

# Package ‘SomaticSignatures’

May 7, 2024

**Type** Package

**Title** Somatic Signatures

**Version** 2.40.0

**Author** Julian Gehring

**Maintainer** Julian Gehring <jg-bioc@gmx.com>

**Description** The SomaticSignatures package identifies mutational signatures of single nucleotide variants (SNVs). It provides a infrastructure related to the methodology described in Nik-Zainal (2012, Cell), with flexibility in the matrix decomposition algorithms.

**URL** <https://github.com/juliangehring/SomaticSignatures>

**Imports** S4Vectors, IRanges, GenomeInfoDb, Biostrings, ggplot2, ggbio, reshape2, NMF, pcaMethods, Biobase, methods, proxy

**Depends** R (>= 3.1.0), VariantAnnotation, GenomicRanges, NMF

**Suggests** testthat, knitr, parallel, BSgenome.Hsapiens.1000genomes.hs37d5, SomaticCancerAlterations, ggdendro, fastICA, sva

**VignetteBuilder** knitr

**ByteCompile** TRUE

**License** MIT + file LICENSE

**BugReports** <https://support.bioconductor.org>

**LazyLoad** yes

**biocViews** Sequencing, SomaticMutation, Visualization, Clustering, GenomicVariation, StatisticalMethod

**git\_url** <https://git.bioconductor.org/packages/SomaticSignatures>

**git\_branch** RELEASE\_3\_19

**git\_last\_commit** c99d3e9

**git\_last\_commit\_date** 2024-04-30

**Repository** Bioconductor 3.19

**Date/Publication** 2024-05-06

## Contents

cluster-spectrum . . . . .	2
decomposition-signatures . . . . .	3
gcContent . . . . .	4
GRanges-converters . . . . .	4
hs-chrs . . . . .	5
kmerFrequency . . . . .	6
kmers-data . . . . .	7
motif-functions . . . . .	7
mutation-distribution . . . . .	8
mutational-normalization . . . . .	9
mutational-plots . . . . .	10
mutational-signatures . . . . .	10
MutationalSignatures . . . . .	11
mutationContext . . . . .	12
numberSignatures . . . . .	13
readMutect . . . . .	15
sca-data . . . . .	16
signature-plots . . . . .	16
signatures21-data . . . . .	18
SomaticSignatures . . . . .	18
variants-utils . . . . .	19
<b>Index</b>	<b>21</b>

---

cluster-spectrum	<i>Cluster Mutational Spectrum</i>
------------------	------------------------------------

---

### Description

Cluster the mutational spectrum by sample or motif.

### Usage

```
clusterSpectrum(m, by = c("sample", "motif"), distance = "Cosine", ...)
```

### Arguments

m	Mutational spectrum matrix
by	Dimension to cluster by.
distance	Distance function used in the clustering.
...	Additional arguments passed to 'hclust'.

### Details

Hierarchical clustering of the motif matrix aka mutational spectrum.

**Value**

An 'hclust' object.

**See Also**

[hclust](#)

[dist](#)

---

decomposition-signatures

*Decomposition Functions for Somatic Signatures*

---

**Description**

Estimate somatic signatures from sequence motifs with a selection of statistical methods.

**Usage**

```
nmfDecomposition(x, r, ..., includeFit = FALSE)
```

```
pcaDecomposition(x, r, ..., includeFit = FALSE)
```

**Arguments**

x	GRanges object [required]
r	Number of signatures [integer, required]
...	Additional arguments passed to 'NMF::nmf' or 'pcaMethods::pca'.
includeFit	Include the fit object returned by the low-level decomposition function in the output.

**Details**

The 'nmfDecomposition' and 'pcaDecomposition' functions estimate a set of 'r' somatic signatures using the NMF or PCA, respectively.

In previous versions of the package, these functions were known as 'nmfSignatures' and 'pcaSignatures', respectively. While they are still available, we recommend using the new naming convention.

