Package: TEQC (via r-universe)

October 30, 2026

Type Package

Title Quality control for target capture experiments

Version 4.35.0

Date 2025-07-23

Author M. Hummel, S. Bonnin, E. Lowy, G. Roma

Maintainer Sarah Bonnin <Sarah.Bonnin@crg.eu>

Description Target capture experiments combine hybridization-based (in solution or on microarrays) capture and enrichment of genomic regions of interest (e.g. the exome) with high throughput sequencing of the captured DNA fragments. This package provides functionalities for assessing and visualizing the quality of the target enrichment process, like specificity and sensitivity of the capture, per-target read coverage and so on.

Depends methods, BiocGenerics (>= 0.1.0), IRanges (>= 1.13.5), Rsamtools, hwriter

Imports S4Vectors, Seqinfo, GenomicRanges, Biobase (>= 2.15.1)

License GPL (>= 2)

LazyLoad yes

biocViews QualityControl, Microarray, Sequencing, Genetics

Config/pak/sysreqs libbz2-dev liblzma-dev xz-utils zlib1g-dev

Repository https://bioc.r-universe.dev

Date/Publication 2026-04-28 12:35:02 UTC

RemoteUrl https://github.com/bioc/TEQC

RemoteRef HEAD

RemoteSha d0ae113b815a6a6442f17ef539901562d966e198

Contents

chrom.barplot 2
coverage.correlation 3

coverage.density 5
coverage.GC 6
coverage.hist 8
coverage.plot 9
coverage.target 11
coverage.targetlength.plot 12
coverage.uniformity 13
covered.k 15
duplicates.barplot 16
fraction.reads.target 18
fraction.target 19
get.baits 20
get.reads 21
get.targets 23
insert.size.hist 24
make.wigfiles 25
multiTEQCreport 26
reads2pairs 28
readsPerTarget 29
TEQCreport 30

Index **33**

`chrom.barplot` *Reads per chromosome barplot*

Description

Barplot of numbers (or fractions) of reads (and targets) falling on each chromosome

Usage

```
chrom.barplot(reads, targets, plotchroms, col = c("darkgreen", "orange"), ylab, legendpos = "topright",
```

Arguments

`reads`	`RangedData` table containing read positions, i.e. output from `get.reads`. To ensure a useful ordering of the bars, the chromosome information ('spaces' of `reads`) should be given as "chr" plus a number/letter [plus further specification], e.g. "chr1", "chrX", "chr17_ctg5_hap1", "chrUn_gl000211".
`targets`	Optional `RangedData` table containing positions of target regions, i.e. output from `get.targets`. The chromosome information should match the one of `reads`. If `targets` is missing, only numbers of reads will be displayed.
`plotchroms`	character vector specifying the chromosomes that shall be included in the plot (and their desired order)
`col`	color(s) of the bars
`ylab`	y-axis label

legendpos	Position of the legend. String from the list "bottomright", "bottom", "bottomleft", "left", "topleft", "top", "topright", "right" and "center". Ignored if targets is missing.
...	graphical parameters passed to barplot

Details

If targets is not specified, absolute read counts per chromosome are shown in the barplot. If targets is provided, fractions of reads and targets are shown. For reads, this is the fraction within the total *number* of reads (since reads are expected to have all the same length). In contrast, for the targets, the fraction of targeted bases on each chromosome is calculated. Since targets might vary in length it is reasonable to account for the actual target *sizes* instead of considering merely numbers of targets per chromosome.

Value

Barplot of reads and optionally targets per chromosome.

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

See Also

get.reads

Examples

```
## get reads and targets
exptPath <- system.file("extdata", package="TEQC")
readsfile <- file.path(exptPath, "ExampleSet_Reads.bed")
reads <- get.reads(readsfile, idcol=4, skip=0)
targetsfile <- file.path(exptPath, "ExampleSet_Targets.bed")
targets <- get.targets(targetsfile, skip=0)

chrom.barplot(reads, targets)
```

---

coverage.correlation *Coverage correlation plot*

Description

Visualization of target coverage correlations between pairs of samples.

Usage

```
coverage.correlation(coveragelist, normalized = TRUE, plotfrac = 0.001, seed = 123, labels, main, pch =
            cex.pch = 2, cex.main = 1.2, cex.corr, font.labels = 1, font.main = 2, ...)
```

Arguments

coveragelist	List where each element is the output of function coverage.target, where option perBase had to be set to TRUE.
normalized	if TRUE, correlation of normalized target coverages will be shown; original coverages otherwise
plotfrac	numeric value between 0 and 1. Coverages for a fraction of plotfrac of all target bases are shown.
seed	seed for random selection of plotfrac bases
labels	sample names that are written in the diagonal panels; if missing, names of coveragelist are taken; if those are NULL, "sample 'i'" is shown
main	main title
pch	plot symbol for the scatter plots
cex.labels, cex.pch, cex.main	sizes of sample labels, plot symbols, main title
cex.corr	size of the correlation values; if missing, sizes are made proportionally to the values of (positive) correlation.
font.labels, font.main	fonts for sample labels and main title
...	further graphical parameters, e.g. limits and symbol color for the scatter plots

Details

If normalized = TRUE, the function calculates normalized coverages: per-base coverages divided by average coverage over all targeted bases. Normalized coverages are not dependent on the absolute quantity of reads and are hence better comparable between different samples or even different experiments.

Value

'pairs'-style plot where upper panels show scatter plot of (a randomly chosen fraction of) coverage values for pairs of samples. The lower panels show the respective Pearson correlation coefficients, calculated using all coverage values (even if not all of them are shown in the scatter plot).

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

See Also

coverage.target, covered.k, coverage.hist, coverage.density, coverage.uniformity, coverage.plot

Examples

```
## get reads and targets
exptPath <- system.file("extdata", package="TEQC")
readsfile <- file.path(exptPath, "ExampleSet_Reads.bed")
reads <- get.reads(readsfile, idcol=4, skip=0)
targetsfile <- file.path(exptPath, "ExampleSet_Targets.bed")
targets <- get.targets(targetsfile, skip=0)

## calculate per-base coverages
Coverage <- coverage.target(reads, targets, perBase=TRUE)

## simulate another sample
r <- sample(length(reads), 0.1 * length(reads))
reads2 <- reads[-r,,drop=TRUE]
Coverage2 <- coverage.target(reads2, targets, perBase=TRUE)

## coverage uniformity plot
covlist <- list(Coverage, Coverage2)
coverage.correlation(covlist, plotfrac=0.1)
```

`coverage.density` *Coverage density plot*

Description

Visualization of target coverage density for one or more samples.

