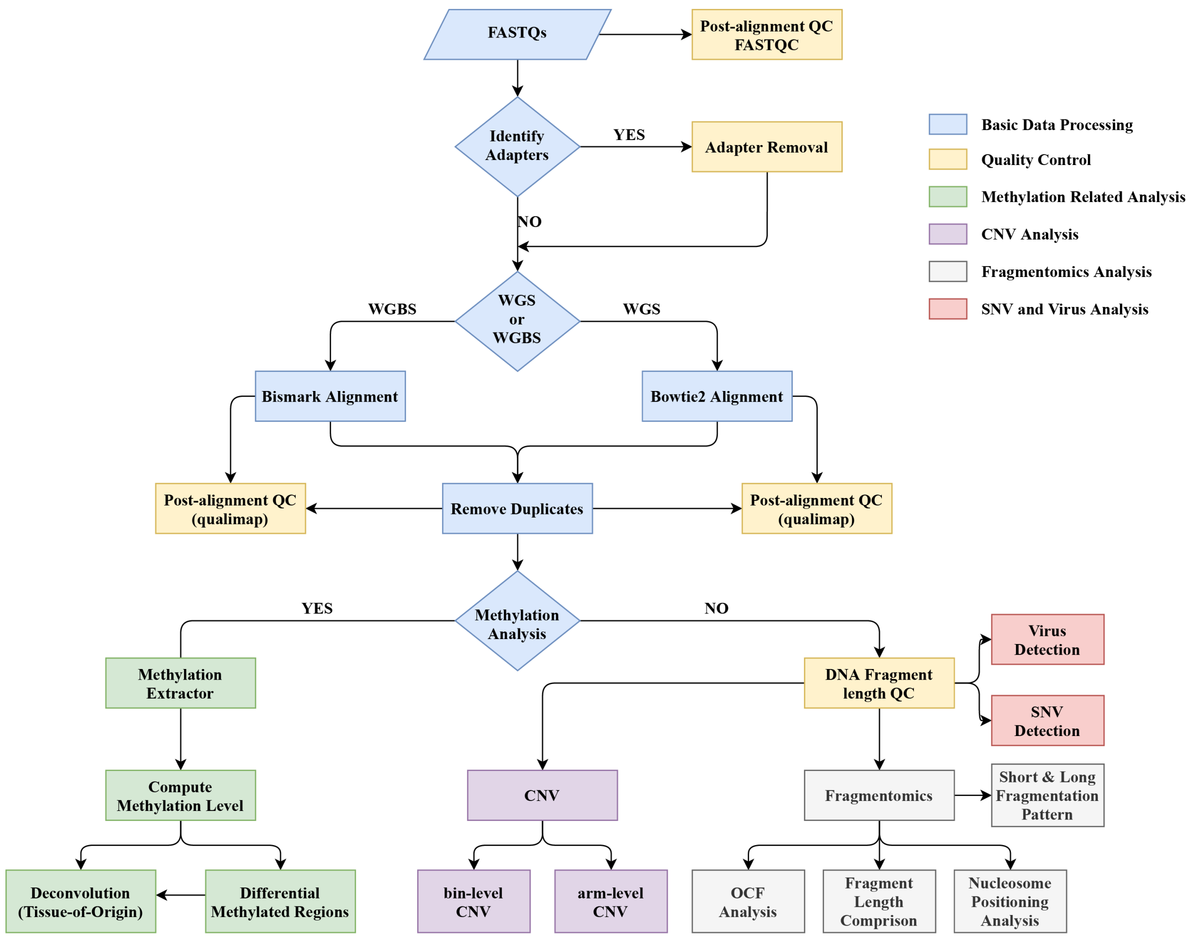

Fig. S2 Dataflow Overview

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://xwanglabthu.github.io/cfDNApipe/environment.yml
```

Then, please 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.

```
# clean before installation
conda clean -y --all

# install environment
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 WGS/WGBS single/paired end** data.

3 Functional Summary

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 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. 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; Guo *et al.*, 2017). 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 more detailed instruction is illustrated in cfDNApipe home page. 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,
)
```

The detailed explanation can be found in cfDNApipe home page and cfDNApipe documentation.

4.2 Case Control Analysis

Here, we show a case-control analysis procedure for paired end WGS data. A more detailed instruction is illustrated in cfDNApipe home page. 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,
)
```

The detailed explanation can be found in cfDNApipe home page and cfDNApipe documentation.

