# Package 'QoRTs'

September 18, 2014

**Version** 0.0.17

**Date** 2014-07-24

**Title** Quality of RNA-seq Tool

**Author** Stephen Hartley, PhD

**Maintainer** Stephen Hartley <stephen.hartley@nih.gov>

**Depends** R (>= 3.0.2), methods

**Suggests** MASS, Cairo, DESeq2, edgeR, knitr, BiocStyle

**Description** An R toolset for organizing, visualizing, and analyzing RNA-Seq Quality-Control data.

**License** file LICENSE

**VignetteBuilder** knitr

**URL** www.EDITME.EDITME

**BugReports** www.EDITME.EDITME

## R topics documented:

---

`build.plotter` *Generating plotters*

---

## Description

Generating QC_Plotter objects, which can be used in many of the QoRT utilities to organize samples in various ways to allow for easy comparison and detection of consistent biases and artifacts.

## Usage

```
build.plotter.summary(res, plotter.params = list())

build.plotter.colorByGroup(res, plotter.params = list())

build.plotter.colorByLane(res, plotter.params = list())

build.plotter.highlightBySample(curr.sample,
                                res,
                                plotter.params = list(),
                                merge.offset.outgroup = TRUE)

build.plotter.highlightBySample.colorByLane(curr.sample,
                                res,
                                plotter.params = list(),
                                merge.offset.outgroup = TRUE)

build.plotter.colorByX(res, color.by.name,
                       color.by.title.name = color.by.name,
                       plotter.params = list())
```

## Arguments

| | |
|---|---|
| `res` | A `QoRT_QC_Results` object, created by `read.qc.results.data`. |
| `curr.sample` | A character string. For the sample highlight summary plots, this should be the sample.ID of the sample that is to be highlighted. |

`merge.offset.outgroup`
: (For advanced users) A logical value. For the sample highlight plots, determines whether the all lanes that do not include the current sample should be treated as a single "outgroup".

`plotter.params`
: (For advanced users) A named list. Allows you to specify colors, offsets, and other similar patterns. By default these will all be set to reasonable values, however, if you want more control over colors, line-transparency, point plotting characters, or similar, then you can specify a named list.

  Any parameters that are not specified in the `plotter.params` list will be left as default.

  Legal parameters are:

  - `"contrasting.colors"`: colors to use for contrast. By default these are set to a series of reasonably-contrasting colors. However, if you have too many different categories then it may be hard to tell some colors apart.
  - `"contrasting.pch"`: point types to use for contrast (see `pch` in graphical parameters). By default this is set to the basic point types, then following through the upper and lower case letters.
  - `"std.lines.lty"`: Line type to use for the "highlighted" bams. (see `lty` in graphical parameters)
  - `"std.lines.lwd"`: Line width to use for the "highlighted" bams. (see `lwd` in graphical parameters)
  - `"std.lines.alpha"`: Alpha transparency value to use on lines for the "highlighted" bams. Numeric value between 0 and 255.
  - `"std.lines.color"`: Color to use for the "highlighted" bams.
  - `"std.points.pch"`: Character to use for points for the "highlighted" bams. (see `pch` in graphical parameters)
  - `"std.points.alpha"`: Alpha transparency value to use on points for the "highlighted" bams. Numeric value between 0 and 255.
  - `"std.points.color"`: Color to use for the "highlighted" bams.
  - `"std.NVC.colors"`: A named list with elements named "A", "T","C", and "G", with each element specifying a color. The colors used to indicate each base for the "highlighted" bams in the nucleotide-rate-by-position plots.
  - `"alt.lines.lty"`: Line type to use for the "non-highlighted" bams. (see `lty` in graphical parameters)
  - `"alt.lines.lwd"`: Line width to use for the "non-highlighted" bams. (see `lwd` in graphical parameters)
  - `"alt.lines.alpha"`: Alpha transparency value to use on lines for the "non-highlighted" bams. Numeric value between 0 and 255.
  - `"alt.lines.color"`: Color to use for the "non-highlighted" bams.
  - `"alt.points.pch"`: Character to use for points for the "non-highlighted" bams. (see `pch` in graphical parameters)
  - `"alt.points.alpha"`: Alpha transparency value to use on points for the "non-highlighted" bams. Numeric value between 0 and 255.
  - `"alt.points.color"`: Color to use for the "non-highlighted" bams.
  - `"alt.NVC.colors"`: A named list with elements named "A", "T","C", and "G", with each element specifying a color. The colors used to indicate each base for the "non-highlighted" bams in the nucleotide-rate-by-position plots.

- "show.legend": DEPRECIATED. Currently nonfunctional.

color.by.name
: (For advanced users) A character string. (TODO: document functionality)

color.by.title.name
: (For advanced users) A character string. (TODO: document functionality)

### Value

A QoRT_Plotter reference object used to create QC summary plots. Depending on which plotter is used, samples/lane-bams can be organized by group, sample, lane, or any arbitrary variable found in the decoder.

---

## get.summary.table *Get summary data table*

---

### Description

Retrieves and compiles a summary data table.

### Usage

```
get.summary.table(res, debugMode)
```

### Arguments

| | |
|---|---|
| res | A QoRT_QC_Results object. |
| debugMode | Logical. If TRUE, debugging data will be printed to the console. |

### Details

Returns summary data in table form.

