An Introduction to *SimFFPE*

Lanying Wei

Modified: Nov 11, 2020. Compiled: May 19, 2021

Contents

1 Introduction

The NGS (Next-Generation Sequencing) reads from FFPE (Formalin-Fixed Paraffin-Embedded) samples contain numerous artificial chimeric reads, which can lead to false positive structural variant calls. These ACRs are derived from the combination of two single-stranded DNA (ss-DNA) fragments with short reverse complementary regions (SRCR). The combined ss-DNA may come from adjacent or distant genomic regions. The *SimFFPE* package simulates these artificial reads as well as normal reads for FFPE samples. The simulation can cover whole genome, or several chromosomes, or large regions, or whole exome, or targeted regions. It also supports enzymatic / random fragmentation and paired-end / single-end sequencing simulations.Fine-tuning can be achieved by adjusting the parameters, and multi-threading is surported.

2 Input

The essential inputs for the simulation include a `FASTA` file of the reference genome, a `Phred score profile` matrix to simulate Phred scores based on the position on the reads, and a DataFrame or GenomicRanges object representing the target regions (optional). The `Phred score profile` can be estimated from existing `BAM` files using `calcPhredScoreProfile` function (two available examples for `Phred score profile` are stored in the 'extdata' directory of *SimFFPE* package).

```
> library(SimFFPE)
> bamFilePath <- system.file("extdata", "example.bam", package = "SimFFPE")
> regionPath <- system.file("extdata", "regionsBam.txt", package = "SimFFPE")
> regions <- read.table(regionPath)
```

```
> PhredScoreProfile <- calcPhredScoreProfile(bamFilePath, targetRegions = regions)
> ## Example Phred score profile with 100 read length
> PhredScoreProfilePath <- system.file("extdata", "PhredScoreProfile1.txt",
+                                      package = "SimFFPE")
> PhredScoreProfile <- as.matrix(read.table(PhredScoreProfilePath, skip = 1))
> colnames(PhredScoreProfile)  <-
+     strsplit(readLines(PhredScoreProfilePath)[1], "\t")[[1]]
> #
> ## Example Phred score profile with 150 read length
>
> PhredScoreProfilePath2 <- system.file("extdata", "PhredScoreProfile2.txt",
+                                      package = "SimFFPE")
> PhredScoreProfile2 <- as.matrix(read.table(PhredScoreProfilePath2, skip = 1))
> colnames(PhredScoreProfile2)  <-
+     strsplit(readLines(PhredScoreProfilePath2)[1], "\t")[[1]]
>
```

The `FASTA` file of reference genome can be read in as *DNAStringSet* with `readDNAStringSet` function from *Biostrings* package. The reference genome example file consists of small regions of 24 chromosomes from human hg19 reference genome.

```
> referencePath <- system.file("extdata", "example.fasta", package = "SimFFPE")
> reference <- readDNAStringSet(referencePath)
> reference

DNAStringSet object of length 24:
     width seq                          names
 [1] 30000 AGACTAACATGGA...TCCTTTCTTTCC 1 dna 20000001:20...
 [2] 30000 ACATTTCCATTTG...GTAGGCGGGGCA 2 dna 20000001:20...
 [3] 30000 TGTTTACACATTT...TGCCCAAAACTT 3 dna 20000001:20...
 [4] 30000 GTTTAACGATCTA...TGTCGTCTGCCT 4 dna 20000001:20...
 [5] 30000 CCACTTATCTTGT...AGGTGTTTGCTA 5 dna 20000001:20...
 ...   ... ...
[20] 30000 TCAGTTTGGGAGG...AATCTCCTTTAG 20 dna 20000001:2...
[21] 30000 CCCTTCTCCTATC...TAAAATACTCAA 21 dna 20000001:2...
[22] 30000 TGGGAAGGTGGGG...GAAATATTTGTT 22 dna 20000001:2...
[23] 30000 AGAAATGATGGCT...AGGCTCTGAAGA X dna 20000001:20...
[24] 30000 ATGGTATTTGGGA...CAAAAAGGAATG Y dna 20000001:20...

>
```

To simulate reads of certain regions, a DataFrame or GenomicRanges object representing the target regions is required (not required when simulating reads on the whole genome / several chromosomes / large regions). The DataFrame representing the target regions should have three columns, which indicate chromosomes, start positions and end positions respectively (one-based coordinate).

```
> regionPath <- system.file("extdata", "regionsSim.txt", package = "SimFFPE")
> targetRegions <- read.table(regionPath)
```

3 Simulation

The simulation includes three steps: 1) Simulate artificial chimeric reads derived from the combination of two ss-DNA segments from adjacent regions on the chromosome. 2) Simulate artificial chimeric reads derived from the combination of two ss-DNA from distant regions (distant regions on the same chromosome, or any regions on different chromosomes). 3) Simulate reads derived from normal sequences. You can also skip any of these steps in the simulation. There are two functions which can be used for the simulation: `readSimFFPE` and `targetReadSimFFPE`.