Usage

```
coverage.density(coveragelist, normalized = TRUE, legend, main, xlab, col, lwd, lty, xlim, ylim, ...)
```

Arguments

`coveragelist`	Output of function `coverage.target`, where option `perBase` had to be set to `TRUE`, i.e. a list with elements `coverageTarget` and `avgTargetCoverage`. Or, when density of several samples shall be visualized, a list with respective outputs of `coverage.target`.
`normalized`	if `TRUE`, densities of normalized coverages will be shown; original coverages otherwise
`legend`	legend text. If missing, names of `coveragelist` will be taken. If `NULL`, no legend will be drawn.
`main`	main title
`xlab`	x-axis label
`col`	line color(s)
`lwd`	line width(s)
`lty`	line style(s)
`xlim`, `ylim`	x- and y-axis coordinate ranges
`...`	further graphical parameters passed to `plot`

Details

If normalized = TRUE, the function calculates normalized coverages: per-base coverages divided by average coverage over all targeted bases. Normalized coverages are not dependent on the absolute quantity of reads and are hence better comparable between different samples or even different experiments.

Value

Line plot(s) showing densities.

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

See Also

coverage.target, covered.k, coverage.hist, coverage.uniformity, coverage.correlation, coverage.plot

Examples

```
## get reads and targets
exptPath <- system.file("extdata", package="TEQC")
readsfile <- file.path(exptPath, "ExampleSet_Reads.bed")
reads <- get.reads(readsfile, idcol=4, skip=0)
targetsfile <- file.path(exptPath, "ExampleSet_Targets.bed")
targets <- get.targets(targetsfile, skip=0)

## calculate per-base coverages
Coverage <- coverage.target(reads, targets, perBase=TRUE)

## coverage density
coverage.density(Coverage)
```

---

coverage.GC *Bait coverage versus GC content plot*

Description

Calculates and plots average normalized coverage per hybridization probe versus GC content of the respective probe. A smoothing spline is added to the scatter plot.

Usage

```
coverage.GC(coverageAll, baits, returnBaitValues = FALSE, linecol = "darkred", lwd, xlab, ylab, pch, co
```

Arguments

coverageAll	RleList containing Rle vectors of per-base coverages for each chromosome, i.e. coverageAll output of coverage.target
baits	A RangedData table holding the hybridization probe ("bait") positions and sequences, i.e. output ofget.baits
returnBaitValues	if TRUE, average coverage, average normalized coverage and GC content per bait are returned
linecol, lwd	color and width of spline curve
xlab, ylab	x- and y-axis labels
pch	plotting character
col, cex	color and size of plotting character
...	further graphical parameters passed to plot

Details

The function calculates average normalized coverages for each bait: the average coverage over all bases within a bait is divided by the average coverage over all bait-covered bases. Normalized coverages are not dependent on the absolute quantity of reads and are hence better comparable between different samples or even different experiments.

Value

A scatterplot with normalized per-bait coverages on the y-axis and GC content of respective baits on the x-axis. A smoothing spline is added to the plot.

If returnBaitValues = TRUE average coverage, average normalized coverage and GC content per bait are returned as 'values' columns of the baits input RangedData table

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

References

Tewhey R, Nakano M, Wang X, Pabon-Pena C, Novak B, Giuffre A, Lin E, Happe S, Roberts DN, LeProust EM, Topol EJ, Harismendy O, Frazer KA. Enrichment of sequencing targets from the human genome by solution hybridization. Genome Biol. 2009; 10(10): R116.

See Also

coverage.target, covered.k, coverage.hist, coverage.plot, coverage.uniformity, coverage.targetlength.plot

Examples

```
## get reads and targets
exptPath <- system.file("extdata", package="TEQC")
readsfile <- file.path(exptPath, "ExampleSet_Reads.bed")
reads <- get.reads(readsfile, idcol=4, skip=0)
targetsfile <- file.path(exptPath, "ExampleSet_Targets.bed")
targets <- get.targets(targetsfile, skip=0)

## calculate per-base coverages
Coverage <- coverage.target(reads, targets, perBase=TRUE)

## get bait positions and sequences
baitsfile <- file.path(exptPath, "ExampleSet_Baits.txt")
baits <- get.baits(baitsfile, chrcol=3, startcol=4, endcol=5, seqcol=2)

## do coverage vs GC plot
coverage.GC(Coverage$coverageAll, baits)
```

`coverage.hist` *Coverage histogram*

Description

Histogram and cumulative density of target base coverages

Usage

```
coverage.hist(coverageTarget, col.hist = "lightblue", col.line = "orange", covthreshold, outline = FALS
```

Arguments

`coverageTarget`	`RleList` containing `Rle` vectors of per-target-base coverages for each chromosome, i.e. `coverageTarget` output from `coverage.target`
`col.hist`	histogram color
`col.line`	color of the cumulative density line
`covthreshold`	indicates with dashed vertical and horizontal lines, which fraction of bases has a coverage of at least `covthreshold`; if missing, no dashed lines are drawn
`outline`	if `FALSE`, outliers (according to `boxplot.stats`) are removed before plotting.
`breaks`	number of cells for the histogram, or string naming an algorithm to compute the number of cells, or function to compute the number of cells, or vector giving the breakpoints between histogram cells (see `?hist`) but the latter option only with equidistant breakpoints
`xlab, ylab`	x- and y-axis labels
`main`	plot title
`lwd`	line width
`...`	further graphical parameters, passed to `plot(histogram)`

Value

Histogram of read coverages for bases within the target. Additionally, a line and the right axis indicate the cumulative fraction of target bases with coverage of at least x. If option `covthreshold` is specified, red dashed lines highlight the cumulative fraction of target bases with at least the specified coverage.

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

See Also

`coverage.target`, `coverage.uniformity`, `coverage.density`, `coverage.plot`, `coverage.targetlength.plot`

Examples

```
## get reads and targets
exptPath <- system.file("extdata", package="TEQC")
readsfile <- file.path(exptPath, "ExampleSet_Reads.bed")
reads <- get.reads(readsfile, idcol=4, skip=0)
targetsfile <- file.path(exptPath, "ExampleSet_Targets.bed")
targets <- get.targets(targetsfile, skip=0)

## calculate per-base coverages
Coverage <- coverage.target(reads, targets, perBase=TRUE)

## coverage histogram
coverage.hist(Coverage$coverageTarget, covthreshold=8)
```

---

coverage.plot *Coverage versus base position plot*

Description

Line plot of per-base coverages along a genomic region. Position of target regions can be shown.