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 datasets are from paper "Lengthening and shortening of plasma DNA in hepatocellular carcinoma patients", "Noninvasive detection of cancer-associated genome-wide hypomethylation and copy number aberrations by plasma DNA bisulfite sequencing" and "Cell-free DNA Comprises an In Vivo Nucleosome Footprint that Informs Its Tissues-Of-Origin" with accession number EGAS00001001024, EGAS00001000566 and GSE71378 respectively. 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 5.1~5.6.

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 home page.

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 Fig. S3.

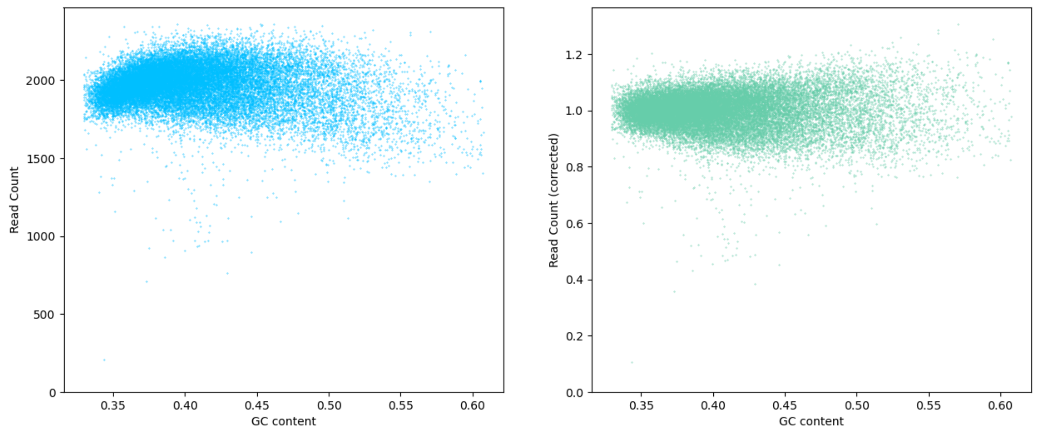

Fig. S3 GC content correction for sample H228

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

```
switchConfigure("cancer")
res_computeCNV = computeCNV(
    caseupstream=case_GCCorrect,
    ctrlupstream=ctrl_GCCorrect,
    stepNum="ARMCNV",
    verbose=verbose,
)
```

Users can find the final Z-score file and the results are shown in Fig. S4.

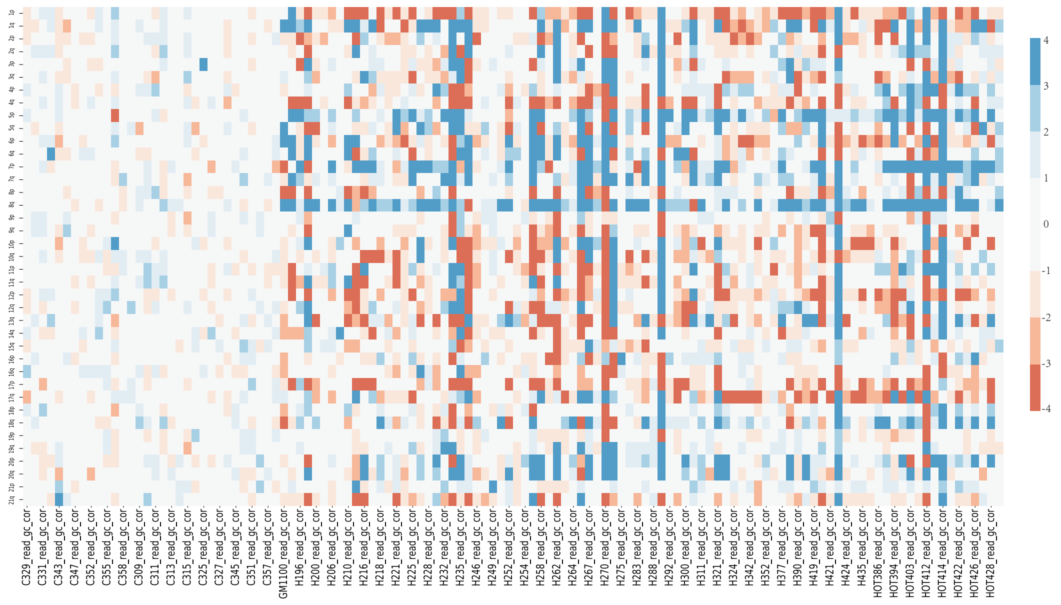

Fig. S4 CNV Z-score for all samples

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 5 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.