---

## makePlot.all.std *Generating all default plots*

---

### Description

Saves MANY compiled QC plots for the given dataset.

### Usage

```
makePlot.all.std(res, outfile.prefix = "./",
                      plotter.params = list(),
                      plot.device.name = "png",
                      plotting.device.params = list(),
                      debugMode,
                      ...)
```

### Arguments

| | |
|---|---|
| `res` | A `QoRT_QC_Results` object, created by `read.qc.results.data`. |
| `outfile.prefix` | A file prefix, used for all output files. Usually the directory to which you want all files to be written. |
| `plotter.params` | Additional parameters (advanced) used in creation of the Plotter objects. See build.plotter. |
| `plot.device.name` | The method to use to save plots. Can be one of:<br>• `"png"` for standard png compression,<br>• `"CairoPNG"` for png compression using package Cairo. Note that this requires the package Cairo. |
| `plotting.device.params` | A named list of parameters to be passed to the graphics device. For example:<br>• `"width = 2000"`<br>Reasonable values for height, width, and pointsize will be chosen by default. |
| `debugMode` | Logical. If TRUE, debugging data will be printed to the console. |
| `...` | Arguments to be passed to methods, such as graphical parameters (see `par`). |

### Details

Saves MANY compiled QC plots for the given dataset.

---

## `makePlot.chrom.type.rates` *Chromosome type rate plot*

---

### Description

Plots the number or percent of read-pairs falling on each type of chromosome.

### Usage

```
makePlot.chrom.type.rates(plotter,
                          plot.rates = TRUE,
                          chromosome.name.style = "UCSC",
                          exclude.autosomes = FALSE,
                          chrom.norm.factors = NULL,
                          custom.chromosome.style.def.function = NULL,
                          return.table = FALSE,
                          debugMode,
                          ...)
```

### Arguments

| | |
|---|---|
| `plotter` | A QoRT_Plotter object. |
| `plot.rates` | A logical value indicating whether to plot percent of the total (for each bam file), rather than read-counts. |
| `chromosome.name.style` | A string value indicating the style of the chromosome names, and also how to split up the categories. There are 4 legal options:<br>• "UCSC": The default. Chromosomes are named: "chr1,chr2,...,chrX,chrY,chrXY,chrM". There are 6 categories: autosome, X, Y, XY, mitochondrial, and other.<br>• "ENSEMBL": Chromosomes are named: "1,2,...,X,Y,XY,MT". There are 6 categories: autosome, X, Y, XY, mitochondrial, and other.<br>• "UCSC_WITH_ERCC": As UCSC, but there is an additional category, which contains all chromosomes that begin with the text "ERCC".<br>• "ENSEMBL_WITH_ERCC": As ENSEMBL, but there is an additional category, which contains all chromosomes that begin with the text "ERCC". |
| `chrom.norm.factors` | (Advanced users) |
| `exclude.autosomes` | A logical value indicating whether to exclude autosomes from the plot. |
| `custom.chromosome.style.def.function` | (For advanced users) If your chromosomes do not match any of the above styles, then you can set your own chromosome style by handing this option a function. The function must take one argument. When handed NULL, it must return a list of all legal categories. When handed a single chromosome name, it must return one of those categories. |
| `return.table` | A Logical value. If TRUE, the function will return a table containing the plotted data. |
| `debugMode` | Logical. If TRUE, debugging data will be printed to the console. |
| `...` | Arguments to be passed to methods, such as graphical parameters (see `par`). |

### Details

TODO!

### Value

By default, this function returns nothing. If the return.table parameter is TRUE, then it returns a data.frame with the data that was plotted.

---

## `makePlot.cigarLength.distribution` *Plot the length distribution of a given cigar operation*

---

### Description

Plots the length distribution of a given cigar operation

### Usage

```
makePlot.cigarLength.distribution(plotter, op,
                                  r2.buffer = NULL,
                                  perMillion = TRUE,
                                  log.x = FALSE,
                                  log.y = FALSE,
                                  debugMode,
                                  ...)
```

### Arguments

| | |
|---|---|
| `plotter` | A QoRT_Plotter object. |
| `op` | A character string naming which cigar op to plot. Must be one of the following:<br>• `"SoftClip"` Soft Clip (Cigar Op "S")<br>• `"HardClip"` Hard Clip (Cigar Op "H")<br>• `"Del"` Deletion from reference (Cigar Op "D")<br>• `"Ins"` Insertion from reference (Cigar Op "I")<br>• `"Pad"` Padding (Cigar Op "P")<br>• `"Splice"` Splice Junction (Cigar Op "N")<br>• `"Aln"` Aligned to reference (Cigar Op "M")<br>Note that makePlot.cigarOp.byCycle(plotter,"SoftClip") gives identical results as makePlot.clipping, although the default value for the `rate.per.million` argument is different. |
| `r2.buffer` | Buffer space to place between the plotting of read 1 and read 2. By default this will choose a reasonable value. |
| `perMillion` | A logical value indicating whether or not to scale the y axis to rate-per-million-reads, rather than rate-per-read. Some people may find the results more readable this way, even though the plots themselves will appear the same. |
| `log.x` | A logical value indicating whether or not to log-scale the x axis. |
| `log.y` | A logical value indicating whether or not to log-scale the y axis. |
| `debugMode` | Logical. If TRUE, debugging data will be printed to the console. |
| `...` | Arguments to be passed to methods, such as graphical parameters (see `par`). |

### Details

x-axis: Length of the cigar operation.

y-axis: Frequency of the given length.

