

# Package ‘SingleMoleculeFootprinting’

December 24, 2024

**Title** Analysis tools for Single Molecule Footprinting (SMF) data

**Version** 2.0.0

**Depends** R (>= 4.4.1)

**Imports** BiocGenerics, Biostrings, BSgenome, dplyr, GenomeInfoDb, GenomicRanges, ggpointdensity, ggplot2, ggrepel, grDevices, IRanges, Matrix, parallel, patchwork, plyr, plyranges, QuasR, RColorBrewer, rlang, S4Vectors, stats, stringr, tibble, tidyr, tidyverse, viridis

**Description** SingleMoleculeFootprinting provides functions to analyze Single Molecule Footprinting (SMF) data. Following the workflow exemplified in its vignette, the user will be able to perform basic data analysis of SMF data with minimal coding effort. Starting from an aligned bam file, we show how to perform quality controls over sequencing libraries, extract methylation information at the single molecule level accounting for the two possible kind of SMF experiments (single enzyme or double enzyme), classify single molecules based on their patterns of molecular occupancy, plot SMF information at a given genomic location.

**biocViews** DNAMethylation, Coverage, NucleosomePositioning, DataRepresentation, Epigenetics, MethylSeq, QualityControl, Sequencing

**BugReports** <https://github.com/Krebslabrep/SingleMoleculeFootprinting/issues>

**License** GPL-3

**Encoding** UTF-8

**Suggests** BSgenome.Mmusculus.UCSC.mm10, devtools, ExperimentHub, knitr, qs, rmarkdown, readr, SingleMoleculeFootprintingData, testthat (>= 3.0.0)

**VignetteBuilder** knitr

**RoxygenNote** 7.3.2

**Config/testthat/edition** 3

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

**git\_branch** RELEASE\_3\_20

**git\_last\_commit** 4d5ea07

**git\_last\_commit\_date** 2024-10-29

**Repository** Bioconductor 3.20

**Date/Publication** 2024-12-23

**Author** Guido Barzaghi [aut, cre] (<<https://orcid.org/0000-0001-6066-3920>>),

Arnaud Krebs [aut] (<<https://orcid.org/0000-0001-7999-6127>>),

Mike Smith [ctb] (<<https://orcid.org/0000-0002-7800-3848>>)

**Maintainer** Guido Barzaghi <guido.barzaghi@embl.de>

## Contents

Arrange_TFBSs_clusters . . . . .	3
BaitCapture . . . . .	4
BinMethylation . . . . .	5
CallContextMethylation . . . . .	5
cbind.fill.Matrix . . . . .	7
CollapseStrands . . . . .	8
CollapseStrandsSM . . . . .	8
CollectCompositeData . . . . .	9
colMeans_drop0 . . . . .	10
CompositeMethylationCorrelation . . . . .	10
CompositePlot . . . . .	12
ConversionRate . . . . .	12
CoverageFilter . . . . .	13
Create_MethylationCallingWindows . . . . .	13
DetectExperimentType . . . . .	14
FilterByConversionRate . . . . .	15
FilterContextCytosines . . . . .	15
filter_reads_from_MethGR . . . . .	16
full.join.granges . . . . .	16
GetQuasRprj . . . . .	17
GetSingleMolMethMat . . . . .	17
GRanges_to_DF . . . . .	18
HierarchicalClustering . . . . .	18
LowCoverageMethRate_RMSE . . . . .	19
MaskSNPs . . . . .	19
MethSM.to.MethGR . . . . .	20
panel.cor . . . . .	20
panel.hist . . . . .	21
panel.jet . . . . .	21
PlotAvgSMF . . . . .	21
PlotSingleMoleculeStack . . . . .	22
PlotSingleSiteSMF . . . . .	23
PlotSM . . . . .	24
Plot_LowCoverageMethRate . . . . .	25
Plot_LowCoverageMethRate_RMSE . . . . .	25
rbind.fill.Matrix . . . . .	26
rowMeans_drop0 . . . . .	26
SingleTFStateQuantificationPlot . . . . .	27
SingleTFStates . . . . .	27
SortReads . . . . .	28
SortReadsBySingleTF . . . . .	28
SortReadsBySingleTF_MultiSiteWrapper . . . . .	29

<i>Arrange_TFBSs_clusters</i>	3
SortReadsByTFCluster . . . . .	31
SortReadsByTFCluster_MultiSiteWrapper . . . . .	32
StateQuantification . . . . .	34
StateQuantificationBySingleTF . . . . .	35
StateQuantificationByTFPair . . . . .	36
StateQuantificationPlot . . . . .	36
SubsetGRangesForSamples . . . . .	37
TFPairStateQuantificationPlot . . . . .	37
TFPairStates . . . . .	38
<b>Index</b>	<b>39</b>

Arrange\_TFBSs\_clusters  
*Convenience function to arrange a list of given TFBSs into clusters*

### Description

For each TFBS, the genomic neighborhood defined by `max_cluster_width` will be scanned for adjacent TFBSs. The hits will be filtered for `min_intersite_distance` where, in case of overlapping TFBSs, the second TFBS will be arbitrarily dropped. These TFBSs plus the central "anchoring" one will define a TFBS cluster. This approach implies that the same TFBS can be employed to design multiple clusters in a sliding-window fashion.

### Usage

```
Arrange_TFBSs_clusters(
  TFBSs,
  max_intersite_distance = 75,
  min_intersite_distance = 15,
  max_cluster_size = 6,
  max_cluster_width = 300,
  add.single.TFs = TRUE
)
```

### Arguments

TFBSs	GRanges object of TFBSs
max_intersite_distance	maximum allowed distance in base pairs between two TFBS centers for them to be considered part of the same cluster. Defaults to 75.
min_intersite_distance	minimum allowed distance in base pairs between two TFBS centers for them not to be discarded as overlapping. This parameter should be set according to the width of the bins used for later sorting. Defaults to 15.
max_cluster_size	maximum number of TFBSs to be contained in any given cluster. Defaults to 6
max_cluster_width	maximum width of TFBS clusters in bps. Defaults to 300
add.single.TFs	Whether to add the TFs not used to create TFBS.clusters to the list for sorting. Defaults to TRUE

**Value**

list with two elements: ClusterCoordinates (GRanges object of clusters coordinates) and ClusterComposition (GRangesList of sites for each cluster)

**Examples**

```
KLF4s = qs::qread(system.file("extdata", "KLF4_chr19.qs", package="SingleMoleculeFootprinting"))
Arrange_TFBSs_clusters(KLF4s)
```

---

BaitCapture	<i>Bait capture efficiency</i>
-------------	--------------------------------

---

**Description**

check bait capture efficiency. Expected to be ~70

**Usage**

```
BaitCapture(sampleFile, genome, baits, clObj = NULL)
```

**Arguments**

sampleFile	QuasR sample sheet
genome	BS genome
baits	GRanges obj of bait coordinates. We provide an example through SingleMoleculeFootprintingData::EnrichmentRegions_mm10.rds()
clObj	cluster object to employ for parallel processing created using the parallel::makeCluster function. Defaults to NULL