Usage

```
coverage.plot(coverageAll, targets, chr, Start, End, Offset = 0, add = FALSE, col.line = 1, col.target =
```

Arguments

`coverageAll`	`RleList` containing `Rle` vectors of per-base coverages for each chromosome, i.e. `coverageAll` output from `coverage.target`
`targets`	optional; `RangedData` table containing positions of target regions, i.e. output from `get.targets`; if missing no genomic regions are highlighted
`chr`	on which chromosome the region to plot is located (string, e.g. "chr1")

Start	genomic position where to start the plot
End	genomic position where to end the plot
Offset	integer; highlight Offset bases on both sides of each targeted region; defaults to 0
add	if TRUE, the coverage line of a new sample is added to an already existing plot
col.line	color of the coverage line
col.target	color of the bar indicating target regions
col.offset	color for highlighting Offset on the sides of target regions
xlab, ylab	x- and y-axis labels
ylim	y-axis coordinate ranges
...	further graphical parameters, passed to plot

Details

If coverage of a new sample is added to an existing plot with add = TRUE, parameters chr, Start, End still have to be specified and should be the same as in the previous call in order to make sense. Parameters targets and Offset can but do not have to be given again. They can also differ from the previous ones, if for the additional sample a different target was captured.

Value

Line plot showing per-base read coverages for a specified genomic region. When positions of target regions are provided, a bar on the bottom indicates their location such that coverage can be related to the captured targets.

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

See Also

coverage.target, make.wigfiles, covered.k, coverage.hist, coverage.uniformity, coverage.targetlength.plot

Examples

```
## get reads and targets
exptPath <- system.file("extdata", package="TEQC")
readsfile <- file.path(exptPath, "ExampleSet_Reads.bed")
reads <- get.reads(readsfile, idcol=4, skip=0)
targetsfile <- file.path(exptPath, "ExampleSet_Targets.bed")
targets <- get.targets(targetsfile, skip=0)

## calculate per-base coverages
Coverage <- coverage.target(reads, targets, perBase=TRUE)

## coverage plot
coverage.plot(Coverage$coverageAll, targets, Offset=100, chr="chr1", Start=11157524, End=11158764)
```

coverage.target *Calculates read coverage*

Description

Calculates average coverage over all target bases, average coverage for each target separately, and per-base coverage for all and for targeted bases

Usage

```
coverage.target(reads, targets, Offset = 0, perTarget = TRUE, perBase = TRUE)
```

Arguments

reads	RangedData table containing positions of sequenced reads, i.e. output from get.reads
targets	RangedData table containing positions of target regions, i.e. output from get.targets
Offset	integer; add Offset bases on both sides to targeted regions and potentially collapse resulting overlapping target regions
perTarget	if TRUE, coverage average and standard deviation per target are calculated and returned
perBase	if TRUE, the per-base coverages i) only for targeted bases and ii) for all sequenced and/or targeted bases, are returned

Value

A list is returned with elements

avgTargetCoverage
: average coverage over all target bases

targetCoverageSD
: standard deviation of coverage of all target bases

targetCoverageQuantiles
: 0% (minium), 25%, 50% (median), 75% and 100% (maximum) quantiles of coverage of all target bases

targetCoverages
: Input RangedData table targets with two additional 'values' columns avgCoverage and coverageSD. The former contains the average coverage for each target, the latter the respective coverage standard deviation. Only returned if perTarget equals TRUE.

coverageAll
: RleList containing a Rle vector for each chromosome with coverages for all bases that are sequenced and/or within a targeted; only returned if perBase equals TRUE

coverageTarget
: RleList containing a Rle vector for each chromosome with coverages for target bases only; only returned if perBase equals TRUE

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

See Also

covered.k, coverage.hist, coverage.uniformity, coverage.plot, coverage.targetlength.plot

Examples

```
## get reads and targets
exptPath <- system.file("extdata", package="TEQC")
readsfile <- file.path(exptPath, "ExampleSet_Reads.bed")
reads <- get.reads(readsfile, idcol=4, skip=0)
targetsfile <- file.path(exptPath, "ExampleSet_Targets.bed")
targets <- get.targets(targetsfile, skip=0)

## total average, per-base and per-target coverages
Coverage <- coverage.target(reads, targets)
```

---

coverage.targetlength.plot

Number of reads or average coverage versus target length scatter plot

---

Description

Plots either numbers of on-target reads or average per-target coverage (or potentially other per-target values) against respective target lengths. A smoothing spline is added to the scatter plot.

Usage

```
coverage.targetlength.plot(targets, plotcolumn, linecol = 2, xlab, ylab, lwd, pch, cex, ...)
```

Arguments

targets	RangedData table containing positions of target regions and further 'values' columns that should be plotted, i.e. output from coverage.target or readsPerTarget
plotcolumn	name or index of column to plot (of the 'values' DataFrame within targets)
linecol	color of spline curve
xlab, ylab	x- and y-axis labels
lwd	line width of spline curve
pch	plotting character
cex	size of plotting character
...	further graphical parameters, passed to plot

Details

coverage.target and readsPerTarget can be used to calculate average per-target coverages and numbers of reads overlapping each target. The values are added to the RangedData table containing the target positions. Such RangedData table can then be used for plotting the calculated values against the respecitve target lengths.

Value

A scatterplot with the given per-target values on the y-axis and corresponding target lengths on the x-axis. A smoothing spline is added to the plot.

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

See Also

coverage.target, readsPerTarget, covered.k, coverage.hist, coverage.uniformity, coverage.plot

Examples

```
## get reads and targets
exptPath <- system.file("extdata", package="TEQC")
readsfile <- file.path(exptPath, "ExampleSet_Reads.bed")
reads <- get.reads(readsfile, idcol=4, skip=0)
targetsfile <- file.path(exptPath, "ExampleSet_Targets.bed")
targets <- get.targets(targetsfile, skip=0)

## get average per-target coverage
Coverage <- coverage.target(reads, targets, perTarget=TRUE)
targets2 <- Coverage$targetCoverages

## get numbers of reads per target
targets2 <- readsPerTarget(reads, targets2)

## coverage vs target length
coverage.targetlength.plot(targets2, plotcolumn="avgCoverage", pch="o")

## coverage vs number of reads per target
coverage.targetlength.plot(targets2, plotcolumn="nReads", pch="o")
```

---

coverage.uniformity *Coverage uniformity plot*

Description

Visualization of target coverage uniformity. A line shows the cumulative fraction of targeted bases that reach at least a certain normalized coverage.

