A quick introduction to AneuFinder

Aaron Taudt *

*aaron.taudt@gmail.com

October 29, 2024

Contents

NOTE: This is the vignette for AneuFinder v1.6.0

1 Introduction

AneuFinder offers functionality for the study of copy number variations (CNV) in whole-genome single cell sequencing (WGSCS) data. Functionality implemented in this package includes:

- Copy number detection using a Hidden Markov Model or changepoint-algorithm on binned read counts.
- Various plotting capabilities like genomewide heatmaps of copy number state and arrayCGH-like plots.
- Export of copy number calls in BED format for upload to the UCSC genome browser.
- Quality metrics.
- Measures for addressing karyotype heterogeneity.

2 Quickstart

The main function of this package is called `Aneufinder()` and performs all the necessary steps to get from aligned reads to interpretable output. *AneuFinder* offers three methods to call copy number variations: (1) A Hidden Markov Model described in [1], (2) an approach based on the *DNAcopy* package adopted for single cells from [2], and (3) an approach described in [3] using the edivisive-algorithm from the *ecp* package.[1] We recommend using the "edivisive" method.

```
library(AneuFinder)
Aneufinder(inputfolder='folder-with-BAM-or-BED', outputfolder='output-directory',
           numCPU=2, method=c('edivisive', 'dnacopy','HMM'))
```

The above command will produce reasonably good results. However, quality of the results can be significantly improved by applying *mappability correction*, *blacklisting* and *GC-content correction* (see section 3). A description of all available parameters can be obtained by typing

```
?Aneufinder
```

After the function has finished, you will find the folder **output-directory** containing all produced files and plots. This folder contains the following *files* and **folders**:

- *AneuFinder.config*: This file contains all the parameters that are necessary to reproduce your analysis. You can specify this file as

  ```
  Aneufinder(..., configfile='AneuFinder.config')
  ```

  to run another analysis with the same parameter settings.
- *chrominfo.tsv*: A tab-separated file with chromosome length information.

[1] Please cite [1] if you use the *HMM* results. Please cite [1] and [2] if you use the *dnacopy* results. Please cite [1] and [3] if you use the default *edivisive* approach.

- **binned**: This folder contains the binned data. If you chose a correction method, you will also see a folder like **binned-GC** in case of GC correction. You can load the data with

```
files <- list.files('output-directory/binned', full.names=TRUE)
binned.data <- loadFromFiles(files)
```

- **BROWSERFILES**: A folder which contains BED files with copy number calls and breakpoint locations that can be uploaded to the UCSC genome browser. If reads.store=TRUE it will also contain a subfolder **data** with the mapped reads in BED format.
- **MODELS**: A folder with all produced Hidden Markov Models. You can load the results for further processing, such as quality control and customized plotting. The folder might be named **MODELS_refined** or **MODELS-StrandSeq**, depending on whether you chose to analyze Strand-seq data or refine breakpoints.

```
files <- list.files('output-directory/MODELS/method-edivisive', full.names=TRUE)
hmms <- loadFromFiles(files)
cl <- clusterByQuality(hmms)
heatmapGenomewide(cl$classification[[1]])
```

- **PLOTS**: All plots that are produced by default will be stored here.
- **data**: Only produced if reads.store=TRUE. This folder stores all the read data as RData objects. This exists mostly for internal usage, namely to estimate confidence intervals and refine breakpoints.

3 A detailed workflow

3.1 Mappability correction

The first step of your workflow should be the production of a reference file for mappability correction. Mappability correction is done via a variable-width binning approach (as compared to fixed-width bins) and requires a euploid reference. You can either simulate this reference file or take a real euploid reference. For optimal results we suggest to use a real reference, e.g. by merging BAM files of single cells from a euploid reference tissue. This can be achieved with the 'samtools merge' command (not part of R). Be careful: All CNVs that are present in the reference will lead to artifacts in the analysis later. This includes sex-chromosomes that are present in one copy only, so we advice to use a female reference and to exclude the Y-chromosome from the analysis. If you have no reference available, you can simulate one with the `simulateReads()` command. You can also skip this step and continue without mappability correction. The algorithm will use fixed-width bins in this case.

```
library(AneuFinder)

## Load human genome
library(BSgenome.Hsapiens.UCSC.hg19)

## Get a BAM file for the estimation of quality scores (adjust this to your experiment)
bamfile <- system.file("extdata", "BB150803_IV_074.bam", package="AneuFinderData")

## Simulate reads of length 51bp for human genome
```