---

## `makePlot.cigarOp.byCycle`

*Plot Cigar Operator Rate*

---

### Description

Plots the rate at which the given CIGAR operator occurs, by read cycle.

### Usage

```
makePlot.cigarOp.byCycle(plotter, op,
                         r2.buffer = NULL,
                         rate.per.million = TRUE,
                         debugMode,
                         ...)
```

### Arguments

| | |
|---|---|
| `plotter` | A QoRTs_Plotter object. |
| `op` | A character string naming which cigar op to plot. Must be one of the following:<br>• `"SoftClip"` Soft Clip (Cigar Op "S")<br>• `"HardClip"` Hard Clip (Cigar Op "H")<br>• `"Del"` Deletion from reference (Cigar Op "D")<br>• `"Ins"` Insertion from reference (Cigar Op "I")<br>• `"Pad"` Padding (Cigar Op "P")<br>• `"Splice"` Splice Junction (Cigar Op "N")<br>• `"Aln"` Aligned to reference (Cigar Op "M")<br>Note that makePlot.cigarOp.byCycle(plotter,"SoftClip") gives identical results as makePlot.clipping, although the default value for the `rate.per.million` argument is different. |
| `r2.buffer` | Buffer space to place between the plotting of read 1 and read 2. By default this will choose a reasonable value. |
| `rate.per.million` | A logical value indicating whether or not to scale the y axis to rate-per-million-reads, rather than rate-per-read. Some people may find the results more readable this way, even though the plots themselves will appear the same. |
| `debugMode` | Logical. If TRUE, debugging data will be printed to the console. |
| `...` | Arguments to be passed to methods, such as graphical parameters (see `par`). |

### Details

x-axis: The read cycle (ie. the base-pair position in the read).

y-axis: The rate at which the bases at the given read-cycle is clipped off.

---

## `makePlot.clipping` *Plot Alignment Clipping*

---

### Description

Plots the rate at which the aligner soft-clips off portions of aligned reads.

### Usage

```
makePlot.clipping(plotter, rate.per.million = FALSE,
                  r2.buffer = NULL,
                  debugMode,
                  ...)
```

### Arguments

| | |
|---|---|
| plotter | A QoRTs_Plotter object. |
| rate.per.million | A logical value indicating whether or not to scale the y axis to rate-per-million-reads, rather than rate-per-read. Some people may find the results more readable this way, even though the plots themselves will appear the same. |
| r2.buffer | Buffer space to place between the plotting of read 1 and read 2. By default this will choose a reasonable value. |
| debugMode | Logical. If TRUE, debugging data will be printed to the console. |
| ... | Arguments to be passed to methods, such as graphical parameters (see par). |

### Details

x-axis: The read cycle (ie. the base-pair position in the read).

y-axis: The rate at which the bases at the given read-cycle is clipped off.

---

## makePlot.dropped.rates
*Read Drop Plot*

---

### Description

Plots the rates at which reads are dropped from analysis for various causes.

### Usage

```
makePlot.dropped.rates(plotter, debugMode, ...)
```

### Arguments

| | |
|---|---|
| plotter | A QoRT_Plotter object. |
| debugMode | Logical. If TRUE, debugging data will be printed to the console. |
| ... | Arguments to be passed to methods, such as graphical parameters (see par). |

### Details

TODO!

---

`makePlot.gc` *Plot GC content*

---

### Description

Plots GC content.

### Usage

```
makePlot.gc(plotter, plot.medians = NULL, plot.means = TRUE,
        debugMode, ...)
```

### Arguments

| | |
|---|---|
| `plotter` | A QoRTs_Plotter object. |
| `plot.medians` | A logical value indicating whether or not to plot the median average GC content for each bam file at the bottom of the plot. Overrides `plot.means`. |
| `plot.means` | A logical value indicating whether or not to plot the mean average GC content for each bam file at the bottom of the plot. |
| `debugMode` | Logical. If TRUE, debugging data will be printed to the console. |
| `...` | Arguments to be passed to methods, such as graphical parameters (see `par`). |

### Details

x-axis: Percent of the read-pairs that is made up of G's or C's.

y-axis: Rate at which the given percent occurs.

---

`makePlot.gene.assignment.rates`
*Gene assignment rate plot*

---

### Description

Plots the rate at which reads are assigned into various categories.

### Usage

```
makePlot.gene.assignment.rates(plotter,  debugMode, ...)
```

### Arguments

| | |
|---|---|
| `plotter` | A QoRT_Plotter object. |
| `debugMode` | Logical. If TRUE, debugging data will be printed to the console. |
| `...` | Arguments to be passed to methods, such as graphical parameters (see `par`). |

### Details

TODO!