**Value**

The 'signature' functions return a list with the elements:

- wMatrix of the form 'motif x signature'
- hMatrix of the form 'sample x signature'
- vMatrix of the form 'motif x sample', containing the reconstruction of 'm' from 'w' and 'h'.
- mInput matrix 'm'
- rNumber of signatures.
- fitFit object returned by the low-level decomposition function, if 'includeFit' is true.

**See Also**

[NMF](#) package  
[pcaMethods](#) package  
[prcomp](#)

---

gcContent	<i>GC Content</i>
-----------	-------------------

---

**Description**

Compute the GC content for regions of a reference sequence.

**Usage**

```
gcContent(regions, ref)
```

**Arguments**

regions	GRanges object with the regions for which the GC content should be computed.
ref	Reference sequence object, as a 'BSgenome' or 'FaFile' object.

**Value**

A numeric vector with the GC content [0,1] for each region.

**Examples**

```

library(BSgenome.Hsapiens.1000genomes.hs37d5)

regs = GRanges(c("1", "2"), IRanges(1e7, width = 100))

gc = gcContent(regs, BSgenome.Hsapiens.1000genomes.hs37d5)

```

---

GRanges-converters	<i>GRanges converter functions</i>
--------------------	------------------------------------

---

**Description**

A set of utilities functions to convert and extract data in 'GRanges' objects.

**Usage**

```

ncbi(x)
ucsc(x)
seqchar(x)

```

## Arguments

x                    A 'GRanges' object or one inheriting from the 'GRanges' class [required].

## Details

- `grangesExtracts` only the 'GRanges' information by dropping the metadata columns of the object. The 'seqinfo' slot is kept.
- `ncbi`, `ucscShorthand` for converting the seqnames notation to 'UCSC' (e.g. 'chr1', 'chrM') or 'NCBI' (e.g. '1', 'MT') notation, respectively. This also sets the 'genome' slot in the 'seqinfo' field to 'NA'.
- `seqcharExtracts` the 'seqnames' as a character vector.

## Value

For 'ncbi', 'ucsc': An object of the same class as the input.

For 'seqchar': A character vector with 'seqnames'.

## See Also

[seqnames](#), [mcols](#)

[seqlevelsStyle](#)

## Examples

```
mutect_path = system.file("examples", "mutect.tsv", package = "SomaticSignatures")
vr1 = readMutect(mutect_path, strip = TRUE)

## extract the GRanges
gr = granges(vr1)

## convert back and forth
gr_ncbi = ncbi(gr)
gr_ucsc = ucsc(gr_ncbi)

identical(gr, gr_ucsc)

## extract the seqnames as a character vector
seq_chars = seqchar(gr)
```

---

hs-chrs

*Human Chromosome Names*

---

## Description

List human chromosome names.

**Usage**

```

hsToplevel()
hsAutosomes()
hsAllosomes()
hsLinear()

```

**Value**

Character vector with chromosome names (NCBI notation).

**Examples**

```

hsToplevel()

hsAutosomes()

hsAllosomes()

hsLinear()

```

---

kmerFrequency

*Kmer Frequency*


---

**Description**

Estimate the occurrence frequency of k-mers in a reference sequence.

**Usage**

```
kmerFrequency(ref, n = 1e4, k = 1, ranges = as(seqinfo(ref), "GRanges"))
```

**Arguments**

ref	A 'BSgenome' or 'FaFile' object matching the respective reference sequence [required].
n	The number of samples to draw [integer, default: 1e4].
k	The 'k'-mer size of the context, including the variant position [integer, default: 3].
ranges	Ranges in respect to the reference sequence to sample from [GRanges, default: take from the 'seqinfo' slot].

**Details**

The k-mer frequency is estimated by random sampling of 'n' locations across the specified 'ranges' of the reference sequence.

**Value**

A named vector, with names corresponding the the k-mer and value to the frequency.