**Value**

bait capture efficiency

**Examples**

```
sampleFile = paste0(tempdir(), "/NRF1Pair_Qinput.txt")

if(file.exists(sampleFile)){
  library(BSgenome.Mmusculus.UCSC.mm10)
  BaitRegions = SingleMoleculeFootprintingData::EnrichmentRegions_mm10.rds()
  BaitCapture(sampleFile = sampleFile, genome = BSgenome.Mmusculus.UCSC.mm10, baits = BaitRegions)
}
```

---

BinMethylation	<i>Summarize methylation inside sorting bins</i>
----------------	--

---

**Description**

Summarize methylation inside sorting bins

**Usage**

```
BinMethylation(MethSM, Bin)
```

**Arguments**

MethSM	Single molecule matrix
Bin	IRanges object with absolute coordinates for single sorting bin.

**Value**

Reads covering bin with their summarized methylation status

**Examples**

```
library(IRanges)
library(GenomicRanges)

MethSM = qs::qread(system.file("extdata", "Methylation_4.qs",
package="SingleMoleculeFootprinting"))[[2]]$SMF_MM_TKO_DE_

TFBSs = qs::qread(system.file("extdata", "TFBSs_1.qs",
package="SingleMoleculeFootprinting"))

motif_center_1 = start(IRanges::resize(TFBSs[1], 1, "center"))
motif_center_2 = start(IRanges::resize(TFBSs[2], 1, "center"))
SortingBins = c(
GRanges("chr6", IRanges(motif_center_1-35, motif_center_1-25)),
GRanges("chr6", IRanges(motif_center_1-7, motif_center_1+7)),
GRanges("chr6", IRanges(motif_center_2-7, motif_center_2+7)),
GRanges("chr6", IRanges(motif_center_2+25, motif_center_2+35))
)

binMethylationValues = BinMethylation(MethSM = MethSM, Bin = SortingBins[1])
```

---

CallContextMethylation

*Call Context Methylation*

---

**Description**

Can deal with multiple samples

**Usage**

```
CallContextMethylation(
  sampleFile,
  samples,
  genome,
  RegionOfInterest,
  coverage = 20,
  ConvRate.thr = NULL,
  returnSM = TRUE,
  clObj = NULL,
  verbose = FALSE
)
```

**Arguments**

sampleFile	QuasR pointer file
samples	vector of unique sample names corresponding to the SampleName field from the sampleFile
genome	BSgenome
RegionOfInterest	GenimocRange representing the genomic region of interest
coverage	coverage threshold as integer for least number of reads to cover a cytosine for it to be carried over in the analysis. Defaults to 20.
ConvRate.thr	Convesion rate threshold. Double between 0 and 1, defaults to NULL. To skip this filtering step, set to NULL. For more information, check out the details section.
returnSM	whether to return the single molecule matrix, defaults to TRUE
clObj	cluster object for parallel processing of multiple samples. For now only used by qMeth call for bulk methylation. Should be the output of a parallel::makeCluster() call
verbose	whether to print out messages while executing. Defaults to FALSE

**Details**

The ConvRate.thr argument should be used with care as it could create biases (e.g. when only one C out of context is present) while generally only marginally cleaning up the data.

**Value**

List with two Granges objects: average methylation call (GRanges) and single molecule methylation call (matrix)

**Examples**

```
sampleFile = NULL
if(!is.null(sampleFile)){
Methylation <- CallContextMethylation(
  sampleFile = sampleFile,
  samples = samples,
  genome = BSgenome.Mmusculus.UCSC.mm10,
  RegionOfInterest = RegionOfInterest,
```

```
coverage = 20,  
returnSM = TRUE,  
ConvRate.thr = NULL,  
clobj = NULL  
)  
}
```

---

cbind.fill.Matrix	<i>Implementation performing a similar operation of rbind.fill.Matrix but for columns</i>
-------------------	---

---

## Description

Implementation performing a similar operation of rbind.fill.Matrix but for columns

## Usage

```
cbind.fill.Matrix(x, y)
```

## Arguments

x	sparse matrix constructed using the function Matrix::sparseMatrix. Should have Dimnames and dims (e.g. when indexing drop=FALSE)
y	sparse matrix constructed using the function Matrix::sparseMatrix. Should have Dimnames and dims (e.g. when indexing drop=FALSE)

## Details

N.b. only possible fill at the moment is 0

## Examples

```
Methylation = qs::qread(system.file("extdata", "Methylation_3.qs",  
package="SingleMoleculeFootprinting"))  
MethSM_1 = Methylation[[2]][[1]]  
Methylation = qs::qread(system.file("extdata", "Methylation_4.qs",  
package="SingleMoleculeFootprinting"))  
MethSM_2 = Methylation[[2]][[1]]  
cbind.fill.Matrix(MethSM_1, MethSM_2)
```

---

CollapseStrands	<i>Collapse strands</i>
-----------------	-------------------------

---

**Description**

Collapse strands

**Usage**

```
CollapseStrands(MethGR, context)
```

**Arguments**

MethGR	Granges obj of average methylation
context	"GC" or "HCG". Broad because indicates just the directionality of collapse.

**Value**

MethGR with collapsed strands (everything turned to - strand)

**Examples**

```
# CollapseStrands(MethGR, "GC")
```

---

CollapseStrandsSM	<i>Collapse strands in single molecule matrix</i>
-------------------	---

---

**Description**

The idea here is that (regardless of context) if a C is on the - strand, calling getSeq on that coord (N.b. unstranded, that's the important bit) will give a "G", a "C" if it's a + strand.

**Usage**

```
CollapseStrandsSM(MethSM, context, genome, chr)
```

**Arguments**

MethSM	Single molecule matrix
context	"GC" or "CG". Broad because indicates just the directionality of collapse.
genome	BSgenome
chr	Chromosome, MethSM doesn't carry this info

**Value**

Strand collapsed MethSM



**Examples**

```
# CollapseStrandsSM(MethSM, "GC", BSgenome.Mmusculus.UCSC.mm10, "chr19")
```

---

CollectCompositeData *Collect bulk SMF data for later composite plotting*

---

**Description**

Collect bulk SMF data for later composite plotting

**Usage**

```
CollectCompositeData(  
  sampleFile,  
  samples,  
  genome,  
  TFBSs,  
  window,  
  coverage = 20,  
  ConvRate.thr = NULL,  
  cores = 1  
)
```

**Arguments**

sampleFile	QuasR sampleFile
samples	vector of unique sample names corresponding to the SampleName field from the sampleFile
genome	BSgenome
TFBSs	GRanges object of TF binding sites to collect info for. We recommend employing 50 to 200 TFBSs.
window	window size to collect methylation information for
coverage	coverage threshold as integer for least number of reads to cover a cytosine for it to be carried over in the analysis. Defaults to 20.
ConvRate.thr	Convesion rate threshold. Double between 0 and 1, defaults to NULL For more information, check out the details section
cores	number of cores to use