Usage

```
coverage.uniformity(coveragelist, addlines = TRUE, add = FALSE, xlab, ylab, xlim, ylim, col, lwd, ...)
```

Arguments

coveragelist	output of function coverage.target, where option perBase had to be set to TRUE, i.e. a list with elements coverageTarget and avgTargetCoverage
addlines	if TRUE, dashed lines are added to the plot that indicate the fractions of bases achieving at least the average or at least half the average coverage
add	if TRUE, the coverage uniformity line of a new sample is added to an already existing plot
xlab, ylab	x- and y-axis labels
xlim, ylim	x- and y-axis coordinate ranges
col	line color
lwd	line width
...	further graphical parameters passed to plot

Details

The function calculates normalized coverages: per-base coverages divided by average coverage over all targeted bases. Normalized coverages are not dependent on the absolute quantity of reads and are hence better comparable between different samples or even different experiments.

Value

Line plot showing the fraction of targeted bases (y-axis) achieving a normalized coverage of at least x. The x-axis by default is truncated at 1, which corresponds to the average normalized coverage. The steeper the curve is falling, the less uniform is the coverage. If addlines = TRUE, dashed lines indicate the fractions of bases achieving at least the average (=1) or at least half (=0.5) the average coverage.

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

References

Gnirke A, Melnikov A, Maguire J, Rogov P, LeProust EM, Brockman W, Fennell T, Giannoukos G, Fisher S, Russ C, Gabriel S, Jaffe DB, Lander ES, Nusbaum C. Solution hybrid selection with ultra-long oligonucleotides for massively parallel targeted sequencing. Nat Biotechnol. 2009; 27(2): 182-9.

See Also

coverage.target, covered.k, coverage.hist, coverage.density, coverage.plot, coverage.targetlength.plot

Examples

```
## get reads and targets
exptPath <- system.file("extdata", package="TEQC")
readsfile <- file.path(exptPath, "ExampleSet_Reads.bed")
reads <- get.reads(readsfile, idcol=4, skip=0)
targetsfile <- file.path(exptPath, "ExampleSet_Targets.bed")
targets <- get.targets(targetsfile, skip=0)

## calculate per-base coverages
Coverage <- coverage.target(reads, targets, perBase=TRUE)

## coverage uniformity plot
coverage.uniformity(Coverage)
```

`covered.k` *Target capture sensitivity*

Description

Calculates fraction of target bases covered by at least k reads

Usage

```
covered.k(coverageTarget, k = c(1, 2, 3, 5, 10, 20))
```

Arguments

`coverageTarget`	`RleList` containing `Rle` vectors of per-target-base coverages for each chromosome, i.e. `coverageTarget` output from `coverage.target`
`k`	integer vector of k-values for which to show fraction of target bases with coverage >= k

Value

Named vector of same length as `k` giving the corresponding fractions of target bases achieving coverages >= k

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

See Also

`coverage.target`, `coverage.hist`, `coverage.uniformity`, `coverage.plot`, `coverage.targetlength.plot`

Examples

```
## get reads and targets
exptPath <- system.file("extdata", package="TEQC")
readsfile <- file.path(exptPath, "ExampleSet_Reads.bed")
reads <- get.reads(readsfile, idcol=4, skip=0)
targetsfile <- file.path(exptPath, "ExampleSet_Targets.bed")
targets <- get.targets(targetsfile, skip=0)

## calculate per-base coverages
Coverage <- coverage.target(reads, targets, perBase=TRUE)

covered.k(Coverage$coverageTarget, k=c(1,10,20))
```

`duplicates.barplot` *Read duplicates barplot*

Description

Barplot showing fractions of reads / read pairs which are unique and for which there are two, three, ... copies. Separate bars are made for on- and off-target reads / read pairs

Usage

```
duplicates.barplot(reads, targets, returnDups=FALSE, truncateX, col=c("red","lightblue"), xlab, ylab,
```

Arguments

`reads`	`RangedData` table containing positions of sequenced reads, i.e. output from `get.reads`. Alternatively, for paired-end data, it can be the output of `reads2pairs` when multiplicities of read *pairs* instead of fraction of single reads shall be visualized.
`targets`	`RangedData` table containing positions of target regions, i.e. output from `get.targets`
`returnDups`	if `TRUE`, on- and off-target read / read pair multiplicities are returned
`truncateX`	integer; show bars only up to a read / read pair multiplicity of `truncateX` (x-axis)
`col`	vector specifying the two colors of bars and legend for on- and off-target read multiplicities
`xlab, ylab`	x- and y-axis labels
`ylim`	y-axis coordinate ranges
`...`	further graphical parameters passed to `barplot`

Details

Single-end reads are considered as duplicates if they have same start end end position. Paired-end read pairs are considered as duplicates if start and end positions of both reads of the pairs are identical. Usually, duplicates are removed before further analyses (e.g. SNP detection), because they could represent PCR artefacts. However, in target capture experiments it is likely to have also many "real" duplicates (actual different molecules that happen to start at same position) due to the "enrichment" of the target regions. The separation in the barplot between on- and off-target reads / read pairs gives an impression on whether on-target there are more reads with higher multiplicites, which hence might indicate a reasonable amount of "real" duplication. A paired-end read pair is considered on-target if at least one of its reads overlaps with a target.

Value

Barplot where the bar heights correspond to fractions of reads / read pairs which are present in the data with the respective number of copies (x-axis). Fractions are calculated separately for on- and off-target reads / read pairs. A read pair is considered on-target if at least one of its reads overlaps with a target. Absolute numbers (in millions) are additionally written on top of the bars.

If `returnDups` equals `TRUE`, a list with two elements `absolute` and `relative` is returned. The former is a matrix that contains the absolute numbers of reads / read pairs for each multiplicity (columns), for both on- and off-target reads / read pairs (rows). The latter gives row-based fractions which correspond to the bar heights.

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

See Also

`get.reads`, `reads2pairs`, `get.targets`

Examples

```
## get reads and targets
exptPath <- system.file("extdata", package="TEQC")
readsfile <- file.path(exptPath, "ExampleSet_Reads.bed")
reads <- get.reads(readsfile, idcol=4, skip=0)
targetsfile <- file.path(exptPath, "ExampleSet_Targets.bed")
targets <- get.targets(targetsfile, skip=0)

## duplicates barplot for single reads
duplicates.barplot(reads, targets, returnDups=TRUE)

## duplicates barplot for read pairs
readpairs <- reads2pairs(reads)
duplicates.barplot(readpairs, targets, returnDups=TRUE)
```

`fraction.reads.target` *Target capture specificity*

Description

Calculates the fraction of reads that align to target regions. Can also be used to retrieve those reads mapping to targets.