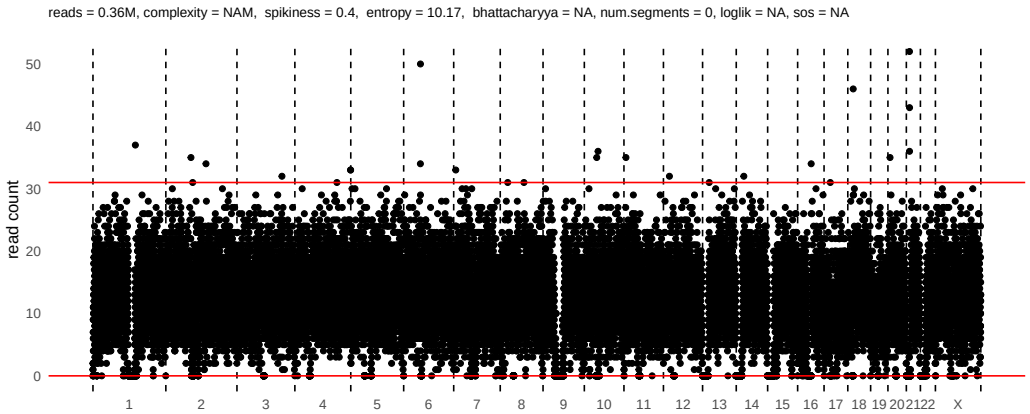
```
# We simulate reads every 500000bp for demonstration purposes, for a real
# application you should use a much denser spacing (e.g. 500bp or less)
simulatedReads.file <- tempfile() # replace this with your destination file
simulateReads(BSgenome.Hsapiens.UCSC.hg19, readLength=51, bamfile=bamfile,
              file=simulatedReads.file, every.X.bp=500000)
```

This simulated FASTQ file must then be aligned with your aligner of choice (ideally the same that you used for your other samples) and given as reference in `Aneufinder(..., variable.width.reference="your-reference-bamfile")`.

3.2 Blacklisting

To further improve the quality of the results and remove artifacts caused by high mappability repeat regions, e.g. near centromers, a blacklist can be used with option `Aneufinder(..., blacklist)`. All reads falling into the regions specified by the blacklist will be discarded when importing the read files. You can either download a blacklist from the UCSC genome browser, e.g. the "DAC Blacklisted Regions from ENCODE/DAC(Kundaje)" mappability track, or make your own. For optimal results, we advice to make your own blacklist from a euploid reference. The following code chunck takes a euploid reference and makes fixed-width bins of 100kb. Bins with read count above and below the 0.999 and 0.05 quantile are taken as blacklist:

```
library(AneuFinder)

## Get a euploid reference (adjust this to your experiment)
bedfile <- system.file("extdata", "hg19_diploid.bam.bed.gz", package="AneuFinderData")

## Make 100kb fixed-width bins
bins <- binReads(bedfile, assembly='hg19', binsizes=100e3,
                 chromosomes=c(1:22,'X'))[[1]]
## Make a plot for visual inspection and get the blacklist
lcutoff <- quantile(bins$counts, 0.05)
ucutoff <- quantile(bins$counts, 0.999)
p <- plot(bins) + coord_cartesian(ylim=c(0,50))
p <- p + geom_hline(aes(yintercept=lcutoff), color='red')
p <- p + geom_hline(aes(yintercept=ucutoff), color='red')
print(p)
```



```
## Select regions that are above or below the cutoff as blacklist
blacklist <- bins[bins$counts <= lcutoff | bins$counts >= ucutoff]
blacklist <- reduce(blacklist)
## Write blacklist to file
blacklist.file <- tempfile()
exportGRanges(blacklist, filename=blacklist.file, header=FALSE,
              chromosome.format='NCBI')
```

3.3 GC-content correction

GC-content can greatly bias the number of read fragments in a WGSCS-experiment. Therefore it is essential to apply a GC-correction step when calling copy-number-variations with `Aneufinder()`. GC-correction can be easily enabled with

```
library(AneuFinder)

## Library for GC correction
library(BSgenome.Hsapiens.UCSC.hg19)

## Produce output files
outputfolder <- tempdir()
Aneufinder(inputfolder = datafolder, outputfolder = outputfolder, assembly = 'hg19',
           correction.method = 'GC', GC.BSgenome = BSgenome.Hsapiens.UCSC.hg19,
           method = 'edivisive')
```