**Value**

data.frame of bulk SMF info ready for plotting

**Examples**

```

sampleFile = NULL
if(!is.null(sampleFile)){
  CollectCompositeData(
    sampleFile = sampleFile,
    samples = samples,
    genome = BSgenome.Mmusculus.UCSC.mm10,
    TFBSs = TopMotifs,
    window = 1000,
    coverage = 20,
    ConvRate.thr = NULL,
    cores = 16
  ) -> CompositeData
}

```

---

colMeans_drop0	<i>Calculate colMeans after dropping zeros</i>
----------------	--

---

**Description**

Calculate colMeans after dropping zeros

**Usage**

```
colMeans_drop0(MethSM)
```

**Arguments**

MethSM            one single molecule sparse matrix

**Value**

colMeans (N.b. this is +1 based)

---

CompositeMethylationCorrelation	<i>Composite Methylation Rate</i>
---------------------------------	-----------------------------------

---

**Description**

Monitor methylation rate distribution in a low coverage samples as compared to a high coverage "reference" one. It bins cytosines with similar methylation rates (as observed in the HighCoverage sample) into bins. A single methylation rate value is computed for each bin



---

CompositePlot                      *Plot composite SMF data*

---

### Description

Will use geom\_point with  $\leq 5000$  points, geom\_hex otherwise

### Usage

```
CompositePlot(CompositeData, span = 0.1, TF)
```

### Arguments

CompositeData    the output of the CollectCompositeData function  
span                      the span parameter to pass to geom\_smooth  
TF                      string of TF name to use for plot title

### Examples

```
# CompositePlot(CompositeData = CompositeData, span = 0.1, TF = "Rest")
```

---

ConversionRate                      *Conversion rate*

---

### Description

calculate sequencing library conversion rate on a chromosome of choice

### Usage

```
ConversionRate(sampleFile, genome, chr = 19, cores = 1)
```

### Arguments

sampleFile            QuasR sample sheet  
genome                      BS genome  
chr                      chromosome to calculate conversion rate on (default: 19)  
cores                      number of cores for parallel processing. Defaults to 1

### Examples

```
# ConversionRate(sampleFile = sampleFile,  
# genome = BSgenome.Mmusculus.UCSC.mm10, chr = 19, cores = 1)
```

---

CoverageFilter	<i>Filter Cs for coverage</i>
----------------	-------------------------------

---

**Description**

Filter Cs for coverage

**Usage**

```
CoverageFilter(MethGR, thr)
```

**Arguments**

MethGR	Granges obj of average methylation
thr	coverage threshold

**Value**

filtered MethGR

---

Create\_MethylationCallingWindows

*Create methylation calling windows to call context methylation in one run for clusters lying proximally to each other*

---

**Description**

Relevant for genome-wide analyses

**Usage**

```
Create_MethylationCallingWindows(
  RegionsOfInterest,
  max_intercluster_distance = 1e+05,
  max_window_width = 5e+06,
  min_cluster_width = 600,
  genomic.seqlengths,
  fix.window.size = FALSE,
  max.window.size = 500
)
```

**Arguments**

RegionsOfInterest

TFBS cluster coordinates analogous to ClusterCoordinates object returned by Arrange\_TFBSs\_clusters function

max\_intercluster\_distance

maximum distance between two consecutive TFBS clusters for them to be grouped in the same window

`max_window_width`  
 upper limit to window width. This value should be adjusted according to the user's system as it determines the amount of memory used in the later context methylation call

`min_cluster_width`  
 lower limit to window width. Corresponds to the scenario when a window contains a single TFBS cluster.

`genomic.seqlenghts`  
 used to fix the windows spanning over chromosome edges. To be fetched by `GenomeInfoDb::seqlenghts()` or equivalent.

`fix.window.size`  
 Defaults to `FALSE`. When `TRUE`, overrides arguments `max_intercluster_distance` and `max_window_width` and produces windows containing a fixed number of `TFBS_clusters`.

`max.window.size`  
 Max number of `TFBS_clusters` per window. Used only when `fix.window.size` is `TRUE`. N.b.: window size could be slightly higher than passed value if `RegionsOfInterest` overlap

**Value**

`GRanges` object of window coordinates to be used for more efficient calls of `CallContextMethylation`

**Examples**

```
KLF4s = qs::qread(system.file("extdata", "KLF4_chr19.qs", package="SingleMoleculeFootprinting"))
Create_MethylationCallingWindows(RegionsOfInterest = KLF4s)
```

---

DetectExperimentType *Detect type of experiment*

---

**Description**

Detect type of experiment

**Usage**

```
DetectExperimentType(Samples)
```

**Arguments**

`Samples`                      `SampleNames` field from QuasR `sampleFile`

**Examples**

```
CacheDir = ExperimentHub::getExperimentHubOption(arg = "CACHE")
sampleFile = paste0(CacheDir, "/NRF1Pair_sampleFile.txt")
samples = suppressMessages(unique(readr::read_delim(sampleFile, delim = "\t")[[2]]))
DetectExperimentType(samples)
```

---

 FilterByConversionRate

*Calculate reads conversion rate*


---

**Description**

Calculate reads conversion rate

**Usage**

```
FilterByConversionRate(MethSM, chr, genome, thr)
```

**Arguments**

MethSM	as comes out of the func GetSingleMolMethMat
chr	Chromosome, MethSM doesn't carry this info
genome	BSgenome
thr	Double between 0 and 1. Threshold below which to filter reads.

**Value**

Filtered MethSM

**Examples**

```
library(BSgenome.Mmusculus.UCSC.mm10)
MethSM = qs::qread(system.file("extdata", "Methylation_3.qs",
package="SingleMoleculeFootprinting"))[[2]]$SMF_MM_TKO_DE_
FilterByConversionRate(MethSM, chr = "chr19",
genome = BSgenome.Mmusculus.UCSC.mm10, thr = 0.8)
```

---

 FilterContextCytosines

*Filter Cytosines in context*


---

**Description**

Filter Cytosines in context

**Usage**

```
FilterContextCytosines(MethGR, genome, context)
```

**Arguments**

MethGR	Granges obj of average methylation
genome	BSgenome
context	Context of interest (e.g. "GC", "CG", ..)