Usage

```
fraction.reads.target(reads, targets, Offset = 0, mappingReads = FALSE)
```

Arguments

`reads`	`RangedData` table containing positions of sequenced reads, i.e. output of `get.reads`. Alternatively, for paired-end data, it can be the output of `reads2pairs` when fraction of on-target read *pairs* shall be calculated instead of fraction of single on-target reads.
`targets`	`RangedData` table containing positions of target regions, i.e. output from `get.targets`
`Offset`	integer; add `Offset` bases on both sides to targeted regions and potentially collapse resulting overlapping target regions
`mappingReads`	if `TRUE`, reduced `RangedData` table with only those reads mapping to target regions is returned. When `reads` is output of `reads2pairs`, `mappingReads` will be the corresponding subset of on-target read pairs.

Value

If `mappingReads` equals `FALSE`, just the fraction of reads / read pairs mapping to targets is returned. When `reads` contains all single reads (i.e. is output of `get.reads`), this is the number of target-overlapping reads, divided by the number of all single reads. When `reads` contains read pairs (i.e. is output of `reads2pairs`), it is the number of read pairs with at least one target-overlapping read, divided by the number of read pairs (= half the number of reads). In case of small targets and large insert sizes the two reads of a pair could be located on both sides of the target without overlap, respectively. Still, the read pair will be counted as on-target, since the corresponding DNA molecule was covering the target.

If `mappingReads` equals `TRUE`, a list is returned with elements

`onTargetFraction`	fraction of reads / read pairs mapping to targets
`mappingReads`	`RangedData` table containing positions of the reads / read pairs mapping to target regions

Note

With the output from `fraction.target` and `fraction.reads.target` the 'enrichment' of the target capture experiment can be calculated as 'fraction of on-target reads / fraction of target within genome'

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

See Also

fraction.target, get.reads, reads2pairs, get.targets

Examples

```
## get reads and targets
exptPath <- system.file("extdata", package="TEQC")
readsfile <- file.path(exptPath, "ExampleSet_Reads.bed")
reads <- get.reads(readsfile, idcol=4, skip=0)
targetsfile <- file.path(exptPath, "ExampleSet_Targets.bed")
targets <- get.targets(targetsfile, skip=0)

## fraction of on-target reads
fraction.reads.target(reads, targets)
```

---

fraction.target *Fraction of the target within the genome*

Description

Calculates the fraction of the reference genome that is targeted

Usage

```
fraction.target(targets, Offset = 0, genome = c(NA, "hg38", "hg19", "hg18"), genomesize)
```

Arguments

targets	RangedData table containing positions of target regions, i.e. output from get.targets
Offset	integer; add Offset bases on both sides to targeted regions and potentially collapse resulting overlapping target regions
genome	genome version targets were designed and reads aligned to. For the given options the total genome size is set automatically. For other genomes or versions, leave this option empty ('NA') and specify the genome size with option 'genomesize'
genomesize	integer: specify the total genome size manually. If 'genomesize' is given, option 'genome' will be ignored.

Value

Returns the fraction of nucleotides within the genome that were targeted.

Note

With the output from fraction.target and fraction.reads.target the 'enrichment' of the target capture experiment can be calculated as 'fraction of on-target reads / fraction of target within genome'

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

See Also

fraction.reads.target, get.targets

Examples

```
exptPath <- system.file("extdata", package="TEQC")
targetsfile <- file.path(exptPath, "ExampleSet_Targets.bed")
targets <- get.targets(targetsfile, skip=0)
fraction.target(targets, genome="hg19")
```

---

get.baits *Read capture hybridization probe positions*

Description

Reads a file containing positions and sequences of the capture hybridization probes and creates a RangedData object.

Usage

```
get.baits(baitsfile, chrcol = 1, startcol = 2, endcol = 3, seqcol = 4, zerobased = TRUE, sep = "\t", heade
```

Arguments

baitsfile	name of file giving the positions and sequences of each hybridization probe ("bait")
chrcol	in which column in baitsfile there is the chromosome information (chromosome information in the file should be in string format, e.g. "chrX")
startcol	in which column there are the starting positions of the baits
endcol	in which column there are the end positions of the baits
seqcol	in which column there are the sequences of the baits
zerobased	if TRUE, start coordinates in baitsfile are assumed to be 0-based and are then converted to 1-based system by adding 1. If FALSE, coordinates are not shifted. In this case they should already be 1-based in baitsfile.
sep	column separator character, defaults to tabs
header	a logical value indicating whether the file contains the names of the variables as its first line; defaults to FALSE
...	further arguments passed to read.delim

Details

The baitsfile containing positions and sequences of hybridization probes has to be created beforehand, in many cases manually. (The function was made like this in order to keep things as general and platform independent as possible.) E.g. with baits designed by Agilent's eArray tool, the baitsfile can be created by merging the files '..._D_BED_...bed' and '..._D_DNAFront_BCBottom_...txt'.

Value

A RangedData table holding the hybridization probe ("bait") positions and sequences. Overlapping or adjacent baits are not collapsed.

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

See Also

get.reads, get.targets

Examples

```
exptPath <- system.file("extdata", package="TEQC")
baitsfile <- file.path(exptPath, "ExampleSet_Baits.txt")
baits <- get.baits(baitsfile, chrcol=3, startcol=4, endcol=5, seqcol=2)
```

---

get.reads *Read genomic positions of sequencing data*

Description

Reads a bedfile containing positions of sequenced read aligned to a reference genome and creates a RangedData object.