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). Besides, based on (Mouliere and Thierry, 2012) and (Jiang and Lo, 2016), necrosis results in much longer fragments, usually > 1000bp. Therfore, longer fragments may reveal the signal from necrosis. cfDNApipe takes this information as an important signature of plasma DNA and can generate distribution plot as well as related statistics.

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.

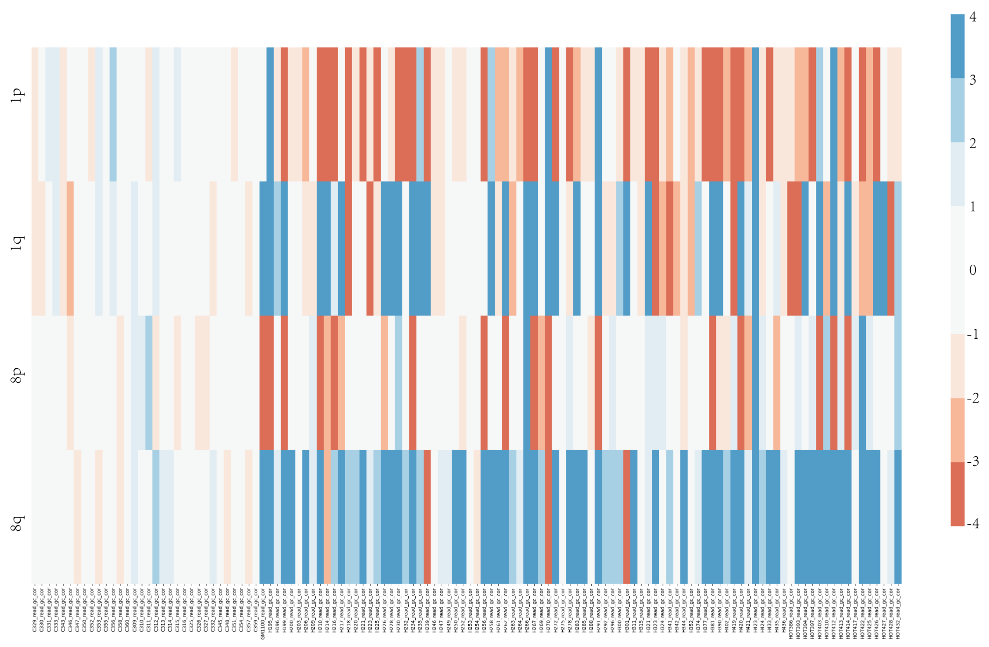

Fig. S5 CNV Z-score in chromatin 1 and 8 for all samples

```python
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.

```python
res_fraglenplot_comp = fraglenplot_comp(
    casebedInput=case_bed, ctrlbedInput=ctrl_bed,
    ratio1=150, ratio2=400, caseupstream=True,
    verbose=verbose)
```

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

```
# only shows 4 sample
                        Short(<150 bp) Rate     Long(>400 bp) Rate      Peak 1      Peak 2
GM1100_fraglen.pickle   0.1961189437656195      0.0018909480762622      165         330
```

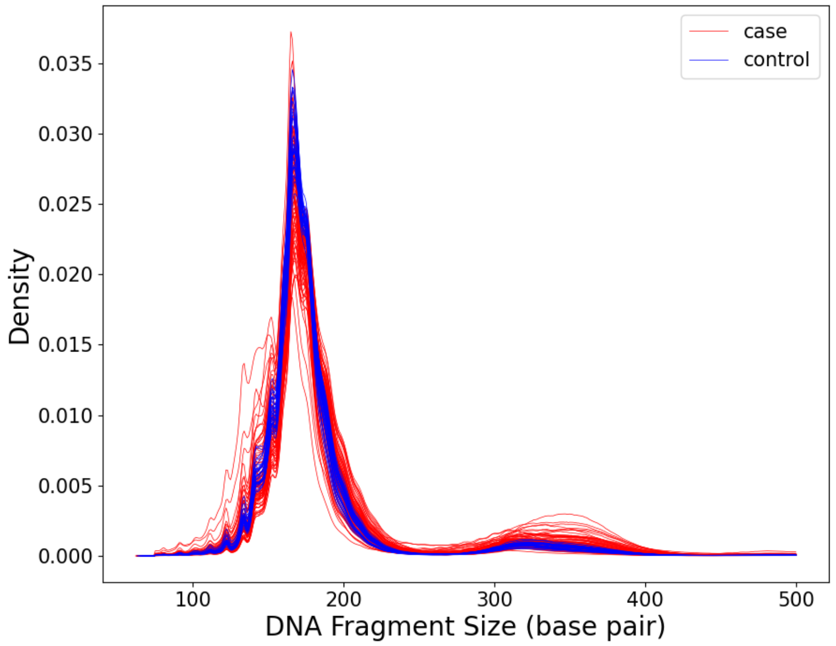

Fig. S6 cfDNA fragment size distribution between HCC patients (case) and the healthy (control)