3.1 Read simulation functions

To simulate reads on whole genome, or several chromosomes, or large regions, please use the `readSimFFPE` function:

```
> ## Simulate reads of the first three sequences of the reference genome
>
> sourceSeq <- reference[1:3]
> outFile1 <- paste0(tempdir(), "/sim1")
> readSimFFPE(sourceSeq, referencePath, PhredScoreProfile2, outFile1,
+             overWrite = TRUE, coverage = 80, readLen = 150,
+             enzymeCut = TRUE, threads = 2)
> #
> ## Simulate reads of defined regions on the first two sequences of
> ## the reference genome
>
> sourceSeq2 <- DNAStringSet(lapply(reference[1:2], function(x) x[1:10000]))
> outFile2 <- paste0(tempdir(), "/sim2")
> readSimFFPE(sourceSeq2, referencePath, PhredScoreProfile2, outFile2,
+             overWrite = TRUE, coverage = 80, readLen = 150, enzymeCut = TRUE)
>
```

To simulate reads on whole exome or targeted regions please use the `targetReadSimFFPE` function:

```
> outFile3 <- paste0(tempdir(), "/sim3")
> targetReadSimFFPE(referencePath, PhredScoreProfile, targetRegions, outFile3,
+                   coverage = 120, readLen = 100, meanInsertLen = 180,
+                   sdInsertLen = 50, enzymeCut = FALSE)
>
```

Additional information can be found on the help pages for the `readSimFFPE` function and the `targetReadSimFFPE` function.

3.2 Fine-tuning of the simulation

Fine-tuning of the simulation is achievable by the adjustments of some parameters of the function `readSimFFPE` and `targetReadSimFFPE`. You can simulate reads in smaller regions during fine-tuning to save the runtime. To illustrate the impact of some of these parameters, screenshots from IGV tools are used (see Figure 1, 2 and 3). These parameters include:

1) enzymeCut: Simulate enzymatic fragmentation. With this fragmentation method, chimeric read pairs with improper pair orientations might be mapped to exactly the same location on the reference genome (see Figure 1).

2) chimericProp: Proportion of artificial chimeric fragments. The higher the value, the greater the proportion of improper paired reads as shown in Figure 1 and 2, the smaller the proportion of proper paired reads as shown in Figure 3, and the larger the proportion of proper paired reads with soft-clips as shown in Figure 3.

3) sameChrProp: Proportion of artifact chimeric fragments that are derived from the combination of two ss-DNA coming from the same chromosome. The higher the value, the greater the proportion of reads with improper pair orientations as shown in Figure 1, and the smaller the proportion of reads with their mates mapped to different chromosomes as shown in Figure 2.

4) adjChimProp: Proportion of adjacent ss-DNA combinations among same chromosomal ss-DNA combinations. sameChrProp * adjChimProp determine the proportion of simulated adjcent ss-DNA combinations.

5) sameStrandProb: Proportion of same-strand ss-DNA combinations among adjacent ss-DNA combinations. The higher the value, the greater the proportion of reads with RR and LL orientations in all reads with improper pair orientations as shown in Figure 1.

6) spikeWidth: The width of chimeric read spike used in the simulation of distant ss-DNA combinations. As shown in Figure 3, some regions are enriched in reads with paired reads mapped to other chromosomes, and some others are scarce. The lengths of these regions are of similar scale, and the parameter "spikeWidth" is used to simulate this length.

7) highNoiseRate and highNoiseProb: The noise rate for each base in noisy reads and the proportion of these noisy reads. These very noisy reads as well as less noisy reads are shown in Figure 3.

Figure 1: Simulated reads with improper pair orientations
These read pairs are mapped in RR, LL, or RL orientations. Black-framed reads are paired reads that are mapped to the same position (enzymatic fragmentation).

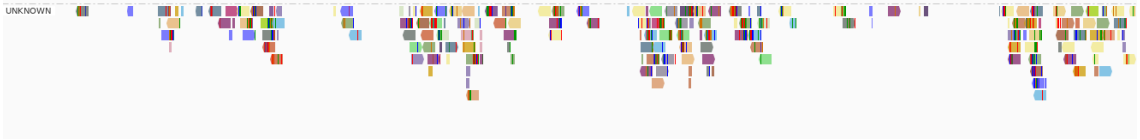

Figure 2: Simulated reads with mate reads mapped to different chromosomes
The different read colors indicate the different chromosomes that their mate reads mapped to.

