Using the SICtools Package

Xiaobin Xing

March 15, 2015

Contents

1 Introduction to SICtools

High-through sequencing has become fundamental for deciphering sequence variants between two paired-samples in different conditions, which has been vastly used in detecting somatic variants between tumor and matched normal samples in current research of oncogenesis.The *SICtools package* is designed to find SNV/Indel differences between two bam files with near relationship in a way of pairwise comparison thourgh parsing the allele frequency of genotypes (single nucleotide and short indel) at each base position across the genome region of interest. The difference is inferred by two complementary measurements, fisher exact test *p.value* and euclidean distance *d.value*. For SNV comparison, the internal input is the base count (A,T,G,C) in a given position, parsed from pileup output from the two bam files; for indel comparison, reads for different indel alleles that span no less than 2bp on both sides of extended indel region (e.g. homopolymer region) are counted as internal input.The candidate variants with relatively lower *p.value* and higher *d.value* can thus be easily identified from the output of SICtools.

2 Getting started with SICtools

Getting started with SICtools for inspection of SNV/Indel difference between two bam files is quite easy. Two critical functions (`snpDiff` and `indelDiff`) will be available in R session with loading the package.

```
library(SICtools)
```

```
## Warning: replacing previous import 'plyr::count' by 'matrixStats::count' when
## loading 'SICtools'
```

3 Function snpDiff()

3.1 Input

The essential capability provided by `SICtools` is its input. The two bam files to be compared should be aligned by same aligner of the same version (important!) and the same reference genome. In theory, the two bam files should be in near relationship, which means that the SNV/Indel differences are not expected too many. The input coordinate for region of interest should be the same format as the reference genome. The argument list of function `snpDiff` is below,

snpDiff(bam1, bam2, refFsa, regChr, regStart, regEnd, minBaseQuality = 13, minMapQuality = 0, nCores = 1, pValueCutOff = 0.05, baseDistCutOff = 0.1, verbose = TRUE)

Except the three main paramters (`bam1`,`bam2` and `refFsa`), the region coordinate arguments (`regChr`,`regStart` and `regEnd`) are necessary to restrict the genome region of interest, since the comparison would be time consuming if the chromosome units of the species are very long, for example, human. Meanwhile, these coordinate arguments make the parallel calculation possible. For instance, the long chromosome could be chunked into pieces and compared in parallel, and the final output would be combined together. Even in the genome region of interest, a parameter `nCores` is to set the threads for parallel calculation, which will greatly short the time in the case of long region input.

In order to control the input quality of reads, two quality filter arguments (`minBaseQuality` and `minMapQuality`) are provided. The `minBaseQuality` score is stranded Phred score of Sanger format for each base. The 'minMapQuality' score is based on the aligner used, which means differnt threshods would be adopted to control the mapping quality of the whole reads for different aligners.

Two output control parameters (`pValueCutOff` and `baseDistCutOff`) are used to filter the output. Since most of genomic positions to compare are the same in theory, `p.value` of exact fisher test would be `1` and `d.value` of Euclidean distance would be `0` in thousands, even millons of positions, which is not expected as output, and will be excluded from the final output. The default `pValueCutOff = 0.05, baseDistCutOff = 0.1` is proper for comparison between germline samples. Lower `baseDistCutOff` and higher `pValueCutOff` is probably needed for somatic samples.

If verbose = TRUE, the progress information of genomic positions will be showed on screen

3.2 Output

The output of SNV/Indel comparisons is a `data.frame`. It will report the base count/read count for each allele, p.value (from fisher exact test) and d.value (from euclidean distance) filtered by pre-defined cutoff of p.value and d.value. If nothing difference, NULL will be returned.