**Value**

filtered Granges obj

**Examples**

```
library(BSgenome.Mmusculus.UCSC.mm10)
MethGR = qs::qread(system.file("extdata", "Methylation_3.qs",
package="SingleMoleculeFootprinting"))[[1]]

FilterContextCytosines(MethGR, BSgenome.Mmusculus.UCSC.mm10, "NGCNN")
```

---

filter\_reads\_from\_MethGR

*Recalculate \*\_T and \*\_M values in MethGR object after filtering reads  
e.g. for conversion rate*

---

**Description**

Recalculate \*\_T and \*\_M values in MethGR object after filtering reads e.g. for conversion rate

**Usage**

```
filter_reads_from_MethGR(MethGR, MethSM, MethSM_filtered, sampleIndex)
```

**Arguments**

MethGR	GRanges object of methylation call
MethSM	Single Molecule methylation matrix
MethSM_filtered	Single Molecule methylation matrix after filtering reads
sampleIndex	index for sample to treat. It serves as a correspondence between the index of the SM matrix and the order samples appear in the elementMetadata() columns

**Value**

MethGR with recalculated counts

---

full.join.granges	<i>Utility function to perform the dplyr full_join operation on GRanges object</i>
-------------------	--

---

**Description**

Utility function to perform the dplyr full\_join operation on GRanges object

**Usage**

```
full.join.granges(MethGR1, MethGR2)
```



**Arguments**

MethGR1	Methylation GRanges as output by the CallContextMethylation() function
MethGR2	Methylation GRanges as output by the CallContextMethylation() function

---

GetQuasRprj	<i>Get QuasRprj</i>
-------------	---------------------

---

**Description**

Get QuasRprj

**Usage**

```
GetQuasRprj(sampleFile, genome)
```

**Arguments**

sampleFile	QuasR pointer file
genome	BSgenome

**Examples**

```
library(BSgenome.Mmusculus.UCSC.mm10)
CacheDir <- ExperimentHub::getExperimentHubOption(arg = "CACHE")
sampleFile = paste0(CacheDir, "/NRF1Pair_sampleFile.txt")
QuasRprj = GetQuasRprj(sampleFile, BSgenome.Mmusculus.UCSC.mm10)
```

---

GetSingleMolMethMat	<i>Get Single Molecule methylation matrix</i>
---------------------	---

---

**Description**

Used internally as the first step in CallContextMethylation

**Usage**

```
GetSingleMolMethMat(QuasRprj, range, sample)
```

**Arguments**

QuasRprj	QuasR project object as returned by calling the QuasR function qAlign on previously aligned data
range	GenimocRange representing the genomic region of interest
sample	One of the sample names as reported in the SampleName field of the QuasR sampleFile provided to qAlign. N.b. all the files with the passed sample name will be used to call methylation

**Value**

List of single molecule methylation matrixes (all Cytosines), one per sample

**Examples**

```
library(BSgenome.Mmusculus.UCSC.mm10)
library(IRanges)
library(GenomicRanges)

CacheDir <- ExperimentHub::getExperimentHubOption(arg = "CACHE")
sampleFile = paste0(CacheDir, "/NRF1Pair_sampleFile.txt")
sample = suppressMessages(readr::read_delim(sampleFile, delim = "\t")[[2]])
QuasRprj = GetQuasRprj(sampleFile, BSgenome.Mmusculus.UCSC.mm10)
range = GRanges("chr6", IRanges(88106000, 88106500))

GetSingleMolMethMat(QuasRprj, range, sample)
```

---

GRanges_to_DF	<i>Manipulate GRanges into data.frame</i>
---------------	---

---

**Description**

Inner utility for LowCoverageMethRateDistribution

**Usage**

```
GRanges_to_DF(GRanges_obj)
```

**Arguments**

GRanges\_obj      GRanges object as returned by CallContextMethylation function

---

HierarchicalClustering	<i>Perform Hierarchical clustering on single reads</i>
------------------------	--

---

**Description**

Perform Hierarchical clustering on single reads

**Usage**

```
HierarchicalClustering(MethSM)
```

**Arguments**

MethSM            Single molecule methylation matrix

---

 LowCoverageMethRate\_RMSE

*Low Coverage Methylation Rate RMSE*


---

**Description**

Calculate Root mean squared error (RMSE) of methylation rate distribution estimates for low coverage samples

**Usage**

```
LowCoverageMethRate_RMSE(BinnedMethRate)
```

**Arguments**

BinnedMethRate data.frame as returned by GRanges\_to\_DF function.

---

 MaskSNPs

*Utility function to remove cytosines whose MTase target genomic context is affected by SNPs*


---

**Description**

Utility function to remove cytosines whose MTase target genomic context is affected by SNPs

**Usage**

```
MaskSNPs(
  Methylation,
  CytosinesToMask,
  MaskSMmat = FALSE,
  SampleStringMatch = list(Cast = "_CTKO", Spret = "_STKO"),
  Experiment
)
```

**Arguments**

Methylation as output by the CallContextMethylation() function

CytosinesToMask

GRanges specifying the coordinate of the cytosines to discard.

MaskSMmat whether the parameter Methylation includes single molecule matrixes

SampleStringMatch

list of per-sample string matches that are used to uniquely identify the relevant column for each species in the Methylation object. Defaults to list(Cast = "\_CTKO", Spret = "\_STKO")

Experiment as detected by the DetectExperimentType() function. Should be either "DE" or "NO"

**Examples**

```
Methylation = qs::qread(system.file("extdata", "Methylation_2.qs",
package="SingleMoleculeFootprinting"))
CytosinesToMask = qs::qread(system.file("extdata", "cytosines_to_mask.qs",
package="SingleMoleculeFootprinting"))

MaskSNPs(Methylation = Methylation, CytosinesToMask = CytosinesToMask, MaskSMmat = FALSE,
SampleStringMatch = list(Cast = "_CTKO", Spret = "_STKO"), Experiment = "DE") -> Methylation_masked
```

---

MethSM.to.MethGR	<i>Compute MethGR from MethSM</i>
------------------	-----------------------------------

---

**Description**

Compute MethGR from MethSM

**Usage**

```
MethSM.to.MethGR(MethSM, chromosome)
```

**Arguments**

MethSM	internal CallContextMethylation
chromosome	string

---

panel.cor	<i>Utility for HighCoverage_MethRate_SampleCorrelation</i>
-----------	--

---

**Description**

Utility for HighCoverage\_MethRate\_SampleCorrelation

**Usage**

```
panel.cor(x, y, digits = 2, prefix = "", cex.cor)
```

**Arguments**

x	x variable
y	y variable
digits	number of digits
prefix	string
cex.cor	graphical param

---

panel.hist	<i>Utility for HighCoverage_MethRate_SampleCorrelation</i>
------------	--

---

**Description**

Utility for HighCoverage\_MethRate\_SampleCorrelation

**Usage**

```
panel.hist(x, ...)
```

**Arguments**

x	data for hist
...	data for hist

---

panel.jet	<i>Utility for HighCoverage_MethRate_SampleCorrelation</i>
-----------	--

---

**Description**

Utility for HighCoverage\_MethRate\_SampleCorrelation

**Usage**

```
panel.jet(...)
```

**Arguments**

...	data for lower pairs panel
-----	----------------------------

---

PlotAvgSMF	<i>Plot average methylation</i>
------------	---------------------------------