```
H195_fraglen.pickle       0.1177497441211625      0.0048737613946009      167         334
C309_fraglen.pickle       0.1152524829833759      0.0034377031387753      166         332
C310_fraglen.pickle       0.1171784039400173      0.0033180126814695      166         332
```

If users want to analysis cfDNA from necrosis, please use parameters `"other_params={"-X": 2000}"` in bismark or bowtie2 and `"ratio2=1000"` for plotting fragment length distribution. This will report cfDNA longer than 1000bp which may come from necrosis.

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 and control samples as well as 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 Fig. S7 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).

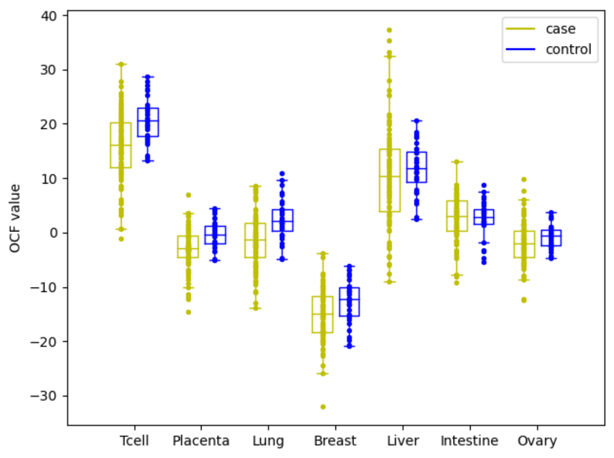

Fig. S7 OCF analysis between HCC patients (case) and the healthy (control)

5.4 Enhanced Fragmentation Analysis Reveals Disorder of Cancer Genome

The former work has announced disorder in arm-level CNV, and (Cristiano *et al.*, 2019) 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 function “bam2bed” in cfDNApipe).

```
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_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
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.

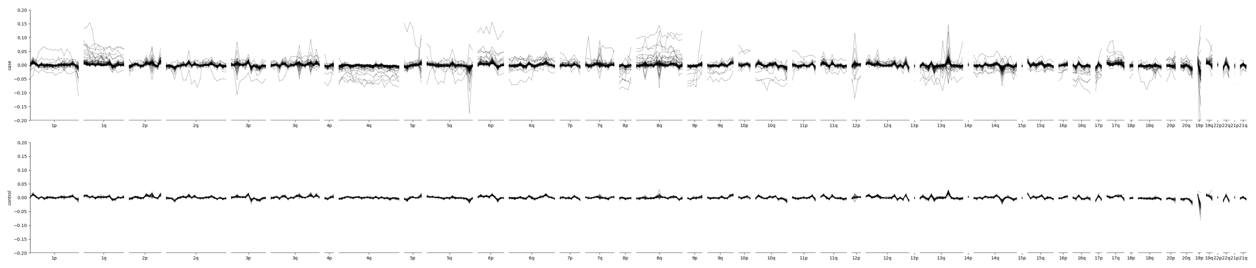

Fig. S8 Fragmentation Profile Between HCC patients (case, top) and the healthy (control, bottom)

5.5 Inferring Nucleosome Positioning

The previous work (Snyder *et al.*, 2016) reports that the intensity of the FFT signal of **nucleosome positioning** around gene body is correlated with gene expression at specific frequency ranges, with a maximum at 177–180 bp for positive correlation and a minimum at ~199 bp for negative correlation. Therefore, **nucleosome positioning** is an importtant feature for identifing cfDNA origin. Researchers proposed a statistics named windowed protection score (WPS) to reveal the **nucleosome positioning** in cfDNA HTS data. cfDNApipe provides function to calculate WPS in any genome region. The following steps illustrate how to compute WPS in arbitrary genome region.