**Examples**

```
library(BSgenome.Hsapiens.1000genomes.hs37d5)

kmer_freq = kmerFrequency(BSgenome.Hsapiens.1000genomes.hs37d5, 1e2, 3)
```

---

 kmers-data

*Kmer datasets*


---

**Description**

3mer base frequencies of human whole-genome and whole-exome sampling, based on the hg19/GRCh37 reference sequence.

For details, see the 'inst/scripts/kmers-data.R' script.

**Value**

Vectors with frequency of k-mers.

**See Also**

[kmerFrequency](#)

**Examples**

```
data(kmers, package = "SomaticSignatures")
```

---

 motif-functions

*Group somatic motifs*


---

**Description**

Tabulate somatic motifs by a grouping variable.

**Usage**

```
motifMatrix(vr, group = "sampleNames", normalize = TRUE)
```

**Arguments**

vr	GRanges object [required]
group	Grouping variable name [character, default: 'sampleNames']
normalize	Normalize to frequency

**Details**

The `'motifMatrix'` function transforms the metadata columns of a `'VRanges'` object, as returned by the `'mutationContext'` function, to a matrix of the form `'motifs x groups'`. This constitutes the bases for the estimation of the signatures. By default (with `'normalize'` set to `TRUE`), the counts are transformed to frequencies, such that the sum of frequencies of each group equal 1. Otherwise (with `'normalize'` set to `FALSE`), the counts for each motifs in a group is returned.

**Value**

Occurance matrix with motifs in rows and samples in columns.

**See Also**

`'mutationContext'`, `'mutationContextMutect'`

**Examples**

```
data(sca_motifs_tiny)
motifMatrix(sca_motifs_tiny, group = "study")
```

---

mutation-distribution *Distributions of mutational locations.*

---

**Description**

Summary and plotting function for characterizing the distributions of mutations along the genome.

**Usage**

```
mutationDistance(x)

plotRainfall(x, group, size = 2, alpha = 0.5, space.skip = 0, ...)
```

**Arguments**

<code>x</code>	A <code>'GRanges'</code> or <code>'VRanges'</code> object [required].
<code>group</code>	The variable name for color groups [optional].
<code>size</code>	Point size [default: 2]
<code>alpha</code>	Alpha value for points [default: 0.5]
<code>space.skip</code>	Space between chromosomes, as defined by <code>'plotGrandLinear'</code> [default: 0]
<code>...</code>	Additional arguments passed to <code>'plotGrandLinear'</code>



**Value**

- `mutationDensity` The position-sorted `GRanges` 'x' with the additional column 'distance', specifying the distance from the previous mutation (or the beginning of the chromosome if it happens to be the first mutation on the chromosome.)
- `plotRainfall` Object of class 'ggbio', as returned by 'plotGrandLinear'.

**See Also**

`plotGrandLinear` from the 'ggbio' package

**Examples**

```
library(GenomicRanges)
library(IRanges)

set.seed(1)
chr_len = 100
gr = GRanges(rep(1:3, each = 10),
  IRanges(start = sample.int(chr_len, 30, replace = FALSE), width = 1),
  mutation = sample(c("A", "C", "G", "T"), 30, replace = TRUE))
seqlengths(gr) = rep(chr_len, 3)

p = plotRainfall(gr)
print(p)
```

---

mutational-normalization

*Normalize Somatic Motifs*

---

**Description**

Normalize somatic motifs, to correct for biases between samples.

**Usage**

```
normalizeMotifs(x, norms)
```

**Arguments**

x	Matrix, as returned by 'motifMatrix' [required]
norms	Vector with normalization factors [required]. The names must match the base sequence names in 'x'.

**Value**

A matrix as 'x' with normalized counts.

**See Also**[motifMatrix](#)


---

mutational-plots      *Mutational Plots*


---

**Description**

Plots for variant analysis

**Usage**

```
plotVariantAbundance(x, group = NULL, alpha = 0.5, size = 2)
```

**Arguments**

x	A VRanges object [required].
group	Grouping variable, refers to a column name in 'x'. By default, no grouping is performed.
alpha	Alpha value for data points.
size	Size value for data points.