---

**Description**

Plot average methylation

**Usage**

```
PlotAvgSMF(
  MethGR,
  MethSM = NULL,
  RegionOfInterest,
  SortedReads = NULL,
  ShowContext = FALSE,
  TFBSs = NULL,
  SNPs = NULL,
  SortingBins = NULL
)
```

**Arguments**

MethGR	Average methylation GRanges obj
MethSM	Single molecule matrix(es)
RegionOfInterest	GRanges interval to plot
SortedReads	List of sorted reads, needs to be passed along with the parameter MethSM. If both are passed, only counts relevant to sorting will be plotted
ShowContext	TRUE or FALSE (default). Causes the genomic context of the plotted cytosines to be displayed as the dot shape
TFBSs	GRanges object of transcription factor binding sites to include in the plot. Assumed to be already subset. Also assumed that the tf names are under the column "TF"
SNPs	GRanges object of SNPs to visualize. Assumed to be already subset. Assumed to have the reference and alternative sequences respectively under the columns "R" and "A"
SortingBins	GRanges object of sorting bins (absolute) coordinate to visualize

**Examples**

```
library(GenomicRanges)

RegionOfInterest = GRanges("chr12", IRanges(20464551, 20465050))
Methylation = qs::qread(system.file("extdata", "Methylation_3.qs",
package="SingleMoleculeFootprinting"))
TFBSs = qs::qread(system.file("extdata", "TFBSs_3.qs", package="SingleMoleculeFootprinting"))

PlotAvgSMF(MethGR = Methylation[[1]], RegionOfInterest = RegionOfInterest, TFBSs = TFBSs)
```

---

PlotSingleMoleculeStack

*Plot single molecule stack*

---

**Description**

Plot single molecule stack

**Usage**

```
PlotSingleMoleculeStack(MethSM, RegionOfInterest)
```

**Arguments**

MethSM	Single molecule methylation matrix
RegionOfInterest	GRanges interval to plot

**Examples**

```
library(GenomicRanges)

RegionOfInterest = GRanges("chr12", IRanges(20464551, 20465050))
Methylation = qs::qread(system.file("extdata", "Methylation_3.qs",
package="SingleMoleculeFootprinting"))

PlotSingleMoleculeStack(MethSM = Methylation[[2]], RegionOfInterest = RegionOfInterest)
```

---

PlotSingleSiteSMF      *Plot SMF data at single site*

---

**Description**

Plot SMF data at single site

**Usage**

```
PlotSingleSiteSMF(
  Methylation,
  RegionOfInterest,
  ShowContext = FALSE,
  TFBSs = NULL,
  SNPs = NULL,
  SortingBins = NULL,
  SortedReads = NULL,
  sorting.strategy = "None"
)
```

**Arguments**

Methylation	Context methylation object as returned by CallContextMethylation function
RegionOfInterest	GRanges interval to plot
ShowContext	TRUE or FALSE (default). Causes the genomic context of the plotted cytosines to be displayed as the dot shape
TFBSs	GRanges object of transcription factor binding sites to include in the plot. Assumed to be already subset. Also assumed that the tf names are under the column "TF"
SNPs	GRanges object of SNPs to visualize. Assumed to be already subset. Assumed to have the reference and alternative sequences respectively under the columns "R" and "A"
SortingBins	GRanges object of sorting bins (absolute) coordinate to visualize
SortedReads	Defaults to NULL, in which case will plot unsorted reads. Sorted reads object as returned by SortReads function or "HC" to perform hierarchical clustering
sorting.strategy	One of "classical" (default), "custom", "hierarchical.clustering" or "None". Determines how to display reads. For details check documentation from PlotSM function.

**Examples**

```

library(GenomicRanges)

RegionOfInterest = GRanges("chr12", IRanges(20464551, 20465050))
Methylation = qs::qread(system.file("extdata", "Methylation_3.qs",
package="SingleMoleculeFootprinting"))
TFBSs = qs::qread(system.file("extdata", "TFBSs_3.qs", package="SingleMoleculeFootprinting"))
SortedReads = SortReadsBySingleTF(MethSM = Methylation[[2]], TFBS = TFBSs)

PlotSingleSiteSMF(Methylation = Methylation,
                  RegionOfInterest = RegionOfInterest,
                  SortedReads = SortedReads,
                  TFBSs = TFBSs)

```

PlotSM

*Wrapper for PlotSingleMoleculeStack function***Description**

adds the convenience of arranging reads before plotting

**Usage**

```

PlotSM(
  MethSM,
  RegionOfInterest,
  sorting.strategy = "classical",
  SortedReads = NULL
)

```

**Arguments**

MethSM	Single molecule methylation matrix
RegionOfInterest	GRanges interval to plot
sorting.strategy	One of "classical" (default), "custom", "hierarchical.clustering" or "None". Set to "classical" for classical one-TF/TF-pair sorting (as described in Sönmezer et al, MolCell, 2021). Should be passed along with argument SortedReads set to the Sorted reads object as returned by SortReads function. If set to "custom", SortedReads should be a list with one item per sample (corresponding to MethSM). If set to "hierarchical.clustering", the function will perform hierarchical clustering in place on a subset of reads. Useful to check for duplicated reads in amplicon sequencing experiments. If set to "None", it will plot unsorted reads. The argument sorting.strategy will always determine how to display reads with priority over the argument SortedReads
SortedReads	Defaults to NULL, in which case will plot unsorted reads. Sorted reads object as returned by SortReads function



**Examples**

```
library(GenomicRanges)

RegionOfInterest = GRanges("chr12", IRanges(20464551, 20465050))
Methylation = qs::qread(system.file("extdata", "Methylation_3.qs",
package="SingleMoleculeFootprinting"))
TFBSs = qs::qread(system.file("extdata", "TFBSs_3.qs",
package="SingleMoleculeFootprinting"))
SortedReads = SortReadsBySingleTF(MethSM = Methylation[[2]], TFBS = TFBSs)

PlotSM(MethSM = Methylation[[2]], RegionOfInterest = RegionOfInterest, SortedReads = SortedReads)
```

---

Plot\_LowCoverageMethRate

*Plot low coverage methylation rate*

---

**Description**

Inner utility for LowCoverageMethRateDistribution

**Usage**

Plot\_LowCoverageMethRate(Plotting\_DF)

**Arguments**

Plotting\_DF      data.frame as returned by GRanges\_to\_DF function.