3.4 Running Aneufinder

The function `Aneufinder()` takes an input folder with BAM or BED files and produces an output folder with results, plots and browserfiles. The following code is an example of how to run `Aneufinder()` with variable-width bins (see section 3.1), blacklist (see section 3.2) and GC-correction. Results will be stored in **outputfolder/MODELS** as RData objects for further processing such as quality filtering and customized plotting.

```
library(AneuFinder)

## First, get some data and reference files (adjust this to your experiment)
var.width.ref <- system.file("extdata", "hg19_diploid.bam.bed.gz", package="AneuFinderData")
blacklist <- system.file("extdata", "blacklist-hg19.bed.gz", package="AneuFinderData")
datafolder <- system.file("extdata", "B-ALL-B", package = "AneuFinderData")
list.files(datafolder) # only 3 cells for demonstration purposes

## [1] "MB140210_I_003.bam.bed.gz" "MB140210_I_004.bam.bed.gz" "MB140210_I_005.bam.bed.gz"

## Library for GC correction
library(BSgenome.Hsapiens.UCSC.hg19)

## Produce output files
outputfolder <- tempdir()
Aneufinder(inputfolder = datafolder, outputfolder = outputfolder, assembly = 'hg19',
           numCPU = 3, binsizes = c(5e5, 1e6), variable.width.reference = var.width.ref,
           chromosomes = c(1:22,'X','Y'), blacklist = blacklist, correction.method = 'GC',
```

```
                GC.BSgenome = BSgenome.Hsapiens.UCSC.hg19, refine.breakpoints=FALSE,
                method = 'edivisive')
```

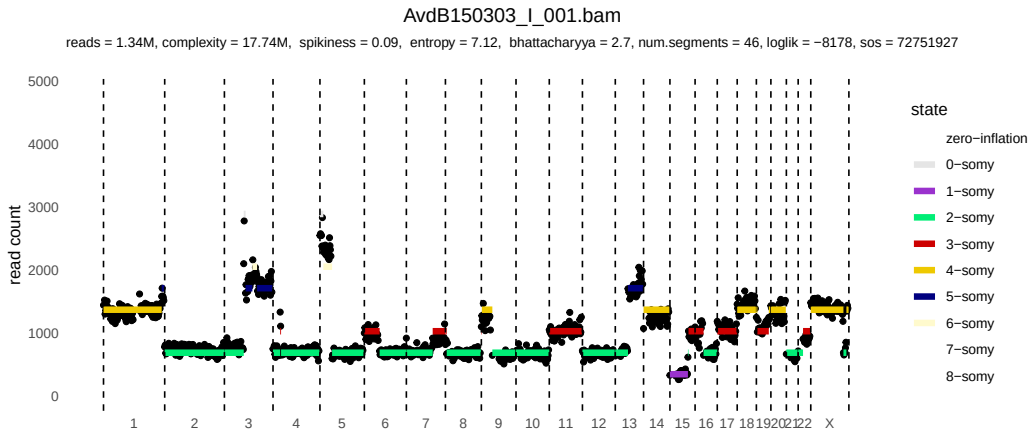
3.5 Loading results and plotting single cells

Once the function `Aneufinder()` has completed, results will be accessible as *.RData* files under **outputfolder/MODELS**. You can load the results into R using

```
library(AneuFinder)
files <- list.files('outputfolder/MODELS/method-edivisive', full.names=TRUE)
models <- loadFromFiles(files)
```