Usage

```
get.reads(readsfile, filetype = c("bed", "bam"), chrcol = 1, startcol = 2, endcol = 3, idcol, zerobased =
```

Arguments

readsfile	name of bedfile giving the positions of aligned reads
#!!	
filetype	Input file type. If "bam", the .bam file is read using scanBam, where flag options isUnmappedQuery=FALSE and isSecondaryAlignment=FALSE are used. Defaults to "bed"
# !!	

chrcol	In which column in the reads bedfile there is the chromosome information (chromosome information in the file should be in string format, e.g. "chrX"). Ignored if filetype = "bam".
startcol	In which column there are the starting positions of the reads. Ignored if filetype = "bam".
endcol	In which column there are the end positions of the reads. Ignored if filetype = "bam".
idcol	In which column there are read identifiers. For single-end data it is optionally. For paired-end data it is required for some functionalities. The two reads of one pair need to have the same ID. Ignored if filetype = "bam" (the ID column is automatically included then). If read IDs include "#0/1" and "#0/2" in the end (indicating read 1 and read 2 of a pair), those characters will be removed from the IDs.
zerobased	if TRUE, start coordinates in readsfile are assumed to be 0-based and are then converted to 1-based system by adding 1. If FALSE, coordinates are not shifted. In this case they should already be 1-based in readsfile. Ignored if filetype = "bam", since scanBam converts 0-based to 1-based coordinates.
sep	Column separator character, defaults to tabs. Ignored if filetype = "bam".
skip	Number of lines of the bedfile to skip before beginning to read data; defaults to 1. Ignored if filetype = "bam".
header	A logical value indicating whether the file contains the names of the variables as its first line; defaults to FALSE. Ignored if filetype = "bam".
...	Further arguments passed to read.delim. Ignored if filetype = "bam".

Value

A RangedData table holding the read positions

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

See Also

get.targets

Examples

```
exptPath <- system.file("extdata", package="TEQC")
readsfile <- file.path(exptPath, "ExampleSet_Reads.bed")
reads <- get.reads(readsfile, idcol=4, skip=0)
```

`get.targets` *Read capture target positions*

Description

Reads a bedfile containing positions of the capture targets and creates a `RangedData` object.

Usage

```
get.targets(targetsfile, chrcol = 1, startcol = 2, endcol = 3, zerobased = TRUE, sep = "\t", skip = 1, hea
```

Arguments

`targetsfile`	name of bedfile giving the positions of each target region
`chrcol`	in which column in the targets bedfile there is the chromosome information (chromosome information in the file should be in string format, e.g. "chrX")
`startcol`	in which column there are the starting positions of the targeted regions
`endcol`	in which column there are the end positions of the targeted regions
`zerobased`	if `TRUE`, start coordinates in `targetsfile` are assumed to be 0-based and are then converted to 1-based system by adding 1. If `FALSE`, coordinates are not shifted. In this case they should already be 1-based in `targetsfile`.
`sep`	column separator character, defaults to tabs
`skip`	number of lines of the bedfile to skip before beginning to read data; defaults to 1
`header`	a logical value indicating whether the file contains the names of the variables as its first line; defaults to FALSE
`...`	further arguments passed to `read.delim`

Value

A `RangedData` table holding the target region positions. Note that overlapping or adjacent regions are collapsed to one region.

Note

Since overlapping regions are collapsed, the input bedfile can also contain positions of the (in most cases overlapping) hybridization probes used for the target capture.

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

See Also

`get.reads`

Examples

```
exptPath <- system.file("extdata", package="TEQC")
targetsfile <- file.path(exptPath, "ExampleSet_Targets.bed")
targets <- get.targets(targetsfile, skip=0)
```

`insert.size.hist` *Insert sizes histogram*

Description

Computes read pair insert sizes, i.e. distance from first base of first read to last base of second read of a read pair, and plots a histogram for all insert sizes.

Usage

```
insert.size.hist(readpairs, returnInserts = FALSE, legendpos="topleft", outline=FALSE, main, xlab, yla
```

Arguments

`readpairs`	`RangedData` table containing positions of read pairs, i.e. output of `reads2pairs` (or the element `readpairs` from the `reads2pairs` output in case single reads without matching pair were found).
`returnInserts`	if `TRUE`, the vector of read pair insert sizes is returned
`legendpos`	position of the legend, e.g. 'topleft' or 'topright'
`outline`	if `FALSE`, outliers (according to `boxplot.stats`) are removed before plotting. If `returnInserts=TRUE`, those outliers are still included in the output.
`main`	plot title
`xlab, ylab`	x- and y-axis labels
`breaks`	e.g. integer specifying the number of cells for the histogram, see `?hist`
`col`	histogram color
`...`	further graphical parameters passed to `hist`

Value

Histogram of read pair insert sizes. Average, standard deviation and median insert size are given in the legend and indicated by lines.

If `returnInserts = TRUE`, a named vector of insert sizes is returned.

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

See Also

`get.reads`, `reads2pairs`, `duplicates.barplot`

Examples

```
## get reads
exptPath <- system.file("extdata", package="TEQC")
readsfile <- file.path(exptPath, "ExampleSet_Reads.bed")
reads <- get.reads(readsfile, idcol=4, skip=0)

## merge to read pairs
readpairs <- reads2pairs(reads)

## insert size histogram
insert.size.hist(readpairs, breaks=10)
```

`make.wigfiles` *Creates wiggle files with per-base coverages*

Description

Prepares wiggle files with (non-zero) per-base coverages for the upload and visualization with genome browsers

Usage

```
make.wigfiles(coverageAll, chroms, trackname = "Coverage", filename = "Coverage")
```

Arguments

`coverageAll`	`RleList` containing `Rle` vectors of per-base coverages for each chromosome, i.e. `coverageAll` output of `coverage.target`
`chroms`	vector of chromosome names for which to produce wiggle files; if missing wiggle files will be produced for all chromosomes on which there are reads
`trackname`	trackname for wiggle file header
`filename`	part of output wiggle file name. Respective chromosome number and '.wig' will be added

Details

Only non-zero coverages will be listed

Value

One or more wiggle files listing per-base (non-zero) read coverages

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

See Also

coverage.target, coverage.plot, covered.k, coverage.hist, coverage.uniformity, coverage.targetlength.plot

Examples

```
## get reads and targets
exptPath <- system.file("extdata", package="TEQC")
readsfile <- file.path(exptPath, "ExampleSet_Reads.bed")
reads <- get.reads(readsfile, idcol=4, skip=0)
targetsfile <- file.path(exptPath, "ExampleSet_Targets.bed")
targets <- get.targets(targetsfile, skip=0)

## calculate per-base coverages
Coverage <- coverage.target(reads, targets, perBase=TRUE)

## create wiggle files for read coverages on chromsomes 13 and 17
make.wigfiles(Coverage$coverageAll, chroms=c("chr13", "chr17"))
```

---

multiTEQCreport *Creates an html report for multiple samples*

Description

Creates an automated html report comparing TEQC analysis results of several samples

Usage

```
multiTEQCreport(singleReportDirs, samplenames, projectName = "", targetsName = "",
       referenceName = "", destDir = "multiTEQCreport", k = c(1, 2, 3, 5, 10, 20), figureFormat = c("jpeg
```

Arguments

singleReportDirs	string of directory names; output directories of function TEQCreport(), launched beforehand for each single sample
samplenames	names of the samples that will be used in tables and figures
projectName	descriptive name for the project / collection of samples; will be written on top of the html report
targetsName	descriptive name of the captured target; will be written on top of the html report
referenceName	descriptive name of the reference genome the reads were aligned against; will be written on top of the html report
destDir	directory where results and html documents shall be saved
k	integer vector of k-values for which to show fraction of target bases with coverage >= k in section 'Sensitivity'
figureFormat	format of the figures produced for the html report (besides pdf graphs)

Details

Before creating the html report for multiple samples, `TEQCreport` has to be run for each of the samples separately. The output directories of those analyses are the input for `multiTEQCreport`. While the creation of single-sample reports is time and memory intensive, `multiTEQCreport` finishes quickly, since it just collects and summarizes the results from the single analyses. The files for the multiple sample html report are created in `destDir`. The report can be viewed by opening `destDir`/index.html in a web browser. Images are saved in `destDir`/image.