3.3 Example

The example will detect SNV differences between two bam files in the region "chr04:962501-1026983". Setting "pValueCutOff=1,baseDistCutOff=0" will detect tiny differences, while the exact same genotype positions will be excluded from output by default.

```
bam1 <- system.file(package='SICtools','extdata','example1.bam')
bam2 <- system.file(package='SICtools','extdata','example2.bam')
refFsa <- system.file(package='SICtools','extdata','example.ref.fasta')

snpDiffDf <- snpDiff(bam1,bam2,refFsa,'chr04',962501,1026983,pValueCutOff=1,baseDistCutOff=0)
```

```
## Warning in setup_parallel(): No parallel backend registered

## [1] "chr04 962501 1026983"

## Warning: 'applyPileups' is deprecated.
## Use 'pileup' instead.
```

```
## See help("Deprecated")
```

```
snpDiffDf
```

```
##      chr     pos ref A1 C1 G1 T1 N1 A2 C2 G2 T2 N2      p.value    d.value
## 1  chr04  962623   G  0  0 34  2  0  0  0 47  0  0 1.851308e-01 0.07856742
## 2  chr04  962801   G  0  0 47  0  0  0  0  0 45  0 2.487221e-27 1.41421356
## 3  chr04  962865   C  2 29  0  0  0  0 49  0  0  0 1.471519e-01 0.09123958
## 4  chr04  962984   G  0  0 28  0  0  0  1 30  1  0 1.000000e+00 0.07654655
## 5  chr04  962998   T  0  0  1 14  0  0  0  0 25  0 3.750000e-01 0.09428090
## 6  chr04  963005   A 16  0  0  0  0 19  1  0  0  0 1.000000e+00 0.07071068
## 7  chr04 1026413   T  1  0  0 15  0  0  0  0 21  0 4.324324e-01 0.08838835
## 8  chr04 1026421   C  0 17  1  0  0  0 22  0  0  0 4.500000e-01 0.07856742
## 9  chr04 1026533   T  0  0  0 22  0  0  0  1 18  0 4.634146e-01 0.07443229
## 10 chr04 1026599   A 16  0  0  0  0 19  0  1  0  0 1.000000e+00 0.07071068
## 11 chr04 1026603   T  0  0  0 17  0  0  0  1 18  0 1.000000e+00 0.07443229
## 12 chr04 1026608   C  0 20  0  0  0  1 19  0  0  0 1.000000e+00 0.07071068
## 13 chr04 1026683   C  0 25  0  0  0 21  0  0  0  0 1.440190e-13 1.41421356
## 14 chr04 1026799   C  2 32  0  0  0  2 32  0  0  0 1.000000e+00 0.00000000
## 15 chr04 1026833   C  1 15  0  0  0  0 29  0  0  0 3.555556e-01 0.08838835
## 16 chr04 1026916   T  0  0  1 19  0  0  0  0 20  0 1.000000e+00 0.07071068
```

A simple scatter plot will show the most different candidates locating at top-right.

plot(-log10(snpDiffDfp.value), snpDiffDfd.value,col='brown')

For more complex situation with hundreds of outputs, sorting the data frame by p.value and d.value would be very helpful to set custom cutoffs after mannually check.

```
snpDiffDfSort <- snpDiffDf[order(snpDiffDf$p.value,snpDiffDf$d.value),]
snpDiffDfSort
```

```
##      chr     pos ref A1 C1 G1 T1 N1 A2 C2 G2 T2 N2      p.value    d.value
## 2  chr04  962801   G  0  0 47  0  0  0  0  0 45  0 2.487221e-27 1.41421356
## 13 chr04 1026683   C  0 25  0  0  0 21  0  0  0  0 1.440190e-13 1.41421356
## 3  chr04  962865   C  2 29  0  0  0  0 49  0  0  0 1.471519e-01 0.09123958
## 1  chr04  962623   G  0  0 34  2  0  0  0 47  0  0 1.851308e-01 0.07856742
## 15 chr04 1026833   C  1 15  0  0  0  0 29  0  0  0 3.555556e-01 0.08838835
## 5  chr04  962998   T  0  0  1 14  0  0  0  0 25  0 3.750000e-01 0.09428090
## 7  chr04 1026413   T  1  0  0 15  0  0  0  0 21  0 4.324324e-01 0.08838835
## 8  chr04 1026421   C  0 17  1  0  0  0 22  0  0  0 4.500000e-01 0.07856742
## 9  chr04 1026533   T  0  0  0 22  0  0  0  1 18  0 4.634146e-01 0.07443229
## 14 chr04 1026799   C  2 32  0  0  0  2 32  0  0  0 1.000000e+00 0.00000000
## 6  chr04  963005   A 16  0  0  0  0 19  1  0  0  0 1.000000e+00 0.07071068
## 10 chr04 1026599   A 16  0  0  0  0 19  0  1  0  0 1.000000e+00 0.07071068
## 12 chr04 1026608   C  0 20  0  0  0  1 19  0  0  0 1.000000e+00 0.07071068
## 16 chr04 1026916   T  0  0  1 19  0  0  0  0 20  0 1.000000e+00 0.07071068
## 11 chr04 1026603   T  0  0  0 17  0  0  0  1 18  0 1.000000e+00 0.07443229
## 4  chr04  962984   G  0  0 28  0  0  0  1 30  1  0 1.000000e+00 0.07654655
```