## makePlot.gene.cdf *Plot the cumulative gene diversity curve*

### Description

Plots the cumulative gene diversity curve

### Usage

```
makePlot.gene.cdf(plotter,
                  sampleWise = FALSE,
                  plot.intercepts = TRUE,
                  label.intercepts = FALSE,
                  debugMode, ...)
```

### Arguments

| | |
|---|---|
| plotter | A QoRT_Plotter object. |
| sampleWise | A logical value indicating whether to compile each sample together across all runs. |
| plot.intercepts | A logical value indicating whether or not to plot the intercepts with the round numbers on the x-axis, in dotted lines. |
| label.intercepts | A logical value indicating whether or not to label the intercepts. No effect if plot.intercepts is FALSE. |
| debugMode | Logical. If TRUE, debugging data will be printed to the console. |
| ... | Arguments to be passed to methods, such as graphical parameters (see par). |

### Details

TODO!

## makePlot.genebody.coverage *Plot Gene-Body coverage distribution*

### Description

Plots Gene-Body coverage distribution

### Usage

```
makePlot.genebody.coverage(plotter, plot.medians,
                    plot.means = TRUE,
                    debugMode, ...)
makePlot.genebody.coverage.UMQuartile(plotter, plot.medians,
                    plot.means = TRUE,
                    debugMode, ...)
makePlot.genebody.coverage.lowExpress(plotter, plot.medians,
                    plot.means = TRUE,
                    debugMode, ...)
```

### Arguments

| | |
|---|---|
| `plotter` | A QoRTs_Plotter object. |
| `plot.medians` | A logical value indicating whether or not to plot the median average for each bamfile at the bottom of the plot. Overrides `plot.means`. |
| `plot.means` | A logical value indicating whether or not to plot the mean average for each bamfile at the bottom of the plot. |
| `debugMode` | Logical. If TRUE, debugging data will be printed to the console. |
| `...` | Arguments to be passed to methods, such as graphical parameters (see `par`). |

### Details

x-axis: Gene-body quantile. By default this is broken up into 40 quantiles containing 2.5

y-axis: Rate at which reads falls into the given quantile of the genes' bodies.

INCOMPLETE!!!!!

---

## `makePlot.insert.size`

*Plot Insert Size Distribution.*

---

### Description

Plots the distribution of the size of the region covered by the two paired reads.

### Usage

```
makePlot.insert.size(plotter, calc.rate = TRUE, pct.cutoff = 0.98,
                plot.medians = TRUE, plot.means = NULL,
                debugMode,
                ...)
```

### Arguments

| | |
|---|---|
| `plotter` | A QoRTs_Plotter object. |
| `calc.rate` | A logical value indicating whether or not to calculate and plot the rate at which each insert size occurs as the y-axis. If this is set to false, it will instead plot the total number of times each insert size occurs. |

| | |
|---|---|
| pct.cutoff | A numeric value between 0 and 1, indicating the quantile within which to limit the x-axis. Generally the default value of 0.98 is perfectly usable. |
| plot.means | A logical value indicating whether or not to plot the mean average for each bamfile at the bottom of the plot. |
| plot.medians | A logical value indicating whether or not to plot the median average for each bamfile at the bottom of the plot. Overrides plot.means. |
| debugMode | Logical. If TRUE, debugging data will be printed to the console. |
| ... | Arguments to be passed to methods, such as graphical parameters (see par). |

## Details

x-axis: The insert size. This is the genomic distance from the start of one read to the end of the other. In other words, the size of the full region covered by the paired reads.

y-axis: The rate at which the given insert size occurs, for each bamfile.

Note: The insert size is calculated by using the alignment as well as the provided gene/splice-junction annotation.

1. If the paired reads align to overlapping regions of the reference genome, then the insert size can be calculated exactly. This is NOT dependant on the annotation. If the two reads align inconsistantly (for example, one read showing a splice junction and the other not), then the read is ignored.
2. If the paired reads do NOT overlap, then the annotation information is required. Using the full set of all known splice junctions that lie between the inner alignment endpoints for the two reads, the shortest possible path from the two endpoints is found. In some rare cases this may underestimate the insert size, if the actual insert does not take the minimal path, but this is rare. In other (somewhat more common) cases this may overestimate the insert size, when the region between the paired reads bridges an unannotated splice junction.

---

# makePlot.legend.box

*Plot legend*

---

## Description

Plots the universal legend for a given plotter object.

## Usage

```
makePlot.legend.box(plotter, debugMode, ...)
makePlot.legend.over(position, plotter, debugMode, ...)
```

## Arguments

| | |
|---|---|
| plotter | A QoRT_Plotter object. |
| position | For makePlot.legend.over, a character string indicating the location where you want the legend to appear. This is passed to legend, and can be any keyword allowed by xy.coords. For example: "top", "topleft","bottomright", etc. |
| debugMode | Logical. If TRUE, debugging data will be printed to the console. |
| ... | Arguments to be passed to methods, such as graphical parameters (see par). |

### Details

makePlot.legend.box creates a new plot (opening the next graphics frame), and writes a small description of the given plotter type along with plotting a color-coded legend (if applicable).

makePlot.legend.over adds a legend to the current plotting frame.

---

## makePlot.mapping.rates
*Plot mapping rates*

---

### Description

Plots the rates at which reads map to the genome.

### Usage

```
makePlot.mapping.rates(plotter, plot.mm = TRUE,
                   y.counts.in.millions = TRUE,
                   debugMode,
                   ...)
```

### Arguments

| | |
|---|---|
| plotter | A QoRT_Plotter object. |
| plot.mm | Plot multi-mapping rates. |
| y.counts.in.millions | Label/scale the y-axis in millions of reads. |
| debugMode | Logical. If TRUE, debugging data will be printed to the console. |
| ... | Arguments to be passed to methods, such as graphical parameters (see par). |

### Details

TODO!