Value

The function is invoked for its side effect

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

References

Hummel M, Bonnin S, Lowy E, Roma G. TEQC: an R-package for quality control in target capture experiments. Bioinformatics 2011; 27(9):1316-7.

See Also

`TEQCreport`

Examples

```
## get reads and targets
exptPath <- system.file("extdata", package="TEQC")
readsfile <- file.path(exptPath, "ExampleSet_Reads.bed")
reads <- get.reads(readsfile, skip=0, idcol=4)
targetsfile <- file.path(exptPath, "ExampleSet_Targets.bed")
targets <- get.targets(targetsfile, skip=0)

## simulated 2nd sample
r <- sample(length(reads), 0.1 * length(reads))
reads2 <- reads[-r,,drop=TRUE]

## Not run:
## create single-sample reports
TEQCreport(sampleName="Test Sample A", targetsName="Human Exome", referenceName="Human Genome",
          destDir="./reportA", reads=reads, targets=targets, genome="hg19")
TEQCreport(sampleName="Test Sample B", targetsName="Human Exome", referenceName="Human Genome",
          destDir="./reportB", reads=reads2, targets=targets, genome="hg19")

## create multi-sample report
multiTEQCreport(singleReportDirs=c("./reportA", "./reportB"), samplenames=c("Sample A","Sample B"),
       projectName="Test Project", targetsName="Human Exome", referenceName="Human Genome",
          destDir="./multiTEQCreport")

## End(Not run)
```

`reads2pairs` *Merges reads to read pairs*

Description

Combines the two reads of a read pair (in case of paired-end data) to a new 'range' starting at the first reads's start position and ending at the second read's end position.

Usage

```
reads2pairs(reads, max.distance)
```

Arguments

`reads`	`RangedData` table containing positions of sequenced reads, i.e. output of `get.reads`. The first 'values' column has to contain read pair identifiers, i.e. when `reads` was created by `get.reads`, the option `idcol` had to be specified. The input can also contain single reads without 'read mate' (e.g. when the first read of a pair did not align to the reference genome, however the second one did align and was still kept). Those single reads will be returned in a separate table `singleReads`. When the two reads in a pair align to different chromosomes, they will also be returned in table `singleReads`.
`max.distance`	Integer value defining the maximum allowed distance between two reads of a pair (from start position of first read to end position of second read). Reads exceeding this distance will be returned in the separate table `singleReads`. If `max.distance` is not specified, reads will be joined to pairs regardless of their distance. Only when the two reads in a pair align to different chromosomes, they will be removed in any case and added to table `singleReads`.

Details

The function puts together the two reads of each pair and creates new ranges spanning both reads and everything in between. Those ranges correspond to the extent of the actual DNA molecules for which both ends were sequenced. The output of the function can be used by several other functions, whenever calculations should be based on read pairs rather than on single reads, e.g. `fraction.reads.target`, `readsPerTarget`, `duplicates.barplot`

Value

If `reads` only contains complete read pairs and for all pairs the respective reads align to the same chromosome and their distances do not exceed `max.distance` (if specified), a `RangedData` object is returned containing positions of the merged reads per pair, ranging from start position of the first read to end position of the second read.

If `reads` also contains single reads, or if reads within a pair are further apart than `max.distance` (if specified) or align to different chromosome, a list is returned with elements

singleReads	RangedData object containing original positions of single reads without 'read mates' and/or read pairs aligning too far apart or on different chromosomes
readpairs	RangedData object containing positions of the merged reads per pair, ranging from start position of the first read to end position of the second read

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

See Also

get.reads, fraction.reads.target, readsPerTarget, duplicates.barplot, insert.size.hist

Examples

```
exptPath <- system.file("extdata", package="TEQC")
readsfile <- file.path(exptPath, "ExampleSet_Reads.bed")
reads <- get.reads(readsfile, idcol=4, skip=0)
readpairs <- reads2pairs(reads)
```

---

readsPerTarget *Numbers of reads per target*

Description

Counts the numbers of reads overlapping each target region

Usage

```
readsPerTarget(reads, targets, Offset = 0)
```

Arguments

reads	RangedData table containing positions of sequenced reads, i.e. output from get.reads
targets	RangedData table containing positions of target regions, i.e. output from get.targets
Offset	integer; add Offset bases on both sides to targeted regions and potentially collapse resulting overlapping target regions

Value

The input RangedData table targets with an additional 'values' column containing numbers of reads overlapping each target

Note

As reads input also the mappingReads output of function fraction.reads.target can be used to speed up calculation. In this case, make sure that targets and Offset parameters were the same in fraction.reads.target as then specified in readsPerTarget.

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

See Also

coverage.target, fraction.reads.target, covered.k, coverage.hist, coverage.uniformity, coverage.plot, coverage.targetlength.plot

Examples

```
## get reads and targets
exptPath <- system.file("extdata", package="TEQC")
readsfile <- file.path(exptPath, "ExampleSet_Reads.bed")
reads <- get.reads(readsfile, idcol=4, skip=0)
targetsfile <- file.path(exptPath, "ExampleSet_Targets.bed")
targets <- get.targets(targetsfile, skip=0)

## number of reads per target
readsPerTarget(reads, targets)
```

TEQCreport *Creates an html report*

Description

Creates an automated html report for the complete TEQC analysis of one sample

Usage

```
TEQCreport(sampleName = "", targetsName = "", referenceName = "", destDir = "TEQCreport",
        reads = get.reads(), targets = get.targets(), Offset = 0, pairedend = FALSE, genome = c(NA, "hg3
        genomesize, k=c(1, 2, 3, 5, 10, 20), covthreshold = 8, CovUniformityPlot = FALSE, CovTargetLengt
        duplicatesPlot = FALSE, baits = get.baits(), WigFiles = FALSE, saveWorkspace = FALSE, figureFor
```