4 Function indelDiff()

Detecting indel differences between two bam files is usually mislead by finding two indel lists seperately and then overlap them. Instead, `indelDiff` will firstly extract the read counts of each indel genotypes in two bam files at the same genomic position, and then calculate `p.value` of fisher exact test and `d.value` of Euclidean distance for this position. In case that the reads can't span the long indel, only reads that cover more than

2bp adjacent the given indel region are taken into consideration.

4.1 Input and Output

The input of `indelDiff` is the same as `snpDiff`, however, the output genotype names of `indelDiff` are different. For function `snpDiff`, 'A', 'T','G' and 'C' are four informative base, while for the output of `indelDiff`, the three genotypes are 'ref','altGt1' and 'altGt2', which means two alternative indel genotypes will be considered in the given position in both bam files.

4.2 Example

A simple input example and its output is

```
indelDiffDf <- indelDiff(bam1,bam2,refFsa,'chr07',828514,828914,pValueCutOff=1,gtDistCutOff=0)
```

```
## Warning in setup_parallel(): No parallel backend registered
```

```
## [1] "chr07"  "828636" "G"
## [1] "chr07"  "828714" "C"
```

```
indelDiffDfSort <- indelDiffDf[order(indelDiffDf$p.value,indelDiffDf$d.value),]
indelDiffDfSort
```

```
##     chr    pos      ref    altGt1 altGt2 refBam1Count altGt1Bam1Count
## 2 chr07 828714 CAAAAAAA CAAAAAAAA   <NA>            0               7
## 1 chr07 828636     GCCA   GCCACCA GTACCA           34               0
##   altGt2Bam1Count refBam2Count altGt1Bam2Count altGt2Bam2Count      p.value
## 2              NA           17               0              NA 2.889305e-06
## 1               0           34               1               1 1.000000e+00
##      d.value
## 2 1.41421356
## 1 0.06804138
```

5 Conclusion

Detecting SNV/Indel difference between two bam files is frequently used in many research fields of high-throughput sequencing. We thus provide these two simple R functions to make the comparison easy and accurate. Though the cutoff of `p.value` and `d.value` of `SICtools` are usually determined by cutsom data, a scatter plot `-log10(p.value) vs. d.value` is very helpful to achieve it based on our own expericence.

6 Session Info

The following package and versions were used in the production of this vignette.

```
sessionInfo()
```

```
## R version 4.3.1 (2023-06-16)
## Platform: aarch64-apple-darwin20 (64-bit)
## Running under: macOS Ventura 13.6.1
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRlapack.dylib;  LAPACK v
##
## locale:
## [1] C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
```

```
##
## time zone: America/New_York
## tzcode source: internal
##
## attached base packages:
## [1] parallel  stats4    stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
##  [1] SICtools_1.32.0      plyr_1.8.8           matrixStats_1.0.0
##  [4] stringr_1.5.0        doParallel_1.0.17    iterators_1.0.14
##  [7] foreach_1.5.2        Rsamtools_2.18.0     Biostrings_2.70.1
## [10] XVector_0.42.0       GenomicRanges_1.54.0 GenomeInfoDb_1.38.0
## [13] IRanges_2.36.0       S4Vectors_0.40.1     BiocGenerics_0.48.0
##
## loaded via a namespace (and not attached):
##  [1] vctrs_0.6.3             crayon_1.5.2            cli_3.6.1
##  [4] knitr_1.43              rlang_1.1.1             xfun_0.39
##  [7] stringi_1.7.12          glue_1.6.2              RCurl_1.98-1.12
## [10] htmltools_0.5.5         rmarkdown_2.23          evaluate_0.21
## [13] bitops_1.0-7            fastmap_1.1.1           lifecycle_1.0.3
## [16] yaml_2.3.7              compiler_4.3.1          codetools_0.2-19
## [19] Rcpp_1.0.11             BiocParallel_1.36.0     digest_0.6.33
## [22] GenomeInfoDbData_1.2.10 magrittr_2.0.3          tools_4.3.1
## [25] zlibbioc_1.48.0
```