**Details**

The 'plotVariantAbundance' shows the variant frequency in relation to the total coverage at each variant position. This can be useful for examining the support of variant calls.

**Value**

A 'ggplot' object.

---

mutational-signatures      *Estimate Somatic Signatures*


---

**Description**

Estimate somatic signatures from sequence motifs with a selection of statistical methods.

**Usage**

```
identifySignatures(m, nSigs, decomposition = nmfDecomposition, ...)
```

**Arguments**

<code>m</code>	Motif matrix, as returned by <code>'motifMatrix'</code> [required].
<code>nSigs</code>	Number of signatures [integer, required].
<code>decomposition</code>	Function to apply for the matrix decomposition. The methods NMF and PCA are already implemented in the functions <code>'nmfDecomposition'</code> and <code>'pcaDecomposition'</code> , respectively.
<code>...</code>	Additional arguments passed to the <code>'decomposition'</code> function.

**Details**

`'identifySignatures'` estimate a set of `'r'` somatic signatures, based on a matrix decomposition method (such as NMF, PCA).

**Value**

An object of class `'MutationalSignatures'`.

**See Also**

The predefined decomposition functions: [nmfDecomposition](#) and [pcaDecomposition](#)  
[mutationContext](#), [mutationContextMutect](#)  
[motifMatrix](#)  
[MutationalSignatures](#) class

**Examples**

```
data("sca_mm", package = "SomaticSignatures")  
  
sigs = identifySignatures(sca_mm, 5)
```

---

MutationalSignatures *'MutationalSignatures' class and methods*

---

**Description**

Object representing of somatic signatures.

**Usage**

```
## S4 method for signature 'MutationalSignatures'  
signatures(object)  
  
## S4 method for signature 'MutationalSignatures'  
samples(object)  
  
## S4 method for signature 'MutationalSignatures'
```

```

observed(object)

## S4 method for signature 'MutationalSignatures'
fitted(object)

## S4 method for signature 'MutationalSignatures'
show(object)

```

### Arguments

object            'MutationalSignatures' object

### Value

help("MutationalSignatures")

### See Also

[identifySignatures](#)

---

mutationContext	<i>mutationContext functions</i>
-----------------	----------------------------------

---

### Description

Extract the sequence context surrounding SNVs from a genomic reference.

### Usage

```

mutationContext(vr, ref, k = 3, strand = FALSE, unify = TRUE, check = FALSE)
mutationContextMutect(vr, k = 3, unify = TRUE)

```

### Arguments

vr	'VRanges' with SNV substitutions, with 'ref' and 'alt' columns filled [required]. Each element of 'ref' and 'alt' have be a single base from the DNA bases (A,C,G,T). For 'mutationContextMutect', an object as returned by the 'readMutect' function.
ref	A 'BSgenome', 'FaFile' or 'TwoBitfile' object representing the reference sequence [required]. More generally, any object with a defined 'getSeq' method can be used.
k	The 'k'-mer size of the context, including the variant position [integer, default: 3]. The variant will be located at the middle of the k-mer which requires 'k' to be odd.
strand	Should all variants be converted to the 'plus' strand? [logical, default: FALSE].

unify	Should the alterations be converted to have a C/T base pair as a reference alleles? [logical, default: TRUE]
check	Should the reference base of 'vr' be checked against 'ref' [logical, default: TRUE]? In case the two references do not match, a warning will be printed.

### Details

The somatic motifs of a SNV, composed out of (a) the base change and (b) the sequence context surrounding the variant, is extracted from a genomic sequence with the 'mutationContext' function. Different types of classes that represent the genomic sequence can be used together with the 'mutationContext' function: 'BSgenome', 'FastaFile' and 'TwoBitFile' objects are supported through Bioconductor by default. See the vignette for examples discussing an analysis with non-reference genomes.