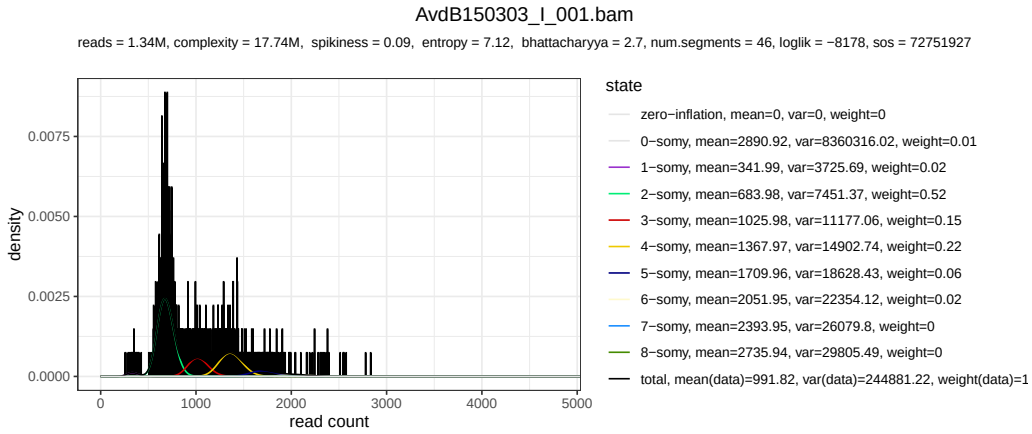
Here are some examples of the plotting functions that are available. Most of these plots are also produced by default by `Aneufinder()` and are available as PDF in **outputfolder/PLOTS**.

```
## Get some pre-produced results (adjust this to your experiment)
results <- system.file("extdata", "primary-lung", "hmms", package="AneuFinderData")
files <- list.files(results, full.names=TRUE)
plot(files[1], type='profile')
```



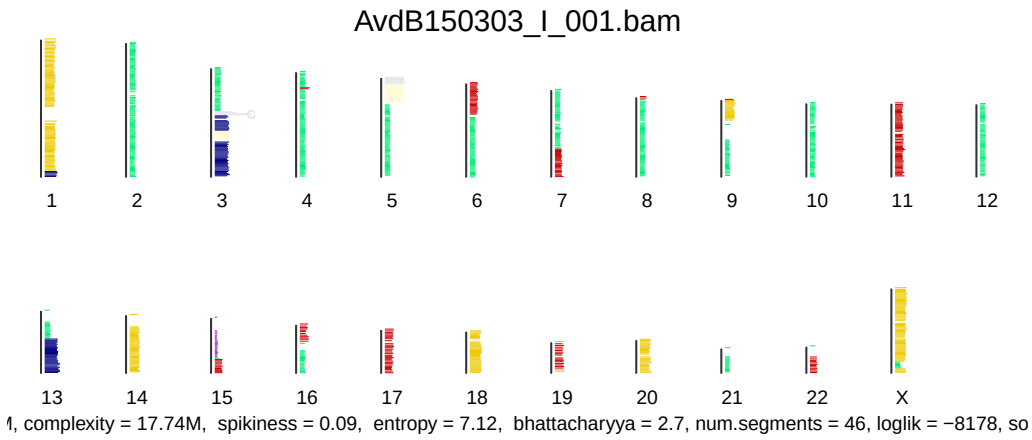

```
plot(files[1], type='histogram')
```

```
## Warning:  The dot-dot notation (‘..density..‘) was deprecated in ggplot2 3.4.0.
## i Please use ‘after_stat(density)‘ instead.
## i The deprecated feature was likely used in the AneuFinder package.
##   Please report the issue to the authors.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was generated.
```




```
plot(files[1], type='karyogram')
```

```
## Warning in plot.karyogram(model, both.strands = both.strands, file = file):  Cannot
find breakpoint coordinates.  Please run 'getBreakpoints' first.
```