---

## makePlot.missingness.rate
*Plot N-Rate by read cycle*

---

### Description

Plots rate by which the sequencer cannot determine the base, by read cycle.

### Usage

```
makePlot.missingness.rate(plotter, r2.buffer = NULL,
                      debugMode, ...)
```

### Arguments

| | |
|---|---|
| plotter | A QoRT_Plotter object. |
| r2.buffer | Buffer space to place between the plotting of read 1 and read 2. By default this will choose a reasonable value. |
| debugMode | Logical. If TRUE, debugging data will be printed to the console. |
| ... | Arguments to be passed to methods, such as graphical parameters (see par). |

### Details

x-axis: Read Cycle

y-axis: Rate at which the sequencer assigns an 'N' base.

---

## makePlot.norm.factors

*Plot Alignment Clipping*

---

### Description

Plots the rate at which the aligner soft-clips off portions of aligned reads.

### Usage

```
makePlot.norm.factors(plotter, by.sample = TRUE,
                      return.table = FALSE,
                      debugMode, ...)
makePlot.norm.factors.vs.TC(plotter,
                      by.sample = TRUE,
                      return.table = FALSE,
                      debugMode, ...)
```

### Arguments

| | |
|---|---|
| plotter | A QoRTs_Plotter object. |
| by.sample | Logical. Whether to combine all lanebams for each sample before calculating normalization factors. By default, normalization factors for each lanebam will be calculated seperately. |
| return.table | Logical. Whether to return a data table containing the data that is plotted. |
| debugMode | Logical. If TRUE, debugging data will be printed to the console. |
| ... | Arguments to be passed to methods, such as graphical parameters (see par). |

### Details

makePlot.norm.factors plots the normalization factors for each sample/lanebam. Note that unless DESeq2 and edgeR are installed, it will only plot total-count normalization by default. Also note that unless calc.DESeq2 = TRUE and/or calc.edgeR = TRUE in the execution of the read.qc.results.data that produced the QC results, only the total counts normalization factors will be calculated.

makePlot.norm.factors.vs.TC plots the ratio of alternative normalization factors against the total count normalization.

### Value

Usually returns nothing, unless return.table is TRUE, in which case it returns a data.frame containing the plotted data for each sample.

---

`makePlot.NVC.clip.matchByClipPosition`
*Plot clipped nucleotide rates, aligned by the distance from the point of clipping.*

---

### Description

WARNING: THESE FUNCTIONS ARE BETA, AND ARE NOT FULLY TESTED OR READY FOR GENERAL USE.

### Usage

```
makePlot.NVC.lead.clip.matchByClipPosition(plotter,
            clip.amt,  r2.buffer,
            label.majority.bases = TRUE,
            label.majority.bases.threshold = 0.5,
            label.majority.bases.cex = 1,
            show.base.legend = TRUE,
            load.results = TRUE,
            debugMode, ...)

makePlot.NVC.tail.clip.matchByClipPosition(plotter,
            clip.amt, r2.buffer,
            label.majority.bases = TRUE,
            label.majority.bases.threshold = 0.5,
            label.majority.bases.cex = 1,
            show.base.legend = TRUE,
            debugMode, ...)
```

### Arguments

| | |
|---|---|
| `plotter` | A QoRT_Plotter object. |
| `clip.amt` | UNDOCUMENTED |
| `r2.buffer` | UNDOCUMENTED |
| `label.majority.bases` | UNDOCUMENTED |
| `label.majority.bases.threshold` | UNDOCUMENTED |
| `label.majority.bases.cex` | UNDOCUMENTED |
| `show.base.legend` | UNDOCUMENTED |
| `load.results` | UNDOCUMENTED |
| `debugMode` | UNDOCUMENTED |
| `...` | Arguments to be passed to methods, such as graphical parameters (see `par`). |

### Details

TODO!

---

`makePlot.NVC.lead.clip`
*Clipped sequence nucleotide-by-position plot*

---

### Description

Plots the nucleotide rates for clipped segments

### Usage

```
makePlot.NVC.lead.clip(plotter, clip.amt,  r2.buffer,
                   points.highlighted = TRUE,
                   label.majority.bases = TRUE,
                   label.majority.bases.threshold = 0.5,
                   label.majority.bases.cex = 1,
                   show.base.legend = TRUE,
                   debugMode,
                   ...)


makePlot.NVC.tail.clip(plotter, clip.amt,  r2.buffer,
                       points.highlighted = TRUE,
                       label.majority.bases = TRUE,
                       label.majority.bases.threshold = 0.5,
                       label.majority.bases.cex = 1,
                       show.base.legend = TRUE,
                       debugMode, ...)
```