---

Plot\_LowCoverageMethRate\_RMSE

*Plot Low Coverage Methylation Rate RMSE*

---

**Description**

Produce barplot of RMSE values calculated for methylation rate distribution estimates of low coverage samples

**Usage**

Plot\_LowCoverageMethRate\_RMSE(RMSE\_DF)

**Arguments**

RMSE\_DF              data.frame as returned by the LowCoverageMethRate\_RMSE function

---

<code>rbind.fill.Matrix</code>	<i>Implementation performing a similar operation of <code>plyr::rbind.fill.matrix</code> but for <code>sparseMatrix</code></i>
--------------------------------	--

---

**Description**

Implementation performing a similar operation of `plyr::rbind.fill.matrix` but for `sparseMatrix`

**Usage**

```
rbind.fill.Matrix(x, y)
```

**Arguments**

<code>x</code>	sparse matrix constructed using the function <code>Matrix::sparseMatrix</code> . Should have <code>Dimnames</code> and <code>dims</code> (e.g. when indexing <code>drop=FALSE</code> )
<code>y</code>	sparse matrix constructed using the function <code>Matrix::sparseMatrix</code> . Should have <code>Dimnames</code> and <code>dims</code> (e.g. when indexing <code>drop=FALSE</code> )

**Details**

N.b. only possible fill at the moment is 0

**Examples**

```
Methylation = qs::qread(system.file("extdata", "Methylation_3.qs",
package="SingleMoleculeFootprinting"))
MethSM_1 = Methylation[[2]][[1]]
Methylation = qs::qread(system.file("extdata", "Methylation_4.qs",
package="SingleMoleculeFootprinting"))
MethSM_2 = Methylation[[2]][[1]]
rbind.fill.Matrix(MethSM_1, MethSM_2)
```

---

<code>rowMeans_drop0</code>	<i>Calculate rowMeans after dropping zeros</i>
-----------------------------	--

---

**Description**

Calculate rowMeans after dropping zeros

**Usage**

```
rowMeans_drop0(MethSM)
```

**Arguments**

<code>MethSM</code>	one single molecule sparse matrix
---------------------	-----------------------------------

**Value**

rowMeans (N.b. this is +1 based)

---

SingleTFStateQuantificationPlot  
*Single TF state quantification bar*

---

**Description**

Single TF state quantification bar

**Usage**

```
SingleTFStateQuantificationPlot(SortedReads)
```

**Arguments**

SortedReads      Sorted reads as returned by SortReadsBySingleTF

---

SingleTFStates      *Hard-coded interpretation of biological states from single TF sorting*

---

**Description**

Hard-coded interpretation of biological states from single TF sorting

**Usage**

```
SingleTFStates()
```

**Value**

list of states

**Examples**

```
SingleTFStates()
```

---

SortReads	<i>Sort reads by single TF</i>
-----------	--------------------------------

---

**Description**

Sort reads by single TF

**Usage**

```
SortReads(MethSM, BinsCoordinates, coverage = NULL)
```

**Arguments**

MethSM	Single molecule matrix
BinsCoordinates	IRanges object of absolute coordinates for sorting bins
coverage	integer. Minimum number of reads covering all sorting bins for sorting to be performed

**Value**

list of sorted reads

**Examples**

```
library(IRanges)

Methylation = qs::qread(system.file("extdata", "Methylation_3.qs",
package="SingleMoleculeFootprinting"))
TFBS = qs::qread(system.file("extdata", "TFBSs_3.qs",
package="SingleMoleculeFootprinting"))

bins = list(c(-35,-25), c(-15,15), c(25,35))
TFBS_center = start(TFBS) + (end(TFBS)-start(TFBS))/2
BinsCoordinates = IRanges(
start = c(TFBS_center+bins[[1]][1], TFBS_center+bins[[2]][1], TFBS_center+bins[[3]][1]),
end = c(TFBS_center+bins[[1]][2], TFBS_center+bins[[2]][2], TFBS_center+bins[[3]][2])
)

SortedReads = SortReads(Methylation[[2]]$SMF_MM_TKO_DE_, BinsCoordinates, coverage = 20)
```

---

SortReadsBySingleTF	<i>Wrapper to SortReads for single TF case</i>
---------------------	--

---

**Description**

Wrapper to SortReads for single TF case

**Usage**

```
SortReadsBySingleTF(
  MethSM,
  TFBS,
  bins = list(c(-35, -25), c(-15, 15), c(25, 35)),
  coverage = 20
)
```

**Arguments**

MethSM	Single molecule matrix list as returned by CallContextMethylation
TFBS	Transcription factor binding site to use for sorting, passed as a GRanges object of length 1
bins	list of 3 relative bin coordinates. Defaults to list(c(-35,-25), c(-15,15), c(25,35)). bins[[1]] represents the upstream bin, with coordinates relative to the start of the TFBS. bins[[2]] represents the TFBS bin, with coordinates relative to the center of the TFBS. bins[[3]] represents the downstream bin, with coordinates relative to the end of the TFBS.
coverage	integer. Minimum number of reads covering all sorting bins for sorting to be performed. Defaults to 20

**Value**

List of reads sorted by single TF

**Examples**

```
Methylation = qs::qread(system.file("extdata", "Methylation_3.qs",
package="SingleMoleculeFootprinting"))
TFBSs = qs::qread(system.file("extdata", "TFBSs_3.qs",
package="SingleMoleculeFootprinting"))

SortedReads = SortReadsBySingleTF(MethSM = Methylation[[2]], TFBS = TFBSs)
```

---

SortReadsBySingleTF\_MultiSiteWrapper

*Convenience wrapper to sort single molecule according to TFBS clusters at multiple sites in the genome*

---

**Description**

The function starts from a list of single TFBSs, arranges them into clusters, calls methylation at the interested sites and outputs sorted reads

**Usage**

```
SortReadsBySingleTF_MultiSiteWrapper(
  sampleFile,
  samples,
  genome,
  coverage = 20,
  ConvRate.thr = NULL,
  CytosinesToMask = NULL,
  TFBSs,
  max_interTF_distance = 1e+05,
  max_window_width = 5e+06,
  min_cluster_width = 600,
  fix.window.size = FALSE,
  max.window.size = NULL,
  sorting_coverage = 30,
  bins = list(c(-35, -25), c(-15, 15), c(25, 35)),
  cores = 1
)
```

**Arguments**

sampleFile	QuasR pointer file
samples	samples to use, from the SampleName field of the sampleFile
genome	BSgenome
coverage	coverage threshold as integer for least number of reads to cover a cytosine for it to be carried over in the analysis. Defaults to 20.
ConvRate.thr	Convesion rate threshold. Double between 0 and 1, defaults to NULL. To skip this filtering step, set to NULL. For more information, check out the details section.
CytosinesToMask	CytosinesToMask object. Passed to MaskSNPs function
TFBSs	GRanges object of transcription factor binding sites coordinates
max_interTF_distance	maximum distance between two consecutive TFBSs for them to be grouped in the same window
max_window_width	upper limit to window width. This value should be adjusted according to the user's system as it determines the amount of memory used in the later context methylation call
min_cluster_width	lower limit to window width. Corresponds to the scenario when a window contains a single TFBS.
fix.window.size	defaults to FALSE. Passed to Create_MethylationCallingWindows function.
max.window.size	defaults to NULL. Passed to Create_MethylationCallingWindows function.
sorting_coverage	integer. Minimum number of reads covering all sorting bins for sorting to be performed. Defaults to 30.