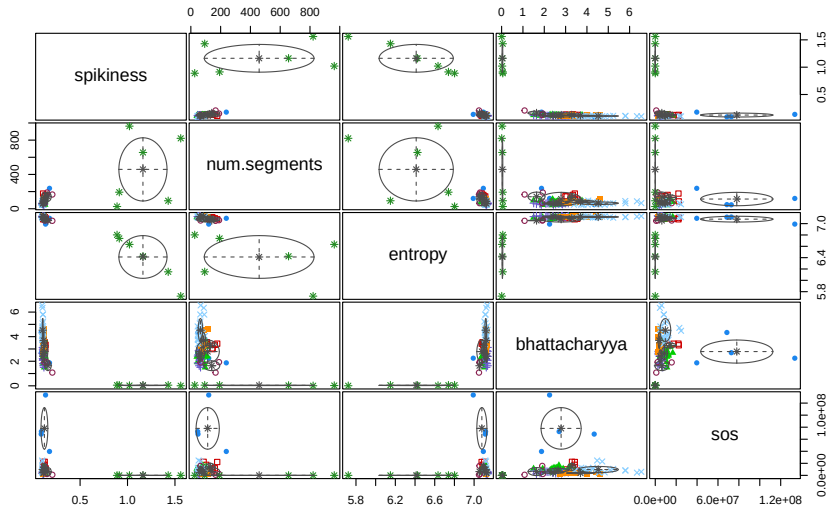


3.6 Quality control

Once the function `Aneufinder()` has completed, results will be accessible as .RData files under **outputfolder/MODELS**. Single cell sequencing is prone to noise and therefore it is a good idea to filter the results by quality to retain only high-quality cells. We found that simple filtering procedures such as cutoffs on the total number of reads etc., are insufficient to distinguish good from bad-quality libraries. Therefore, we have implemented a multivariate clustering approach that works on multiple quality metrics (see ?clusterByQuality for details) to increase robustness of the filtering. Here is an example demonstrating the usage of `clusterByQuality()`.

```
library(AneuFinder)

## Get some pre-produced results (adjust this to your experiment)
results <- system.file("extdata", "primary-lung", "hmms", package="AneuFinderData")
files <- list.files(results, full.names=TRUE)
```

```
## Cluster by quality, please type ?getQC for other available quality measures
cl <- clusterByQuality(files, measures=c('spikiness','num.segments','entropy','bhattacharyya','sos'))
plot(cl$Mclust, what='classification')
```

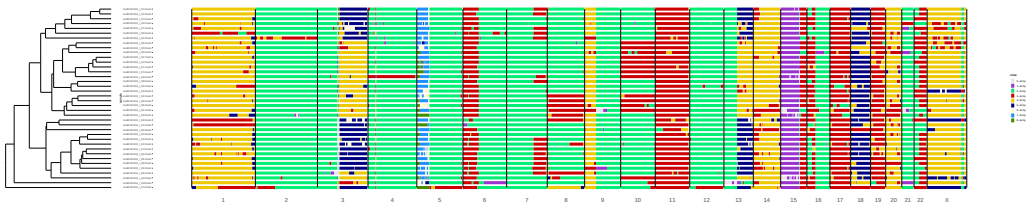



```
print(cl$parameters)
```

```
##      spikiness num.segments  entropy bhattacharyya          sos
## [1,] 0.1060501     64.30637 7.118107    4.52769354  9671255.144
## [2,] 0.1119017     58.86160 7.126918    2.41535861  4722433.201
## [3,] 0.1162552     85.34653 7.124391    3.69585953  3842048.636
## [4,] 0.1182611     72.81237 7.105328    2.64484856  8766630.278
## [5,] 0.1238624    113.06280 7.078982    2.78844568 77991481.726
## [6,] 0.1261168    130.20767 7.095900    3.28808888 14095817.324
## [7,] 0.1611050    140.16982 7.055041    1.63693596  6301025.747
## [8,] 1.1625668    458.66667 6.411838    0.03769717     1156.659
```

```
## Apparently, the last cluster corresponds to failed libraries
## while the first cluster contains high-quality libraries
```

```
## Select the two best clusters and plot it
selected.files <- unlist(cl$classification[1:2])
heatmapGenomewide(selected.files)
```