### Arguments

| | |
|---|---|
| `plotter` | A QoRT_Plotter object. |
| `clip.amt` | Numeric value. The number of bases clipped. These methods only plot the sequence for a single specific clip.amt. |
| `r2.buffer` | Buffer space to place between the plotting of read 1 and read 2. By default this will choose a reasonable value. |
| `points.highlighted` | A logical value. Determines whether to ever plot points on top of the lines. This can be useful for identifying outliers. If TRUE, then all highlighted lane-bams will be overlaid with points using their designated pch symbol. If the plotter does not highlight any lanebams, then points will be overlaid on ALL lines. |
| `label.majority.bases` | A logical value indicating whether to label the genotypes of read cycles in which the most common base has a frequency exceeding label.majority.bases.threshold (see below). |
| `label.majority.bases.threshold` | A numeric value indicating the cutoff above which the most common base will be labelled, if label.majority.bases is set TRUE (see above). |

label.majority.bases.cex
: The cex value fed to text() that is used to determine how big the labels are to be, if label.majority.bases is TRUE. (see par for information on cex).

show.base.legend
: A logical value indicating whether to print the base-color legend along the right edge of the plot.

debugMode
: Logical. If TRUE, debugging data will be printed to the console.

...
: Arguments to be passed to methods, such as graphical parameters (see par).

### Details

TODO!

---

## makePlot.qual.pair *Plot quality score by read cycle*

---

### Description

Plots the Phred quality score as a function of the read cycle for both reads.

### Usage

```
makePlot.qual.pair(plotter, y.name, r2.buffer = NULL,
                debugMode, ...)
```

### Arguments

plotter
: A QoRTs_Plotter object.

y.name
: The name of the quality score metric to plot. Must be one of:
  - "min"
  - "lowerQuartile"
  - "median"
  - "upperQuartile"
  - "max"

r2.buffer
: Buffer space to place between the plotting of read 1 and read 2. By default this will choose a reasonable value.

debugMode
: Logical. If TRUE, debugging data will be printed to the console.

...
: Arguments to be passed to methods, such as graphical parameters (see par).

### Details

These plots display information about the phred quality score (y-axis) as a function of the position in the read (x-axis). Five statistics can be plotted: minimum, maximum, upper and lower quartiles, and median. These statistics are calculated individually for each bam file and each read position (ie, each plotted line corresponds to a bam file).

Note that the Phred score is always an integer, and as such these plots would normally be very difficult to read because lines would be plotted directly on top of one another. To reduce this problem, the lines are vertically offset from one another. Most plotters offset each libe by lane.ID.

MORE DETAILS!

`makePlot.raw.NVC` *Nucleotide-rate by read cycle plot*

## Description

Plots the nucleotide rate by read cycle

## Usage

```
makePlot.raw.NVC(plotter,  r2.buffer = NULL,
                         points.highlighted = TRUE,
                         label.majority.bases = FALSE,
                         label.majority.bases.threshold = 0.5,
                         label.majority.bases.cex = 0.5,
                         show.base.legend = TRUE,
                         debugMode,
                         ...)

makePlot.minus.clipping.NVC(plotter,  r2.buffer = NULL,
                         points.highlighted = TRUE,
                         label.majority.bases = FALSE,
                         label.majority.bases.threshold = 0.5,
                         label.majority.bases.cex = 0.5,
                         show.base.legend = TRUE,
                         debugMode,
                         ...)
```

## Arguments

| | |
|---|---|
| `plotter` | A QoRT_Plotter object. |
| `r2.buffer` | Buffer space to place between the plotting of read 1 and read 2. By default this will choose a reasonable value. |
| `points.highlighted` | A logical value. Determines whether to ever plot points on top of the lines. This can be useful for identifying outliers. If TRUE, then all highlighted lane-bams will be overlaid with points using their designated pch symbol. If the plotter does not highlight any lanebams, then points will be overlaid on ALL lines. |
| `label.majority.bases` | A logical value indicating whether to label the genotypes of read cycles in which the most common base has a frequency exceeding label.majority.bases.threshold (see below). |
| `label.majority.bases.threshold` | A numeric value indicating the cutoff above which the most common base will be labelled, if label.majority.bases is set TRUE (see above). |
| `label.majority.bases.cex` | The cex value fed to text() that is used to determine how big the labels are to be, if label.majority.bases is TRUE. (see `par` for information on cex). |
| `show.base.legend` | A logical value indicating whether to print the base-color legend along the right edge of the plot. |

| | |
|---|---|
| debugMode | Logical. If TRUE, debugging data will be printed to the console. |
| ... | Arguments to be passed to methods, such as graphical parameters (see par). |

## Details

TODO!

---

`makePlot.splice.junction.event.rates`
*Plot Splice Junction Event Rates*

---

## Description

Plots the rates at which splice junctions occur.

## Usage

```
makePlot.splice.junction.event.rates(plotter,
                                     calc.ratePerRead = TRUE,
                                     calc.proportion = FALSE,
                                     high.low.cutoff = 4,
                                     debugMode = DEFAULTDEBUGMODE,
                                     ...)

makePlot.splice.junction.event.rates.split.by.type(plotter,
                                     calc.rate = TRUE,
                                     high.low.cutoff = 4,
                                     debugMode = DEFAULTDEBUGMODE,
                                     ...)
```

## Arguments

| | |
|---|---|
| plotter | A QoRT_Plotter object. |
| calc.ratePerRead | TODO!!!!! |
| calc.proportion | TODO!!!!! |
| high.low.cutoff | TODO!!!!! |
| debugMode | Activates debug mode, which causes more verbose reporting. |
| calc.rate | TODO!!!!! |
| ... | Arguments to be passed to methods, such as graphical parameters (see par). |

## Details

TODO!