For mutect variant calls, all relevant information is already contained in the results and somatic motifs can be constructed by using the 'mutationContextMutect' function, without the need for the reference sequence.

### Value

The original 'VRanges' object 'vr', with the additional columns

alteration	DNAStrngSet with 'refalt'.
context	DNAStrngSet with '..N..' of length 'k', where N denotes the variant position.

### See Also

[readMutect](#) for mutationContextMutect

'showMethods("getSeq")' for genomic references that can be used

### Examples

```
mutect_path = system.file("examples", "mutect.tsv", package = "SomaticSignatures")
vr1 = readMutect(mutect_path)
ct1 = mutationContextMutect(vr1)
```

---

numberSignatures	<i>Number of Signatures</i>
------------------	-----------------------------

---

### Description

Assessment of the number of signatures in the data.

### Usage

```
assessNumberSignatures(m, nSigs, decomposition = nmfDecomposition, ..., nReplicates = 1)

plotNumberSignatures(gof)
```

**Arguments**

<code>m</code>	Mutational spectrum matrix, same as used for 'identifySignatures'.
<code>nSigs</code>	Vector of integers with the numbers of signatures that should be tested. See the 'nSigs' argument for 'identifySignatures'.
<code>decomposition</code>	Function to apply for the matrix decomposition. See the 'decomposition' argument for 'identifySignatures'.
<code>...</code>	Additional arguments passed to the 'decomposition' function. See the '...' argument for 'identifySignatures'.
<code>nReplicates</code>	How many runs should be used for assessing a value of 'nSigs'? For decomposition methods with random seeding, values greater than 1 should be used.
<code>gof</code>	Data frame, as returned of 'assessNumberSignatures'
.	.

**Details**

Compute the decomposition for a given number of signatures, and assess the goodness of the reconstruction between the observed and fitted mutational spectra  $M$  and  $V$ , respectively. The residual sum of squares (RSS)

$$RSS = \sum_{i,j} (M_{ij} - V_{ij})^2$$

and the explained variance

$$evar = 1 - \frac{RSS}{\sum_{i,j} V_{ij}^2}$$

are used as summary statistics which can generally applied to all decomposition approaches.

The 'plotNumberSignatures' function visualizes the results of the 'assessNumberSignatures' analysis. Statistics of the individual runs are shown as gray crosses, whereas the mean across the runs is depicted in red.

If a decomposition method uses random seeding and hence recomputing the decomposition of the same data can yield different results, evaluating the summary statistics will give more reliable estimates of the number of signatures. This applies to some NMF algorithms, for example. Methods with a deterministic decomposition, such as the standard PCA, do not need this, since repeated computations will yield the same decomposition. This behaviour is controlled by the 'nReplicates' parameter, where the default of '1' corresponds to a single run.

In practice, these summary statistics should not be trusted blindly, but rather interpreted together with biological knowledge and scientific reasoning. For a discussion of the interpretation of these statistics with special focus on the NMF decomposition, please refer to the references listed below.

**Value**

- assessNumberSignatures: A data frame with the RSS and explained variance for each run
- plotNumberSignatures: A ggplot object

## References

Hutchins LN, Murphy SM, Singh P and Graber JH (2008): 'Position-dependent motif characterization using non-negative matrix factorization.' *Bioinformatics*, <http://dx.doi.org/10.1093/bioinformatics/btn526>

## See Also

[identifySignatures](#)

[rss](#) and [evar](#) functions of the [NMF](#) package.

## Examples

```
data("sca_mm", package = "SomaticSignatures")

nSigs = 2:8
stat = assessNumberSignatures(sca_mm, nSigs, nReplicates = 3)

plotNumberSignatures(stat)
```

---

readMutect

*readMutect*

---

## Description

Import 'mutect' calls.