3.7 Karyotype measures

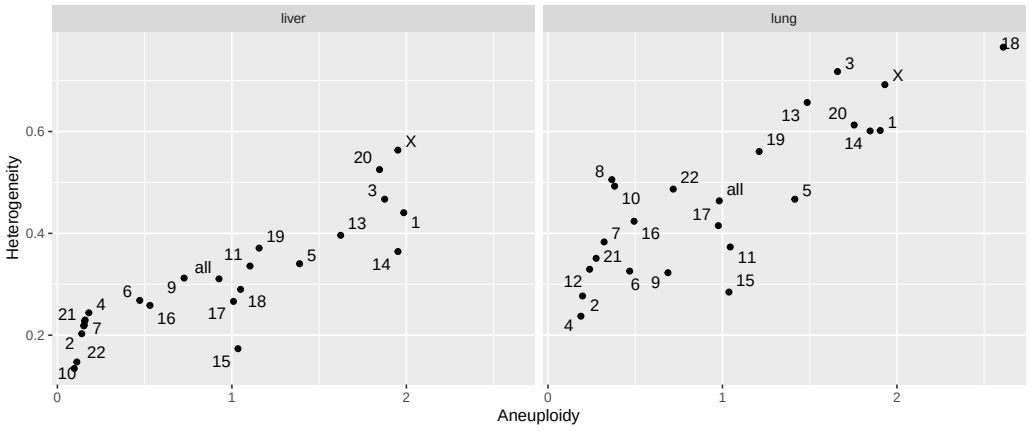
This package implements two measures to quantify karyotype heterogeneity, an *aneuploidy* and a *heterogeneity* score. Both measures are independent of the number of cells, the length of the genome, and take into account every position in the genome. The following example compares the heterogeneity and aneuploidy between a primary lung cancer and the corresponding liver metastasis.

```
library(AneuFinder)

## Get some pre-produced results (adjust this to your experiment)
results <- system.file("extdata", "primary-lung", "hmms", package="AneuFinderData")
files.lung <- list.files(results, full.names=TRUE)
results <- system.file("extdata", "metastasis-liver", "hmms", package="AneuFinderData")
files.liver <- list.files(results, full.names=TRUE)

## Get karyotype measures
k.lung <- karyotypeMeasures(files.lung)
k.liver <- karyotypeMeasures(files.liver)

## Print the scores in one data.frame
df <- rbind(lung = k.lung$genomewide, liver = k.liver$genomewide)
print(df)
```

```
##       Aneuploidy Heterogeneity
## lung   0.9818370     0.4638431
## liver  0.9262749     0.3106615
```

```
## While the aneuploidy is similar between both cancers, the heterogeneity is
## nearly twice as high for the primary lung cancer.
plotHeterogeneity(hmms.list = list(lung=files.lung, liver=files.liver))
```

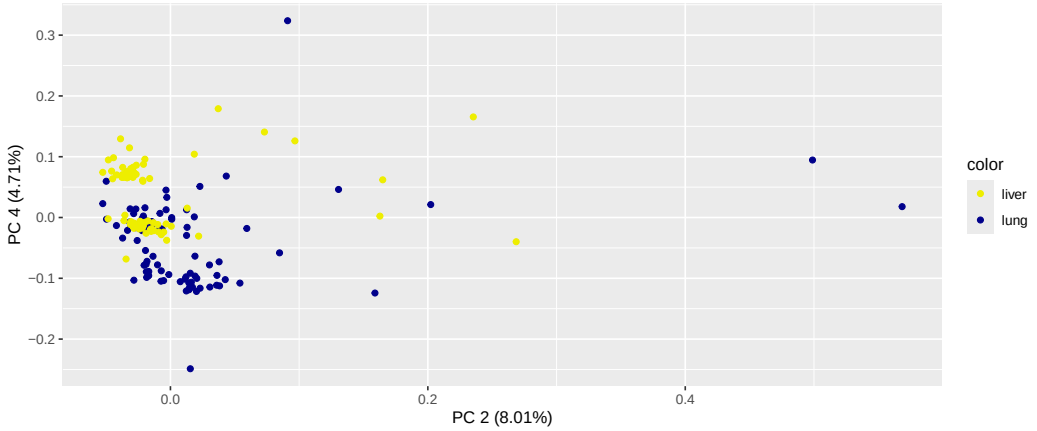
```
## Warning: ggrepel: 2 unlabeled data points (too many overlaps). Consider increasing
max.overlaps
```



3.8 Principal component analysis

The following code let's you plot a PCA for a selection of single cells.

```
## Get results from a small-cell-lung-cancer
lung.folder <- system.file("extdata", "primary-lung", "hmms", package="AneuFinderData")
lung.files <- list.files(lung.folder, full.names=TRUE)
## Get results from the liver metastasis of the same patient
liver.folder <- system.file("extdata", "metastasis-liver", "hmms", package="AneuFinderData")
liver.files <- list.files(liver.folder, full.names=TRUE)
## Plot a clustered heatmap
classes <- c(rep('lung', length(lung.files)), rep('liver', length(liver.files)))
labels <- c(paste('lung',1:length(lung.files)), paste('liver',1:length(liver.files)))
plot_pca(c(lung.files, liver.files), colorBy=classes, PC1=2, PC2=4)
```