## makePlot.splice.junction.loci.counts
*Splice Junction Loci Count Plot*

### Description

Plots the rate at which splice junction loci fall into various categories.

### Usage

```
makePlot.splice.junction.loci.counts(plotter,
                                     calc.rate = FALSE,
                                     high.low.cutoff = 4,
                                     debugMode, ...)
```

### Arguments

| | |
|---|---|
| plotter | A QoRT_Plotter object. |
| calc.rate | Logical. If TRUE, the proportion of loci in each category will be calculated and plotted, rather than the raw number. |
| high.low.cutoff | (For advanced users only!) The cutoff between "high" and "low" expression junction loci. Note that this must match the cutoff used by the jarfile QC execution. |
| debugMode | Logical. If TRUE, debugging data will be printed to the console. |
| ... | Arguments to be passed to methods, such as graphical parameters (see par). |

### Details

TODO!

## makePlot.strandedness.test
*Strandedness Test Plot*

### Description

Plots the apparent strandedness of the reads.

### Usage

```
makePlot.strandedness.test(plotter, plot.target.boxes = FALSE,
                           debugMode, ...)
```

### Arguments

| | |
|---|---|
| `plotter` | A QoRT_Plotter object. |
| `plot.target.boxes` | A logical value. If true, then green target boxes will be printed over the area in which all points should be expected to fall. |
| `debugMode` | Logical. If TRUE, debugging data will be printed to the console. |
| `...` | Arguments to be passed to methods, such as graphical parameters (see `par`). |

### Details

TODO!

---

## `makePlot.summary` *Generating summary plots*

---

### Description

Creates a large multi-frame summary plot, or a report PDF file.

### Usage

```
makePlot.summary.basic(res, outfile.prefix = NULL,
                    outfile.suffix = "plot.full.summary.png",
                    plotter.params = list(),
                    plot.device.name,
                    plotting.device.params = list(),
                    verbose = TRUE,
                    debugMode,
                    ...)

makePlot.summary.colorByGroup(res, outfile.prefix = NULL,
                    outfile.suffix = "plot.colorByGroup.png",
                    plotter.params = list(),
                    plot.device.name,
                    plotting.device.params = list(),
                    verbose = TRUE,
                    debugMode,
                    ...)

makePlot.summary.colorByLane(res, outfile.prefix = NULL,
                    outfile.suffix = "plot.colorByLane.png",
                    plotter.params = list(),
                    plot.device.name,
                    plotting.device.params = list(),
                    verbose = TRUE,
                    debugMode,
                    ...)

makePlot.summary.sample.highlight(res,
                    curr.sample,
```

```
                        outfile.prefix = NULL,
                        outfile.suffix,
                        plotter.params = list(),
                        plot.device.name,
                        plotting.device.params = list(),
                        verbose = TRUE,
                        debugMode,
                        ...)

  makePlot.summary.sample.highlight.andColorByLane(res,
                        curr.sample,
                        outfile.prefix = NULL,
                        outfile.suffix,
                        plotter.params = list(),
                        plot.device.name,
                        plotting.device.params = list(),
                        verbose = TRUE,
                        debugMode,
                        ...)
```

## Arguments

| | |
|---|---|
| `res` | A `QoRT_QC_Results` object, created by `read.qc.results.data`. |
| `curr.sample` | A character string. For the sample highlight summary plots, this should be the sample.ID of the sample that is to be highlighted. |
| `outfile.prefix` | A file prefix, used for all output files. Usually the file directory to which you want all files to be written. If left as the default value, then it will attempt to plot to the default device rather than to file (for most setups, when running R in interactive mode this will make the plot pop up as a seperate window). |
| `outfile.suffix` | A file suffix. This will be set to a descriptively-named string ending with ".png", by default. |
| `plotter.params` | Additional parameters (advanced) used in creation of the Plotter objects. See build.plotter. |
| `plot.device.name` | The method to use to save plots. Can be one of:<br>• `"png"` for standard png compression,<br>• `"CairoPNG"` for png compression using package Cairo. Note that this requires the package Cairo.<br>• `"pdf"` for a multi-page pdf report.<br>• `"CairoPDF"` for a multi-page pdf report using package Cairo. Note that this requires the package Cairo. |
| `plotting.device.params` | A named list of parameters to be passed to the graphics device. For example:<br>• `"width = 2000"`<br>Reasonable values for height, width, and pointsize will be chosen by default. |
| `verbose` | Logical. If TRUE, more information will be printed to the console. |

| | |
|---|---|
| debugMode | Logical. If TRUE, debugging data will be printed to the console. |
| ... | Arguments to be passed to methods, such as graphical parameters (see par). |

### Details

WIP, Documentation Incomplete!

---

## makePlot.summary.sample.highlight.all
*Generating sample highlight plots*

---

### Description

Generates multiple sample highlight summary plots, one for every sample.