## Usage

```
readMutect(file, columns, strip = FALSE)
```

## Arguments

file	Location of the mutect tsv files [character, required]
columns	Names of columns to import from the file [character vector, optional, default: missing]. If missing, all columns will be imported.
strip	Should additional columns be imported? [logical, default: FALSE]. If TRUE, return only the bare 'VRanges' object.

## Details

The 'readMutect' functions imports the mutational calls of a '\*.tsv' file returned by the 'mutect' caller to a 'VRanges' object. For a description of the information of the columns, please refer to the mutect documentation.

## Value

A 'VRanges' object, with each row corresponding to one variant in the original file.

## References

Cibulskis, Kristian, Michael S. Lawrence, Scott L. Carter, Andrey Sivachenko, David Jaffe, Carrie Sougnez, Stacey Gabriel, Matthew Meyerson, Eric S. Lander, and Gad Getz. "Sensitive Detection of Somatic Point Mutations in Impure and Heterogeneous Cancer Samples." *Nature Biotechnology* advance online publication (February 10, 2013). doi:10.1038/nbt.2514.

[http://www.broadinstitute.org/cancer/cga/mutect\\_run](http://www.broadinstitute.org/cancer/cga/mutect_run)

## Examples

```
mutect_path = system.file("examples", "mutect.tsv", package = "SomaticSignatures")
vr1 = readMutect(mutect_path)
vr2 = readMutect(mutect_path, strip = TRUE)
```

---

sca-data

*SomaticCancerAlterations Results*

---

## Description

Motif matrix and 5 estimated signatures (NMF) from the somatic variant calls in the 'SomaticCancerAlterations' package. For details, see the vignette of the 'SomaticSignatures' package.

## See Also

SomaticCancerAlterations package

## Examples

```
data(sca_motifs_tiny, package = "SomaticSignatures")
data(sca_mm, package = "SomaticSignatures")
data(sca_sigs, package = "SomaticSignatures")
```

---

signature-plots

*Plot Mutational Signatures*

---

## Description

Visualize estimated signatures, sample contribution, and mutational spectra.



**Usage**

```
plotObservedSpectrum(s, colorby = c("sample", "alteration"))
plotFittedSpectrum(s, colorby = c("sample", "alteration"))

plotMutationSpectrum(vr, group, colorby = c("sample", "alteration"), normalize = TRUE)

plotSignatureMap(s)
plotSignatures(s, normalize = FALSE, percent = FALSE)

plotSampleMap(s)
plotSamples(s, normalize = FALSE, percent = FALSE)
```

**Arguments**

s	MutationalSignatures object [required]
vr	VRanges object
colorby	Which variable to use for the coloring in the spectra representation.
normalize	Plot relative contributions (TRUE) instead of absolute (FALSE) ones.
percent	Display the results as fraction (FALSE) or percent (TRUE)
.	
group	Charactering string that represents the variable name used for grouping.

**Details**

With the plotting function, the obtained signatures and their occurrence in the samples can be visualized either as a heatmap ('plotSignatureMap', 'plotSampleMap') or a barchart ('plotSignature', 'plotSamples').

Since the plotting is based on the 'ggplot2' framework, all properties of the plots can be fully controlled by the user after generating the plots. Please see the examples for some customizations and the 'ggplot2' documentation for the entire set of options.

**Value**

A 'ggplot' object, whose properties can further be changed

**See Also**

See the 'ggplot2' package for customizing the plots.