Arguments

sampleName	descriptive sample name; will be written on top of the html report
targetsName	descriptive name of the captured target; will be written on top of the html report
referenceName	descriptive name of the reference genome the reads were aligned against; will be written on top of the html report
destDir	directory where results and html documents shall be saved
reads	RangedData table containing positions of sequenced reads, or call to get.reads to read in positions from a bed or BAM file
targets	RangedData table containing positions of target regions, or call to get.targets to read in positions from a bed file

Offset	integer; add Offset bases on both sides to targeted regions and potentially collapse resulting overlapping target regions
pairedend	if TRUE, data will be considered to be paired-end data, i.e. reads will be "merged" to read pairs, and chromosome bar plot, specificity, enrichment and duplicate analysis (if selected) will be based on read pairs rather than on single reads
genome	genome version targets were designed and reads aligned to. For the given options the total genome size is set automatically. For other genomes or versions, leave this option empty ('NA') and specify the genome size with option 'genomesize'
genomesize	integer: specify the total genome size manually. If 'genomesize' is given, option 'genome' will be ignored.
k	integer vector of k-values for which to show fraction of target bases with coverage >= k in 'Sensitivity' table. Passed to covered.k
covthreshold	integer indicating which coverage value shall be highlighted by dashed lines in the coverage histogram. Passed to coverage.hist
CovUniformityPlot	if TRUE, a coverage uniformity plot is created, see coverage.uniformity
CovTargetLengthPlot	if TRUE, coverage vs target length plots are created, see coverage.targetlength.plot
CovGCPlot	if TRUE, a coverage vs GC content plot is created, see coverage.GC
duplicatesPlot	if TRUE, a duplicates barplot is created, see duplicates.barplot
baits	A RangedData table holding the hybridization probe ("bait") positions and sequences, or call to get.baits to read in positions from a bed file. Only needed if CovGCPlot = TRUE.
WigFiles	if TRUE, wiggle files with per-base coverage are created for each chromosome
saveWorkspace	if TRUE, an R workspace with objects reads, targets and output of coverage.target and reads2pairs (in case pairedend = TRUE) are saved in destDir to be available for further analyses
figureFormat	format of the figures produced for the html report (besides pdf graphs)

Details

TEQC analysis is performed and files for an html report are created in destDir. The report can be viewed by opening destDir/index.html in a web browser. Images are saved in destDir/image. Wiggle files (in case WigFiles = TRUE) are saved in destDir/wiggle. A table with general target coverage statistics, a table with average coverage values per target, a table with cumulative fractions of targeted bases with certain coverage and the R workspace containing R objects for potential further analysis (in case saveWorkspace = TRUE) are saved in destDir.

Value

The function is invoked for its side effect

Author(s)

Manuela Hummel <m.hummel@dkfz.de>

References

Hummel M, Bonnin S, Lowy E, Roma G. TEQC: an R-package for quality control in target capture experiments. Bioinformatics 2011; 27(9):1316-1317

See Also

get.reads, get.targets, fraction.target, fraction.reads.target, coverage.target, readsPerTarget, reads2pairs, covered.k, coverage.hist, coverage.uniformity, coverage.targetlength.plot, coverage.GC, get.baits, make.wigfiles

Examples

```
## get reads and targets files
exptPath <- system.file("extdata", package="TEQC")
readsfile <- file.path(exptPath, "ExampleSet_Reads.bed")
targetsfile <- file.path(exptPath, "ExampleSet_Targets.bed")

## create report
## Not run:
TEQCreport(sampleName="Test Sample", targetsName="Human Exome", referenceName="Human Genome",
          destDir="report", reads=get.reads(readsfile, skip=0, idcol=4),
          targets=get.targets(targetsfile, skip=0), genome="hg19")
## End(Not run)
```

Index

∗ **file**
  get.baits, 20
  get.reads, 21
  get.targets, 23
  make.wigfiles, 25
∗ **hplot**
  chrom.barplot, 2
  coverage.correlation, 3
  coverage.density, 5
  coverage.GC, 6
  coverage.hist, 8
  coverage.plot, 9
  coverage.targetlength.plot, 12
  coverage.uniformity, 13
  duplicates.barplot, 16
  insert.size.hist, 24
∗ **math**
  coverage.target, 11
  covered.k, 15
  fraction.reads.target, 18
  fraction.target, 19
  reads2pairs, 28
  readsPerTarget, 29
∗ **misc**
  multiTEQCreport, 26
  TEQCreport, 30

boxplot.stats, *8*, *24*

chrom.barplot, 2
coverage.correlation, 3, *6*
coverage.density, *4*, 5, *9*, *14*
coverage.GC, 6, *31*, *32*
coverage.hist, *4*, *6*, *7*, 8, *10*, *12–15*, *26*, *30–32*
coverage.plot, *4*, *6*, *7*, 9, *9*, *12–15*, *26*, *30*
coverage.target, *4–10*, 11, *12–15*, *25*, *26*, *30–32*
coverage.targetlength.plot, *7*, *9*, *10*, 12, *12*, *14*, *15*, *26*, *30–32*
coverage.uniformity, *4*, *6*, *7*, *9*, *10*, *12*, 13, *13*, *15*, *26*, *30–32*
covered.k, *4*, *6*, *7*, *10*, *12–14*, 15, *26*, *30–32*

duplicates.barplot, 16, *24*, *28*, *29*, *31*

fraction.reads.target, 18, *18*, *20*, *28–30*, *32*
fraction.target, *18*, 19, *19*, *20*, *32*

get.baits, *7*, 20, *31*, *32*
get.reads, *2*, *3*, *11*, *16–19*, 21, *21*, *23*, *24*, *28–30*, *32*
get.targets, *2*, *9*, *11*, *16–22*, 23, *29*, *30*, *32*

insert.size.hist, 24, *29*

make.wigfiles, *10*, 25, *32*
multiTEQCreport, 26

RangedData, *2*, *7*, *9*, *11*, *12*, *16*, *18*, *19*, *21–24*, *28–31*
reads2pairs, *16–19*, *24*, 28, *31*, *32*
readsPerTarget, *12*, *13*, *28*, 29, *29*, *32*
Rle, *7–9*, *11*, *15*, *25*
RleList, *7–9*, *11*, *15*, *25*

scanBam, *21*, *22*

TEQCreport, *27*, 30