### Usage

```
makePlot.summary.sample.highlight.all(res,
                    outfile.prefix = "./",
                    plotter.params = list(),
                    plot.device.name = "png",
                    plotting.device.params = list(),
                    verbose = TRUE, debugMode,
                    ...)

makePlot.summary.sample.highlight.andColorByLane.all(res,
                    outfile.prefix = "./",
                    plotter.params = list(),
                    plot.device.name = "png",
                    plotting.device.params = list(),
                    verbose = TRUE, debugMode,
                    ...)
```

### Arguments

| | |
|---|---|
| res | A QoRT_QC_Results object, created by read.qc.results.data. |
| outfile.prefix | A file prefix, used for all output files. Usually the directory to which you want all files to be written. |
| plotter.params | Additional parameters (advanced) used in creation of the Plotter objects. See build.plotter. |
| plot.device.name | The method to use to save plots. Can be one of:<br>• "png" for standard png compression,<br>• "CairoPNG" for png compression using package Cairo. Note that this requires the package Cairo. |
| plotting.device.params | A named list of parameters to be passed to the graphics device. For example:<br>• "width = 2000" |

| | |
|---|---|
| | Reasonable values for height, width, and pointsize will be chosen by default. |
| verbose | Logical. If TRUE, more info will be printed to the console. |
| debugMode | Logical. If TRUE, debugging data will be printed to the console. |
| ... | Arguments to be passed to methods, such as graphical parameters (see par). |

## Details

WIP, Documentation Incomplete!

---

`read.qc.results.data` *Reading QC results data*

---

## Description

Creates a QoRT_QC_Results object using a set of QC result data files.

## Usage

```
read.qc.results.data(infile.dir,
                     decoder.files,
                     decoder.data.frame,
                     calc.DESeq2 = FALSE,
                     calc.edgeR = FALSE,
                     debugMode)
```

## Arguments

| | |
|---|---|
| infile.dir | REQUIRED. The base file directory where all the QC results data is stored. |
| decoder.files | Either a character string or a character vector, containing the decoder.<br>Either decoder.files OR decoder.data.frame must be set. Not both. |
| decoder.data.frame | A data.frame containing the decoder information. |
| calc.DESeq2 | Logical. If TRUE, this function will attempt to load the DESeq2 package. If the DESeq2 package is found, it will then calculate DESeq2's geometric normalization factors (also known as "size factors") for each replicate and for each sample. |
| calc.edgeR | Logical. If TRUE, this function will attempt to load the edgeR package. If the edgeR package is found, it will then calculate all of edgeR's normalization factors (also known as "size factors") for each replicate and for each sample. |
| debugMode | Logical. If TRUE, debugging data will be printed to the console. |

**Details**

Single Decoder: The "decoder" is a file or data frame that describes each replicate. It must have one row per replicate, with the following columns:

- lanebam.ID: The base identifier for the individual replicate. Must be unique!
- lane.ID: The identifier for the lane, run, or batch.
- group.ID: The identifier for the biological condition for the given replicate.
- sample.ID: The identifier for the specific biological replicate from which the replicate belongs. Note that this is distinct from "lanebam.ID" because in many RNA-Seq studies each "sample" can have multiple technical replicates, as multiple sequencing runs may be needed to acquire sufficient reads for analysis.
- qc.data.dir: (OPTIONAL) This column indicates the subdirectory in which the replicate's QC data was written. If this column is missing, it is assumed to be equal to the lanebam.ID.
- input.read.pair.count: (OPTIONAL) This column contains the number of input reads, before alignment. This is used later to calculate mapping rate.
- multi.mapped.read.pair.count: (OPTIONAL) This column contains the number of reads that were multi-mapped. This must be included for multi-mapping rate to be calculated.
- cycle.CT: (OPTIONAL) This column contains the length of the reads for the replicate, in base-pairs. By default this is calculated automatically, so this column never needs to be set.

Dual Decoder: Alternatively, two decoders can be set. In this case the first decoder should be the lanebam decoder, and the second should be the sample decoder. The lanebam decoder should have one row per replicate, with the following columns:

- lanebam.ID: The base identifier for the individual replicate. Must be unique!
- lane.ID: The identifier for the lane, run, or batch.
- sample.ID: The identifier for the specific biological replicate from which the replicate belongs. Note that this is distinct from "lanebam.ID" because in many RNA-Seq studies each "sample" can have multiple technical replicates, as multiple sequencing runs may be needed to acquire sufficient reads for analysis.
- qc.data.dir: (OPTIONAL) This column indicates the subdirectory in which the replicate's QC data was written. If this column is missing, it is assumed to be equal to the lanebam.ID. Must be unique!
- input.read.pair.count: (OPTIONAL) This column contains the number of input reads, before alignment. This is used later to calculate mapping rate.
- multi.mapped.read.pair.count: (OPTIONAL) This column contains the number of reads that were multi-mapped. This must be included for multi-mapping rate to be calculated.
- cycle.CT: (OPTIONAL) This column contains the length of the reads for the replicate, in base-pairs. By default this is calculated automatically, so this column never needs to be set.

The sample decoder should have one row per replicate, with the following columns:

- group.ID: The identifier for the biological condition for the given replicate.
- sample.ID: The identifier for the specific biological replicate from which the replicate belongs. Note that this is distinct from "lanebam.ID" because in many RNA-Seq studies each "sample" can have multiple technical replicates, as multiple sequencing runs may be needed to acquire sufficient reads for analysis. Must be unique!

All decoders are allowed to contain other columns in addition to the ones listed here, so long as their names are distinct. Columns do not need to appear in any particular order, so long as they are named according to the specifications above.

# Index