First, select the genome regions.

The previous work (Snyder *et al.*, 2016) selected the first 10 kb of gene bodies. However, TF binding regions also show different signal compared with flank region (Ulz *et al.*, 2019). TF binding regions is changed in different cell type and different cell state and gene annotation is updating continuously. Therefore, cfDNApipe do not provide a pre-set region file instead of telling the users how to get the genome regions from gencode annotation files.

In this part, we will illustrate how to get gene body from gencode annotation files.

Users can download gencode annotation files from gencode database, the commonly used files are gencode.v19.annotation.gtf.gz for hg19 and gencode.v37.annotation.gtf.gz for hg38. Here, we use hg19 as an example.

```
library(rtracklayer)
library(dplyr)
```

```
anno_raw <- import("gencode.v19.annotation.gtf.gz")

# get all genes
anno_raw <- anno_raw[which(anno_raw$type == "gene"), ]

anno <- data.frame(gene_id = anno_raw$gene_id, chr = seqnames(anno_raw),
                   start = start(anno_raw), end = end(anno_raw),
                   strand = strand(anno_raw))

# get genome region downstream 10000bp from TSS
for (i in nrow(anno)) {
    if (anno$strand[i] == "+") {
        anno$start = anno$start - 1
        anno$end = anno$start + 10000
    } else {
        anno$start =anno$end + 1 - 10000
        anno$end = anno$end + 1
    }
}

# remove invalid
anno <- anno[which(anno$chr %in% paste0("chr", c(1:22, "X"))), ]
anno <- anno[which(anno$start > 0), ]

write.table(x = anno, file = "transcriptAnno-v19.tsv", sep = "\t",
            col.names = FALSE, row.names = FALSE, quote = FALSE)
```

Note: Be aware about the feature number in your annotation file. From the above Rscript, 57820 genes remained. Users should filter the features such as protein coding genes. Linux system limits file number in a folder, therefore, shrink the rows in annotation file is necessary.

From the above code, users can get a tsv file named "transcriptAnno-v19.tsv" which saves the genome region downstream 10000bp from gene TSS. Users can get customized regions like TF binding regions. For a better illustration, we added one region in the end of "transcriptAnno-v19.tsv". This region is from alpha-satellite region in chr12 which shows a strongly positioned nucleosomes signal reported by previous work (Snyder *et al.*, 2016).

```
alpha_satellite chr12   34443000    34446000    +
```

Second, compute WPS for these regions. Low coverage sample shows weak WPS signal, therefore, we use only one sample IC17 (Hepatocellular Carcinoma, 42.08X coverage) from the previous work (Snyder *et al.*, 2016) for better illustration. Users can perform the following analysis by using the pre-set pipeline results or aligned bam files from customized script.

```
from cfDNApipe import *

pipeConfigure(
    threads=60,
    genome="hg19",
    refdir="/home/zhangwei/Genome/hg19_bowtie2",
    outdir="/opt/tsinghua/zhangwei/nucleosome_positioning",
    data="WGS",
    type="paired",
    JavaMem="8G",
    build=True,
)

# be aware all the samples should in a list
res1 = bam2bed(bamInput=["SRR2130016-rmdup.bam"], upstream=True)

res2 = runWPS(upstream=res1, tsvInput="transcriptAnno-v19.tsv")
```

We can find the output files saved in the folder "intermediate_result/step_02_runWPS/SRR2130016.bed" and plot the WPS in Fig. S9.

```
data <- read.table("SRR2130016.bed_alpha_satellite.tsv.gz")
WPS <- data$V5

x <- seq(34443000, 34446000)

plot(x = x, y = WPS, type = "l",
     xlab = "genome coordinate", ylab = "WPS", lwd = "1")
```