bins	list of 3 relative bin coordinates. Defaults to list(c(-35,-25), c(-15,15), c(25,35)). bins[[1]] represents the upstream bin, with coordinates relative to the start of the most upstream TFBS. bins[[2]] represents all the TFBS bins, with coordinates relative to the center of each TFBS. bins[[3]] represents the downstream bin, with coordinates relative to the end of the most downstream TFBS.
cores	number of cores to use for parallel processing of multiple Methylation Calling Windows (i.e. groupings of adjacent TFBS clusters)

### Value

list where [[1]] is the TFBSs GRanges object describing coordinates TFBSs used to sort single molecules [[2]] is a list of SortedReads nested per TFBS\_cluster and sample [[3]] is a tibble reporting the count (and frequency) of reads per state, sample and TFBS cluster

### Examples

```
sampleFile = NULL
if(!is.null(sampleFile)){
  SortReadsBySingleTF_MultiSiteWrapper(
    sampleFile = sampleFile,
    samples = samples,
    genome = BSgenome.Mmusculus.UCSC.mm10,
    coverage = 20, ConvRate.thr = NULL,
    CytosinesToMask = NULL,
    TFBSs = KLF4s,
    max_interTF_distance = NULL, max_window_width = NULL, min_cluster_width = NULL,
    fix.window.size = TRUE, max.window.size = 50,
    cores = 4
  ) -> sorting_results
}
```

---

SortReadsByTFCluster *Wrapper to SortReads for TF cluster case*

---

### Description

Wrapper to SortReads for TF cluster case

### Usage

```
SortReadsByTFCluster(
  MethSM,
  TFBS_cluster,
  bins = list(c(-35, -25), c(-7, 7), c(25, 35)),
  coverage = 30
)
```

**Arguments**

MethSM	Single molecule matrix list as returned by CallContextMethylation
TFBS_cluster	Transcription factor binding sites to use for sorting, passed as a GRanges object of length > 1
bins	list of 3 relative bin coordinates. Defaults to list(c(-35,-25), c(-7,7), c(25,35)). bins[[1]] represents the upstream bin, with coordinates relative to the start of the most upstream TFBS. bins[[2]] represents all the TFBS bins, with coordinates relative to the center of each TFBS. bins[[3]] represents the downstream bin, with coordinates relative to the end of the most downstream TFBS.
coverage	integer. Minimum number of reads covering all sorting bins for sorting to be performed. Defaults to 30

**Value**

List of reads sorted by TF cluster

**Examples**

```
Methylation = qs::qread(system.file("extdata", "Methylation_4.qs",
package="SingleMoleculeFootprinting"))
TFBSs = qs::qread(system.file("extdata", "TFBSs_1.qs",
package="SingleMoleculeFootprinting"))

SortedReads = SortReadsByTFCluster(MethSM = Methylation[[2]], TFBS_cluster = TFBSs)
```

---

SortReadsByTFCluster\_MultiSiteWrapper

*Convenience wrapper to sort single molecule according to TFBS clusters at multiple sites in the genome*

---

**Description**

The function starts from a list of single TFBSs, arranges them into clusters, calls methylation at the interested sites and outputs sorted reads

**Usage**

```
SortReadsByTFCluster_MultiSiteWrapper(
  sampleFile,
  samples,
  genome,
  coverage = 20,
  ConvRate.thr = 0.8,
  CytosinesToMask = NULL,
  TFBSs,
  max_intersite_distance = 75,
  min_intersite_distance = 15,
  max_cluster_size = 10,
  max_cluster_width = 300,
```



```

    add.single.TFs = TRUE,
    max_intercluster_distance = 1e+05,
    max_window_width = 5e+06,
    min_cluster_width = 600,
    fix.window.size = FALSE,
    max.window.size = NULL,
    sorting_coverage = 30,
    bins = list(c(-35, -25), c(-7, 7), c(25, 35)),
    cores = 1
)

```

### Arguments

sampleFile	QuasR pointer file
samples	samples to use, from the SampleName field of the sampleFile
genome	BSgenome
coverage	coverage threshold as integer for least number of reads to cover a cytosine for it to be carried over in the analysis. Defaults to 20.
ConvRate.thr	Convesion rate threshold. Double between 0 and 1, defaults to 0.8. To skip this filtering step, set to NULL. For more information, check out the details section.
CytosinesToMask	CytosinesToMask object. Passed to MaskSNPs function
TFBSs	GRanges object of transcription factor binding sites coordinates
max_intersite_distance	maximum allowed distance in base pairs between two TFBS centers for them to be considered part of the same cluster. Defaults to 75.
min_intersite_distance	minimum allowed distance in base pairs between two TFBS centers for them not to be discarded as overlapping. This parameter should be set according to the width of the bins used for later sorting. Defaults to 15.
max_cluster_size	maximum number of TFBSs to be contained in any given cluster. Defaults to 10
max_cluster_width	maximum cluster width in bp. Defaults to 300
add.single.TFs	whether to add to output the TFBSs that didn't make it into clusters. Defaults to TRUE
max_intercluster_distance	maximum distance between two consecutive TFBS clusters for them to be grouped in the same window
max_window_width	upper limit to window width. This value should be adjusted according to the user's system as it determines the amount of memory used in the later context methylation call
min_cluster_width	lower limit to window width. Corresponds to the scenario when a window contains a single TFBS cluster.
fix.window.size	defaults to FALSE. Passed to Create_MethylationCallingWindows function.
max.window.size	defaults to NULL. Passed to Create_MethylationCallingWindows function.

sorting_coverage	integer. Minimum number of reads covering all sorting bins for sorting to be performed. Defaults to 30.
bins	list of 3 relative bin coordinates. Defaults to list(c(-35,-25), c(-7,7), c(25,35)). bins[[1]] represents the upstream bin, with coordinates relative to the start of the most upstream TFBS. bins[[2]] represents all the TFBS bins, with coordinates relative to the center of each TFBS. bins[[3]] represents the downstream bin, with coordinates relative to the end of the most downstream TFBS.
cores	number of cores to use for parallel processing of multiple Methylation Calling Windows (i.e. groupings of adjacent TFBS clusters)

**Value**

list where [[1]] is the TFBS\_Clusters object describing coordinates and composition of the TFBS clusters used to sort single molecules [[2]] is a list of SortedReads nested per TFBS\_cluster and sample [[3]] is a tibble reporting the count (and frequency) of reads per state, samples and TFBS cluster

**Examples**

```

sampleFile = NULL
if(!is.null(sampleFile)){
  SortReadsByTFCluster_MultiSiteWrapper(
    sampleFile = sampleFile,
    samples = samples,
    genome = BSgenome.Mmusculus.UCSC.mm10,
    coverage = 20, ConvRate.thr = NULL,
    CytosinesToMask = NULL,
    TFBSs = KLF4s,
    max_interTF_distance = NULL, max_window_width = NULL, min_cluster_width = NULL,
    fix.window.size = TRUE, max.window.size = 50,
    cores = 4
  ) -> sorting_results
}

```

---

StateQuantification    *Convenience for calculating state frequencies*

---

**Description**

Convenience for calculating state frequencies

**Usage**

```
StateQuantification(SortedReads, states)
```

**Arguments**

SortedReads	List of sorted reads (can be multiple samples) as returned by either read sorting function (SortReads, SortReadsBySingleTF, SortReadsByTFCluster)
states	states reporting the biological interpretation of patterns as return by either SingleTFStates or TFPairStates functions. If NULL (default) will return frequencies without biological interpretation.

