Package ‘sangeranalyseR’

May 26, 2026

Type Package

Title sangeranalyseR: a suite of functions for the analysis of Sanger sequence data in R

Version 1.23.0

Date 2024-04-24

biocViews Genetics, Alignment, Sequencing, SangerSeq, Preprocessing, QualityControl, Visualization, GUI

Description This package builds on sangerseqR to allow users to create contigs from collections of Sanger sequencing reads. It provides a wide range of options for a number of commonly-performed actions including read trimming, detecting secondary peaks, and detecting indels using a reference sequence. All parameters can be adjusted interactively either in R or in the associated Shiny applications. There is extensive online documentation, and the package can outputs detailed HTML reports, including chromatograms.

License GPL-2 | file LICENSE

URL https://github.com/roblanf/sangeranalyseR

BugReports https://github.com/roblanf/sangeranalyseR/issues

Encoding UTF-8

Depends R (>= 4.0.0), Biostrings, DECIPHER, sangerseqR

Imports ape, BiocGenerics, BiocParallel, S4Vectors, data.table, DT, excelR, ggdendro, grDevices, graphics, gridExtra, logger, methods, openxlsx, parallel, plotly, pwalign, Rcpp, rmarkdown (>= 2.9), seqinr, shiny, shinycssloaders, shinydashboard, shinyjs, shinyWidgets, stats, stringr, tools, utils

LinkingTo Rcpp

RoxygenNote 7.3.2

VignetteBuilder knitr

Suggests testthat (>= 2.1.0), withr, BiocManager, BiocStyle, knitr (>= 1.33), reshape2, zeallot

Collate 'AllGenerics.R' 'ClassChromatogramParam.R' 'ClassObjectResults.R' 'ClassQualityReport.R' 'ClassSangerRead.R' 'ClassSangerAlignment.R' 'ClassSangerContig.R' 'Constructors.R' 'GlobalTrimApp.R'

'LoadMessage.R' 'MethodSangerAlignment.R'
'MethodSangerContig.R' 'MethodSangerRead.R' 'MethodShared.R'
'MethodsQualityReport.R' 'RcppExports.R'
'ShinySangerAlignmentServer.R' 'ShinySangerAlignmentUI.R'
'ShinySangerContigServer.R' 'ShinySangerContigUI.R'
'ShinyServerModule.R' 'UtilitiesFunc.R'
'UtilitiesFuncInputChecker.R' 'data.R'
'sangeranalyseR_package.R' 'sangeranalyseR_show_method.R'

git_url https://git.bioconductor.org/packages/sangeranalyseR

git_branch devel

git_last_commit f6375ae

git_last_commit_date 2026-05-05

Repository Bioconductor 3.24

Date/Publication 2026-05-25

Author Rob Lanfear [aut],
Kuan-Hao Chao [aut, cre]

Maintainer Kuan-Hao Chao <ntueeb05howard@gmail.com>

Contents

ChromatogramParam-class . . . 3
chromatogram_overwrite . . . 4
chromatogram_plotly . . . 5
generateReport . . . 6
generateReportSA . . . 7
generateReportSC . . . 8
generateReportSR . . . 9
globalTrimApp . . . 9
launchApp . . . 10
launchAppSA . . . 11
launchAppSC . . . 11
MakeBaseCalls . . . 12
ObjectResults-class . . . 12
primaryAASeqS1 . . . 13
primaryAASeqS2 . . . 14
primaryAASeqS3 . . . 14
qualityBasePlot . . . 15
QualityReport-class . . . 15
QualityReport-class-qualityBasePlot . . . 17
QualityReport-class-updateQualityParam . . . 17
qualityReportData . . . 18
readTable . . . 19
SangerAlignment . . . 19
SangerAlignment-class . . . 23
SangerAlignment-class-generateReportSA . . . 26
SangerAlignment-class-launchAppSA . . . 27
SangerAlignment-class-updateQualityParam . . . 27
SangerAlignment-class-writeFastaSA . . . 28
sangerAlignmentData . . . 29

sangeranalyseR 29
SangerContig 30
SangerContig-class 33
SangerContig-class-generateReportSC 36
SangerContig-class-launchAppSC 37
SangerContig-class-readTable 38
SangerContig-class-updateQualityParam 39
SangerContig-class-writeFastaSC 40
sangerContigData 41
SangerRead 41
SangerRead-class 43
SangerRead-class-generateReportSR 45
SangerRead-class-MakeBaseCalls 46
SangerRead-class-qualityBasePlot 47
SangerRead-class-readTable 48
SangerRead-class-updateQualityParam 48
SangerRead-class-writeFastaSR 49
sangerReadFData 50
updateQualityParam 51
writeFasta 52
writeFastaSA 53
writeFastaSC 54
writeFastaSR 54

Index **56**

`ChromatogramParam-class` *ChromatogramParam*

Description

An S4 class storing chromatogram related inputs in a SangerRead S4 object.