**Examples**

```
data("sca_sigs", package = "SomaticSignatures")

plotSamples(sigs_nmf)

plotSignatures(sigs_nmf, normalize = TRUE)
```

```

## customize the plots ##
p = plotSamples(sigs_nmf)

library(ggplot2)
## (re)move the legend
p = p + theme(legend.position = "none")
## change the axis labels
p = p + xlab("Studies")
## add a title
p = p + ggtitle("Somatic Signatures in TGCA WES Data")
## change the color scale
p = p + scale_fill_brewer(palette = "Blues")
## decrease the size of x-axis labels
p = p + theme(axis.text.x = element_text(size = 9))

p

```

---

signatures21-data      *21 Signatures*

---

### Description

Published signatures, taken from <ftp://ftp.sanger.ac.uk/pub/cancer/AlexandrovEtAl/signatures.txt>

### References

Alexandrov, Ludmil B., Serena Nik-Zainal, David C. Wedge, Samuel A. J. R. Aparicio, Sam Behjati, Andrew V. Biankin, Graham R. Bignell, et al. Signatures of Mutational Processes in Human Cancer. *Nature* 500, no. 7463 (August 22, 2013): 415-21. doi:10.1038/nature12477.

### Examples

```

data(signatures21, package = "SomaticSignatures")

head(signatures21)

```

---

SomaticSignatures      *SomaticSignatures package*

---

### Description

Identifying somatic signatures of single nucleotide variants. This package provides a infrastructure related to the methodology described in Nik-Zainal (2012, Cell), with flexibility in the matrix decomposition algorithms.

## Details

The 'SomaticSignatures' package offers the framework for identifying mutational signatures of single nucleotide variants (SNVs) from high-throughput experiments. In the concept of mutational signatures, a base change resulting from an SNV is regarded in term of motifs which embeds the variant in the context of the surrounding genomic sequence. Based on the frequency of such motifs across samples, mutational signatures and their occurrence in the samples can be estimated. An introduction into the methodology and a use case are illustrated in the vignette of this package.

## Author(s)

Julian Gehring, Bernd Fischer, Michael Lawrence, Wolfgang Huber: SomaticSignatures: Inferring Mutational Signatures from Single Nucleotide Variants. 2015, bioRxiv preprint, <http://dx.doi.org/10.1101/010686>

Maintainer: Julian Gehring, EMBL Heidelberg <julian.gehring@embl.de>

## References

Nik-Zainal, Serena, Ludmil B. Alexandrov, David C. Wedge, Peter Van Loo, Christopher D. Greenman, Keiran Raine, David Jones, et al. "Mutational Processes Molding the Genomes of 21 Breast Cancers." *Cell* 149, no. 5 (May 25, 2012): 979-993. doi:10.1016/j.cell.2012.04.024.

Alexandrov, Ludmil B., Serena Nik-Zainal, David C. Wedge, Samuel A. J. R. Aparicio, Sam Behjati, Andrew V. Biankin, Graham R. Bignell, et al. "Signatures of Mutational Processes in Human Cancer." *Nature* 500, no. 7463 (August 22, 2013): 415-421. doi:10.1038/nature12477.

Gaujoux, Renaud, and Cathal Seoighe. "A Flexible R Package for Nonnegative Matrix Factorization." *BMC Bioinformatics* 11, no. 1 (July 2, 2010): 367. doi:10.1186/1471-2105-11-367.

Stacklies, Wolfram, Henning Redestig, Matthias Scholz, Dirk Walther, and Joachim Selbig. "pcaMethods - A Bioconductor Package Providing PCA Methods for Incomplete Data." *Bioinformatics* 23, no. 9 (May 1, 2007): 1164-1167. doi:10.1093/bioinformatics/btm069.

## Examples

```
vignette(package = "SomaticSignatures")
```

---

variants-utils

*Utility functions*

---

## Description

Utility functions

## Usage

```
dfConvertColumns(x, from = "character", to = "factor")
```

**Arguments**

x	A 'data.frame' to convert [required].
from	The class of the columns to be converted [default: 'character'].
to	The class of the columns to be converted to [default: 'factor'].

**Details**

The 'dfConvertColumns' converts all columns of a data frame with class 'from' to the class 'to'.

**Value**

A 'data.frame' object.