**Value**

tibble with state frequency information

**Examples**

```
Methylation = qs::qread(system.file("extdata", "Methylation_4.qs",
package="SingleMoleculeFootprinting"))
TFBSs = qs::qread(system.file("extdata", "TFBSs_1.qs",
package="SingleMoleculeFootprinting"))
SortedReads = SortReadsByTFCluster(MethSM = Methylation[[2]], TFBS_cluster = TFBSs)
StateQuantification(SortedReads = SortedReads, states = TFPairStates())
```

---

StateQuantificationBySingleTF

*Convenience for calculating state frequencies after sorting reads by single TF*

---

**Description**

wraps around StateQuantification function

**Usage**

```
StateQuantificationBySingleTF(SortedReads)
```

**Arguments**

SortedReads      List of sorted reads (can be multiple samples) as returned by SortReadsBySingleTF (or SortReads run with analogous parameters)

**Value**

tibble with state frequency information

**Examples**

```
Methylation = qs::qread(system.file("extdata", "Methylation_3.qs",
package="SingleMoleculeFootprinting"))
TFBSs = qs::qread(system.file("extdata", "TFBSs_3.qs",
package="SingleMoleculeFootprinting"))
SortedReads = SortReadsBySingleTF(MethSM = Methylation[[2]], TFBS = TFBSs)
StateQuantificationBySingleTF(SortedReads = SortedReads)
```

---

StateQuantificationByTFPair

*Convenience for calculating state frequencies after sorting reads by TF pair*

---

### Description

wraps around StateQuantification function

### Usage

```
StateQuantificationByTFPair(SortedReads)
```

### Arguments

SortedReads      List of sorted reads (can be multiple samples) as returned by SortReadsByTF-Cluster run for clusters of size 2 (or SortReads run with analogous parameters)

### Value

tibble with state frequency information

### Examples

```
Methylation = qs::qread(system.file("extdata", "Methylation_4.qs",
package="SingleMoleculeFootprinting"))
TFBSs = qs::qread(system.file("extdata", "TFBSs_1.qs",
package="SingleMoleculeFootprinting"))
SortedReads = SortReadsByTFCluster(MethSM = Methylation[[2]], TFBS_cluster = TFBSs)
StateQuantificationByTFPair(SortedReads = SortedReads)
```

---

StateQuantificationPlot

*Plot states quantification bar*

---

### Description

Plot states quantification bar

### Usage

```
StateQuantificationPlot(SortedReads, states)
```

### Arguments

SortedReads      Sorted reads object as returned by SortReads function  
states              either SingleTFStates() or TFPairStates()

**Value**

Bar plot quantifying states

**Examples**

```
library(GenomicRanges)

RegionOfInterest = GRanges("chr12", IRanges(20464551, 20465050))
Methylation = qs::qread(system.file("extdata", "Methylation_3.qs",
package="SingleMoleculeFootprinting"))
TFBSs = qs::qread(system.file("extdata", "TFBSs_3.qs", package="SingleMoleculeFootprinting"))
SortedReads = SortReadsBySingleTF(MethSM = Methylation[[2]], TFBS = TFBSs)

StateQuantificationPlot(SortedReads = SortedReads, states = SingleTFStates())
```

---

SubsetGRangesForSamples

*Subset Granges for given samples*

---

**Description**

Inner utility for LowCoverageMethRateDistribution

**Usage**

```
SubsetGRangesForSamples(GRanges_obj, Samples)
```

**Arguments**

GRanges_obj	GRanges object as returned by CallContextMethylation function
Samples	vector of sample names as they appear in the SampleName field of the QuasR sampleFile

---

TFPairStateQuantificationPlot

*TF pair state quantification bar*

---

**Description**

TF pair state quantification bar

**Usage**

```
TFPairStateQuantificationPlot(SortedReads)
```

**Arguments**

SortedReads	Sorted reads as returned by SortReadsByTFCluster
-------------	--

---

TFPairStates	<i>Design states for TF pair case</i>
--------------	---------------------------------------

---

**Description**

Design states for TF pair case

**Usage**

TFPairStates()

**Value**

list of states

**Examples**

TFPairStates()

# Index

Arrange\_TFBSs\_clusters, 3

BaitCapture, 4  
BinMethylation, 5

CallContextMethylation, 5  
cbind.fill.Matrix, 7  
CollapseStrands, 8  
CollapseStrandsSM, 8  
CollectCompositeData, 9  
colMeans\_drop0, 10  
CompositeMethylationCorrelation, 10  
CompositePlot, 12  
ConversionRate, 12  
CoverageFilter, 13  
Create\_MethylationCallingWindows, 13

DetectExperimentType, 14

filter\_reads\_from\_MethGR, 16  
FilterByConversionRate, 15  
FilterContextCytosines, 15  
full.join.granges, 16

GetQuasRprj, 17  
GetSingleMolMethMat, 17  
GRanges\_to\_DF, 18

HierarchicalClustering, 18

LowCoverageMethRate\_RMSE, 19

MaskSNPs, 19  
MethSM.to.MethGR, 20

panel.cor, 20  
panel.hist, 21  
panel.jet, 21  
Plot\_LowCoverageMethRate, 25  
Plot\_LowCoverageMethRate\_RMSE, 25  
PlotAvgSMF, 21  
PlotSingleMoleculeStack, 22  
PlotSingleSiteSMF, 23  
PlotSM, 24

rbind.fill.Matrix, 26

rowMeans\_drop0, 26

SingleTFStateQuantificationPlot, 27  
SingleTFStates, 27  
SortReads, 28  
SortReadsBySingleTF, 28  
SortReadsBySingleTF\_MultiSiteWrapper, 29  
SortReadsByTFCluster, 31  
SortReadsByTFCluster\_MultiSiteWrapper, 32  
StateQuantification, 34  
StateQuantificationBySingleTF, 35  
StateQuantificationByTFPair, 36  
StateQuantificationPlot, 36  
SubsetGRangesForSamples, 37

TFPairStateQuantificationPlot, 37  
TFPairStates, 38