4 Session Info

```
toLatex(sessionInfo())
```

- R Under development (unstable) (2024-10-21 r87258), `x86_64-pc-linux-gnu`
- Locale: `LC_CTYPE=en_US.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=en_GB`, `LC_COLLATE=C`, `LC_MONETARY=en_US.UTF-8`, `LC_MESSAGES=en_US.UTF-8`, `LC_PAPER=en_US.UTF-8`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.UTF-8`, `LC_IDENTIFICATION=C`
- Time zone: `America/New_York`
- TZcode source: `system (glibc)`
- Running under: `Ubuntu 24.04.1 LTS`
- Matrix products: default
- BLAS: `/home/biocbuild/bbs-3.21-bioc/R/lib/libRblas.so`
- LAPACK: `/usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.12.0`
- Base packages: base, datasets, grDevices, graphics, methods, stats, stats4, utils
- Other packages: AneuFinder 1.35.0, AneuFinderData 1.33.0, BSgenome 1.75.0, BSgenome.Hsapiens.UCSC.hg19 1.4.3, BiocGenerics 0.53.0, BiocIO 1.17.0, Biostrings 2.75.0, GenomeInfoDb 1.43.0, GenomicRanges 1.59.0, IRanges 2.41.0, S4Vectors 0.45.0, XVector 0.47.0, cowplot 1.1.3, ggplot2 3.5.1, rtracklayer 1.67.0

- Loaded via a namespace (and not attached): Biobase 2.67.0, BiocManager 1.30.25, BiocParallel 1.41.0, BiocStyle 2.35.0, DNAcopy 1.81.0, DelayedArray 0.33.0, GenomeInfoDbData 1.2.13, GenomicAlignments 1.43.0, MASS 7.3-61, Matrix 1.7-1, MatrixGenerics 1.19.0, R6 2.5.1, RCurl 1.98-1.16, Rcpp 1.0.13, Rsamtools 2.23.0, S4Arrays 1.7.0, SparseArray 1.7.0, SummarizedExperiment 1.37.0, UCSC.utils 1.3.0, XML 3.99-0.17, abind 1.4-8, bamsignals 1.39.0, bitops 1.0-9, cli 3.6.3, codetools 0.2-20, colorspace 2.1-1, compiler 4.5.0, crayon 1.5.3, curl 5.2.3, digest 0.6.37, doParallel 1.0.17, dplyr 1.1.4, ecp 3.1.6, evaluate 1.0.1, fansi 1.0.6, farver 2.1.2, fastmap 1.2.0, foreach 1.5.2, generics 0.1.3, ggdendro 0.2.0, ggrepel 0.9.6, glue 1.8.0, grid 4.5.0, gtable 0.3.6, highr 0.11, htmltools 0.5.8.1, httr 1.4.7, iterators 1.0.14, jsonlite 1.8.9, knitr 1.48, labeling 0.4.3, lattice 0.22-6, lifecycle 1.0.4, magrittr 2.0.3, matrixStats 1.4.1, mclust 6.1.1, munsell 0.5.1, parallel 4.5.0, pillar 1.9.0, pkgconfig 2.0.3, plyr 1.8.9, reshape2 1.4.4, restfulr 0.0.15, rjson 0.2.23, rlang 1.1.4, rmarkdown 2.28, scales 1.3.0, stringi 1.8.4, stringr 1.5.1, tibble 3.2.1, tidyselect 1.2.1, tinytex 0.53, tools 4.5.0, utf8 1.2.4, vctrs 0.6.5, withr 3.0.2, xfun 0.48, yaml 2.3.10, zlibbioc 1.53.0

References

[1] Bjorn Bakker, Aaron Taudt, Mirjam E. Belderbos, David Porubsky, Diana C. J. Spierings, Tristan V. de Jong, Nancy Halsema, Hinke G. Kazemier, Karina Hoekstra-Wakker, Allan Bradley, Eveline S. J. M. de Bont, Anke van den Berg, Victor Guryev, Peter M. Lansdorp, Maria Colomé-Tatché, and Floris Foijer. Single-cell sequencing reveals karyotype heterogeneity in murine and human malignancies. *Genome Biology*, 17(1):115, may 2016. doi:10.1186/s13059-016-0971-7.

[2] Tyler Garvin, Robert Aboukhalil, Jude Kendall, Timour Baslan, Gurinder S Atwal, James Hicks, Michael Wigler, and Michael C Schatz. Interactive analysis and assessment of single-cell copy-number variations. *Nature Methods*, 12(11):1058–1060, sep 2015. doi:10.1038/nmeth.3578.

[3] Aaron Sebastian Taudt. *Hidden Markov models for the analysis of next-generation-sequencing data*. PhD thesis, University of Groningen, 2018.