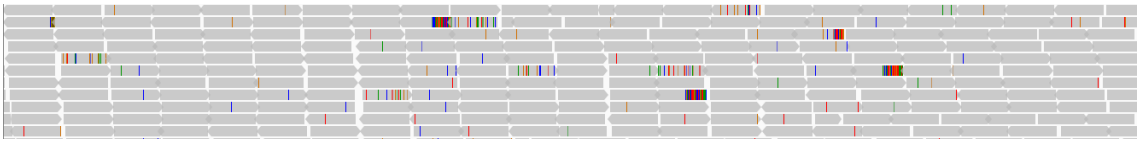

Figure 3: Simulated reads mapped in proper pair
Mismatches and soft-clips are shown as colored vertical lines in reads.

A Session info

```
> packageDescription("SimFFPE")

Package: SimFFPE
Type: Package
Title: NGS Read Simulator for FFPE Tissue
Version: 1.5.0
Authors@R: person("Lanying", "Wei",
      email="lanying.wei@uni-muenster.de", role =
      c("aut", "cre"), comment = c(ORCID =
      "0000-0002-4281-8017"))
Description: The NGS (Next-Generation Sequencing)
      reads from FFPE (Formalin-Fixed
      Paraffin-Embedded) samples contain numerous
      artifact chimeric reads (ACRS), which can lead
      to false positive structural variant calls.
      These ACRs are derived from the combination of
      two single-stranded DNA (ss-DNA) fragments with
      short reverse complementary regions (SRCRs).
      This package simulates these artifact chimeric
      reads as well as normal reads for FFPE samples
      on the whole genome / several chromosomes /
      large regions.
License: LGPL-3
Encoding: UTF-8
Depends: Biostrings
Imports: dplyr, foreach, doParallel, truncnorm,
      GenomicRanges, IRanges, Rsamtools, parallel,
      graphics, stats, utils, methods
Suggests: BiocStyle
biocViews: Sequencing, Alignment, MultipleComparison,
      SequenceMatching, DataImport
git_url:
      https://git.bioconductor.org/packages/SimFFPE
git_branch: master
git_last_commit: c1d0d02
git_last_commit_date: 2021-05-19
Date/Publication: 2021-05-19
Author: Lanying Wei [aut, cre]
      (<https://orcid.org/0000-0002-4281-8017>)
Maintainer: Lanying Wei <lanying.wei@uni-muenster.de>
Built: R 4.1.0; ; 2021-05-19 22:25:41 UTC; unix

-- File: /tmp/Rtmp5MGxcC/Rinst2e040b16f8b816/SimFFPE/Meta/package.rds

> sessionInfo()

R version 4.1.0 beta (2021-05-03 r80259)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 20.04.2 LTS
```

```
Matrix products: default
BLAS:   /home/biocbuild/bbs-3.14-bioc/R/lib/libRblas.so
LAPACK: /home/biocbuild/bbs-3.14-bioc/R/lib/libRlapack.so

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_GB              LC_COLLATE=C
 [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] stats4    parallel  stats     graphics  grDevices
[6] utils     datasets  methods   base

other attached packages:
[1] SimFFPE_1.5.0       Biostrings_2.61.0
[3] GenomeInfoDb_1.29.0 XVector_0.33.0
[5] IRanges_2.27.0      S4Vectors_0.31.0
[7] BiocGenerics_0.39.0

loaded via a namespace (and not attached):
 [1] pillar_1.6.1           compiler_4.1.0
 [3] BiocManager_1.30.15    bitops_1.0-7
 [5] iterators_1.0.13       tools_4.1.0
 [7] zlibbioc_1.39.0        digest_0.6.27
 [9] tibble_3.1.2           evaluate_0.14
[11] lifecycle_1.0.0        pkgconfig_2.0.3
[13] rlang_0.4.11           foreach_1.5.1
[15] DBI_1.1.1              rstudioapi_0.13
[17] yaml_2.2.1             xfun_0.23
[19] GenomeInfoDbData_1.2.6 dplyr_1.0.6
[21] knitr_1.33             generics_0.1.0
[23] vctrs_0.3.8            tidyselect_1.1.1
[25] glue_1.4.2             R6_2.5.0
[27] fansi_0.4.2            BiocParallel_1.27.0
[29] rmarkdown_2.8          purrr_0.3.4
[31] magrittr_2.0.1         ellipsis_0.3.2
[33] Rsamtools_2.9.0        codetools_0.2-18
[35] htmltools_0.5.1.1      GenomicRanges_1.45.0
[37] assertthat_0.2.1       BiocStyle_2.21.0
[39] utf8_1.2.1             RCurl_1.98-1.3
[41] doParallel_1.0.16      truncnorm_1.0-8
[43] crayon_1.4.1
```