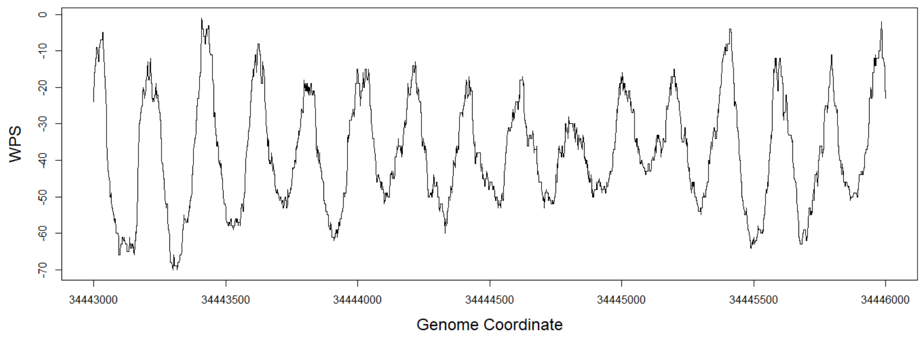

Fig. S9 Nucleosome positioning of alpha satellite region in chr12

For the following analysis like FFT and correlation with gene expression, these analysis is highly user specific and can be easily performed from the results of cfDNApipe.

5.6 Inferring Tissue-Of-Origin based on Deconvolution

Inferring tissue-of-origin from cfDNA data is of great potential for further clinical applications. For example, as the previous study (Fiala and Diamandis, 2018) mentioned, cancerous cells release more DNA into plasma. Therefore, inferring tissue-of-origin can be used for early cancer detection. In cfDNApipe, tissue-of-origin analysis can be achieved through OCF analysis or deconvolution (Zhang *et al.*, 2021).

Sun, et al. proposed a deconvolution strategy for inferring tissue proportion from WGBS data (Sun *et al.*, 2015). The paper from Moss, et al. also exhibits the feasibility of using methylation-based deconvolution (Moss *et al.*, 2018). We collected WGBS data of plasma cfDNA from 32 healthy people and 24 HCC patients (A and B stage) applied a novel deconvolution algorithm on this dataset to infer the fraction of liver-derived cfDNA in each sample by using an external methylation reference from a former study.

The basic analysis can be achieved by using the pre-set pipeline. Here, we start with the extracted methylation files.

First, set global parameters and compute methylation level for regions from previous study (Sun *et al.*, 2015). The deconvolution can be accomplished using only one command in python.

```python
import glob
from cfDNApipe import *

pipeConfigure2(
    threads=100,
    genome="hg19",
    refdir=r"path_to_reference/hg19_bismark",
    outdir=r"path_to_output/WGBS",
    data="WGBS",
    type="single",
    case="HCC",
    ctrl="Healthy",
    JavaMem="10G",
    build=True,
)

hcc = glob.glob("/WGBS/HCC/intermediate_result/step_06_compress_methyl/*.gz")
ctr = glob.glob("/WGBS/Healthy/intermediate_result/step_06_compress_methyl/*.gz")

verbose = False

switchConfigure("HCC")
hcc1 = calculate_methyl(tbxInput=hcc,
        bedInput="plasmaMarkers_hg19.bed",
        upstream=True, verbose=verbose)
hcc2 = deconvolution(upstream=hcc1)

switchConfigure("Healthy")
ctr1 = calculate_methyl(tbxInput=ctr,
        bedInput="plasmaMarkers_hg19.bed",
        upstream=True, verbose=verbose)
ctr2 = deconvolution(upstream=ctr1)
```

Note: The default bedInput for class calculate_methyl is CpG island regions. Therefore, we should change this file to match the external reference provided in cfDNApipe. Of course, user defined files can be passed to deconvolution easily. For detailed parameter explanation, please see here.