# Index

- \* **IO**
  - readMutect, 15
- \* **datasets**
  - kmers-data, 7
  - sca-data, 16
  - signatures21-data, 18
- \* **manip**
  - GRanges-converters, 4
  - mutationContext, 12
- \* **package**
  - SomaticSignatures, 18
- \* **utilities**
  - GRanges-converters, 4
- assessNumberSignatures
  - (numberSignatures), 13
- cluster-spectrum, 2
- clusterSpectrum (cluster-spectrum), 2
- decomposition-signatures, 3
- dfConvertColumns (variants-utils), 19
- dist, 3
- evar, 15
- findSignatures (mutational-signatures), 10
- fitted (MutationalSignatures), 11
- fitted, MutationalSignatures-method
  - (MutationalSignatures), 11
- gcContent, 4
- GRanges-converters, 4
- hclust, 3
- hs-chrs, 5
- hsAllosomes (hs-chrs), 5
- hsAutosomes (hs-chrs), 5
- hsLinear (hs-chrs), 5
- hsToplevel (hs-chrs), 5
- identifySignatures, 12, 15
- identifySignatures
  - (mutational-signatures), 10
- k3we (kmers-data), 7
- k3wg (kmers-data), 7
- kmerFrequency, 6, 7
- kmers (kmers-data), 7
- kmers-data, 7
- mcols, 5
- motif-functions, 7
- motifMatrix, 10, 11
- motifMatrix (motif-functions), 7
- mutation-distribution, 8
- mutational-normalization, 9
- mutational-plots, 10
- mutational-signatures, 10
- MutationalSignatures, 11, 11
- MutationalSignatures-class
  - (MutationalSignatures), 11
- mutationContext, 11, 12
- mutationContextMutect, 11
- mutationContextMutect
  - (mutationContext), 12
- mutationDistance
  - (mutation-distribution), 8
- ncbi (GRanges-converters), 4
- NMF, 4, 15
- nmfDecomposition, 11
- nmfDecomposition
  - (decomposition-signatures), 3
- nmfSignatures
  - (decomposition-signatures), 3
- normalizeMotifs
  - (mutational-normalization), 9
- numberSignatures, 13
- observed (MutationalSignatures), 11

- observed, MutationalSignatures-method  
(MutationalSignatures), 11
- pcaDecomposition, 11
- pcaDecomposition  
(decomposition-signatures), 3
- pcaMethods, 4
- pcaSignatures  
(decomposition-signatures), 3
- plotFittedSpectrum (signature-plots), 16
- plotGrandLinear, 9
- plotMutationSpectrum (signature-plots),  
16
- plotNumberSignatures  
(numberSignatures), 13
- plotObservedSpectrum (signature-plots),  
16
- plotRainfall (mutation-distribution), 8
- plotSampleMap (signature-plots), 16
- plotSamples (signature-plots), 16
- plotSignatureMap (signature-plots), 16
- plotSignatures (signature-plots), 16
- plotVariantAbundance  
(mutational-plots), 10
- prcomp, 4
  
- readMutect, 13, 15
- rss, 15
  
- samples (MutationalSignatures), 11
- samples, MutationalSignatures-method  
(MutationalSignatures), 11
- sca-data, 16
- sca\_mm (sca-data), 16
- sca\_motifs\_tiny (sca-data), 16
- sca\_sigs (sca-data), 16
- seqchar (GRanges-converters), 4
- seqlevelsStyle, 5
- seqnames, 5
- show (MutationalSignatures), 11
- show, MutationalSignatures-method  
(MutationalSignatures), 11
- signature-plots, 16
- signatures (MutationalSignatures), 11
- signatures, MutationalSignatures-method  
(MutationalSignatures), 11
- signatures21 (signatures21-data), 18
- signatures21-data, 18
- sigs\_nmf (sca-data), 16
- sigs\_pca (sca-data), 16
- SomaticSignatures, 18
- SomaticSignatures-package  
(SomaticSignatures), 18
- ucsc (GRanges-converters), 4
  
- variants-utils, 19