Slots

`baseNumPerRow` It defines maximum base pairs in each row. The default value is `100`.

`heightPerRow` It defines the height of each row in chromatogram. The default value is `200`.

`signalRatioCutoff` The ratio of the height of a secondary peak to a primary peak. Secondary peaks higher than this ratio are annotated. Those below the ratio are excluded. The default value is `0.33`.

`showTrimmed` The logical value storing whether to show trimmed base pairs in chromatogram. The default value is `TRUE`.

Author(s)

Kuan-Hao Chao

Examples

```
Chromatogram <- new("ChromatogramParam",
                    baseNumPerRow      = 100,
                    heightPerRow       = 200,
                    signalRatioCutoff  = 0.33,
                    showTrimmed        = TRUE)
```

---

chromatogram_overwrite

Static base-R chromatogram renderer with a corrected color palette.

Description

Reimplementation of `sangerseqR::chromatogram` with a fix for base color rendering. Intended for static (PDF / PNG) export. For interactive embedding in Shiny see `chromatogram_plotly`.

Usage

```
chromatogram_overwrite(
  obj,
  trim5 = 0,
  trim3 = 0,
  showcalls = c("primary", "secondary", "both", "none"),
  width = 100,
  height = 2,
  cex.mtext = 1,
  cex.base = 1,
  ylim = 3,
  filename = NULL,
  showtrim = FALSE,
  showhets = TRUE,
  colors = "default"
)
```

Arguments

`obj`	A sangerseq or SangerRead instance.
`trim5`	Integer; number of bases to mark as 5' trimmed.
`trim3`	Integer; number of bases to mark as 3' trimmed.
`showcalls`	One of `"primary"`, `"secondary"`, `"both"`, or `"none"`.
`width`	Bases per row.
`height`	Plot height per row (relative units).
`cex.mtext`	Text size for marginal annotations.
`cex.base`	Text size for base-call labels.
`ylim`	Maximum y-axis multiplier (relative to robust mean).
`filename`	Optional path to write a PDF.
`showtrim`	Logical; if TRUE, shade the trim regions.
`showhets`	Logical; if TRUE, mark heterozygous positions.
`colors`	Either `"default"`, `"cb_friendly"`, or a length-5 character vector of hex colours.

Value

Invisibly returns NULL; called for its side effect of plotting (or writing to `filename`).

Examples

```
data(sangerReadFData)

chromatogram_overwrite(sangerReadFData)
```

`chromatogram_plotly` *Render a Sanger chromatogram as an interactive Plotly widget*

Description

Wraps the four trace channels (A/C/G/T) of a sangerseq / SangerRead object into a single `plotly` htmlwidget that renders via WebGL (`scattergl`). Intended for embedding in Shiny dashboards where the static `chromatogram_overwrite` would be too heavy.

Usage

```
chromatogram_plotly(
  obj,
  trim5 = 0,
  trim3 = 0,
  max_points = 8000L,
  showtrim = FALSE,
  colors = "default"
)
```

Arguments

`obj`	A sangerseq or SangerRead instance with a populated `traceMatrix`.
`trim5`	Integer; if `showtrim` is TRUE, shade the first `trim5` positions to indicate the 5' trim region.
`trim3`	Integer; if `showtrim` is TRUE, shade the last `trim3` positions to indicate the 3' trim region.
`max_points`	Integer cap on the number of points rendered per channel. When the trace exceeds `max_points` it is downsampled by uniform stride.
`showtrim`	Logical; whether to overlay shaded trim regions.
`colors`	Either `"default"`, `"cb_friendly"`, or a length-5 character vector of hex colours for (A, T, C, G, other).