Next, plot liver proportion between HCC patients and healthy people in R.

```r
library(ggplot2)

HCC <- read.table(
    file = "path_to_output/WGBS/Healthy/intermediate_result/step_02_deconvolution/result.txt",
    header = TRUE, sep = "\t", row.names = 1
    )
CTR <- read.table(
    file = "path_to_output/WGBS/HCC/intermediate_result/step_02_deconvolution/result.txt",
    header = TRUE, sep = "\t", row.names = 1
    )

HCC.liver <- unlist(HCC["Liver", ])
CTR.liver <- unlist(CTR["Liver", ])
p <- wilcox.test(x = CTR.liver, y = HCC.liver, alternative = "less")$p.value

data <- data.frame(Class = c(rep("Healthy", 32), rep("HCC", 24)),
                    Liver_Prop = c(CTR.liver, HCC.liver))

ggplot(data, aes(x=Class, y=Liver_Prop, fill=Class)) +
    geom_boxplot() +
    labs(x = paste("T-TEST:", p, sep = " ")) +
    labs(y = "Liver Proportion") +
    theme(axis.text.x = element_text(size = 16)) +
    theme(axis.title = element_text(size = 16)) +
    theme(legend.text = element_text(size = 16)) +
    theme(title = element_text(size = 16)) +
    theme(axis.text = element_text(size = 16)) +
    theme(panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(),
          panel.background = element_blank(),
          axis.line = element_line(colour = "black"),
          panel.border = element_rect(colour = "black", fill=NA, size=2))
```

The result is shown in Fig. S10. cfDNApie estimated a higher liver-derived fractions in cfDNA from HCC patients with an average value of 15.2%, while those from healthy people with an average value of 7.6%. The liver-derived fractions in cfDNA between HCC patients and healthy people shows a sig-nificant statistical difference (Mann–Whitney *U* test, *p*-value = 9.36×10^{-5}), which is consistent with the former discovery (Fiala and Diamandis, 2018). The results suggest that cfDNApipe has the potential to infer tissue-of-origin and detect changes of proportions from different tissues.

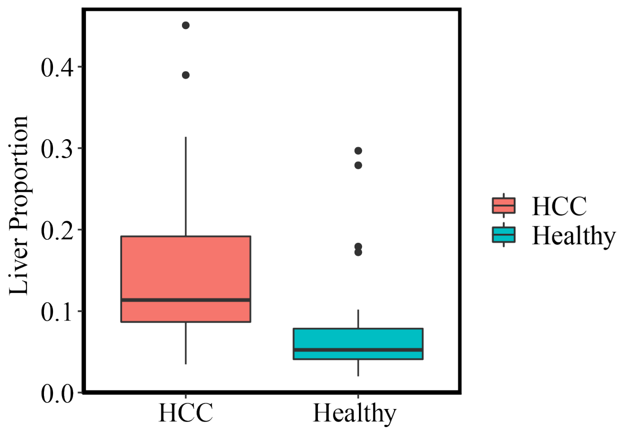

Fig. S10 Liver Proportion from HCC patients and healthy people WGBS data

6 Others

cfDNApipe contains other useful functions such as germline and somatic mutation detection (McKenna *et al.*, 2010) as well as virus detection (Kim *et al.*, 2016). For more detailed information, please see cfDNApipe home page.

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.

Cristiano,S. *et al.* (2019) Genome-wide cell-free dna fragmentation in patients with cancer. *Nature*, **570**, 385–389.

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

Fiala,C. and Diamandis,E.P. (2018) Utility of circulating tumor dna in cancer diagnostics with emphasis on early detection. *BMC medicine*, **16**, 1–10.

Guo,S. *et al.* (2017) Identification of methylation haplotype blocks aids in deconvolution of heterogeneous tissue samples and tumor tissue-of-origin mapping from plasma dna. *Nature genetics*, **49**, 635–642.

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.

Jiang,P. and Lo,Y.D. (2016) The long and short of circulating cell-free dna and the ins and outs of molecular diagnostics. *Trends in Genetics*, **32**, 360–371.

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.

Moss,J. *et al.* (2018) Comprehensive human cell-type methylation atlas reveals origins of circulating cell-free dna in health and disease. *Nature communications*, **9**, 1–12.

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

Mouliere,F. and Thierry,A.R. (2012) The importance of examining the proportion of circulating dna originating from tumor, microenvironment and normal cells in colorectal cancer patients. *Expert opinion on biological therapy*, **12**, S209–S215.

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.* (2015) Plasma dna tissue mapping by genome-wide methylation sequencing for noninvasive prenatal, cancer, and transplantation assessments. *Proceedings of the National Academy of Sciences*, **112**, E5503–E5512.

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.

Ulz,P. *et al.* (2019) Inference of transcription factor binding from cell-free dna enables tumor subtype prediction and early detection. *Nature communications*, **10**, 1–11.

Zhang,W. *et al.* (2021) ARIC: Accurate and robust inference of cell type proportions from bulk gene expression or dna methylation data. *bioRxiv*.

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