Value

A `plotly` htmlwidget. The returned object carries a `downsample_info` attribute reporting the original and rendered point counts plus the stride.

Examples

```
data(sangerReadFData)

chromatogram_plotly(sangerReadFData)
```

`generateReport` *Method generateReport*

Description

A method which generates final reports of the SangerRead, SangerContig, and SangerAlignment instance.

Usage

```
generateReport(
  object,
  outputDir = NULL,
  includeSangerContig = TRUE,
  includeSangerRead = TRUE,
  colors = "default",
  ...
)
```

Arguments

`object`	A SangerRead, SangerContig, or SangerAlignment S4 instance.
`outputDir`	The output directory of the generated HTML report.
`includeSangerContig`	The parameter that decides whether to include SangerContig level report. The value is `TRUE` or `FALSE` and the default is `TRUE`.
`includeSangerRead`	The parameter that decides whether to include SangerRead level report. The value is `TRUE` or `FALSE` and the default is `TRUE`.
`colors`	A vector for users to set the colors of (A, T, C, G, else). There are three options for users to choose from. 1. "default": (green, blue, black, red, purple). 2. "cb_friendly": ((0, 0, 0), (199, 199, 199), (0, 114, 178), (213, 94, 0), (204, 121, 167)). 3. Users can set their own colors with a vector with five elements.
`...`	Further generateReportSR, generateReportSC, and generateReportSA related parameters.

Value

A `SangerRead`, `SangerContig`, or `SangerAlignment` object.

Author(s)

Kuan-Hao Chao

Examples

```
data(sangerReadFData)
data(sangerContigData)
data(sangerAlignmentData)

generateReport(sangerReadFData)
generateReport(sangerReadFData, colors="cb_friendly")
generateReport(sangerContigData)
generateReport(sangerContigData, colors="cb_friendly")
generateReport(sangerAlignmentData)
generateReport(sangerAlignmentData, colors="cb_friendly")
```

`generateReportSA` *Method generateReportSA*

Description

Method generateReportSA

Usage

```
generateReportSA(
  object,
  outputDir = NULL,
  includeSangerContig = TRUE,
  includeSangerRead = TRUE,
  colors = "default",
  ...
)
```

Arguments

`object` A SangerAlignment S4 instance.

`outputDir` The output directory of the generated HTML report.

`includeSangerContig` The parameter that decides whether to include SangerContig level report. The value is `TRUE` or `FALSE` and the default is `TRUE`.

`includeSangerRead` The parameter that decides whether to include SangerRead level report. The value is `TRUE` or `FALSE` and the default is `TRUE`.

`colors` A vector for users to set the colors of (A, T, C, G, else). There are three options for users to choose from. 1. "default": (green, blue, black, red, purple). 2. "cb_friendly": ((0, 0, 0), (199, 199, 199), (0, 114, 178), (213, 94, 0), (204, 121, 167)). 3. Users can set their own colors with a vector with five elements.

`...` Further generateReportSA-related parameters.

Value

The output absolute path to the SangerAlignment's HTML file.

Examples

```
data(sangerAlignmentData)

generateReportSA(sangerAlignmentData)
```

`generateReportSC` *Method generateReportSC*

Description

Method generateReportSC

Usage

```
generateReportSC(
  object,
  outputDir = NULL,
  includeSangerRead = TRUE,
  colors = "default",
  ...
)
```

Arguments

`object`	A SangerContig S4 instance.
`outputDir`	The output directory of the generated HTML report.
`includeSangerRead`	The parameter that decides whether to include SangerRead level report. The value is `TRUE` or `FALSE` and the default is `TRUE`.
`colors`	A vector for users to set the colors of (A, T, C, G, else). There are three options for users to choose from. 1. "default": (green, blue, black, red, purple). 2. "cb_friendly": ((0, 0, 0), (199, 199, 199), (0, 114, 178), (213, 94, 0), (204, 121, 167)). 3. Users can set their own colors with a vector with five elements.
`...`	Further generateReportSC-related parameters.

Value

The output absolute path to the SangerContig's HTML file.

Examples

```
data(sangerContigData)

generateReportSC(sangerContigData)
```

generateReportSR *Method generateReportSR*

Description

Method generateReportSR

Usage

```
generateReportSR(object, outputDir = NULL, colors = "default", ...)
```

Arguments

`object`	A SangerRead S4 instance.
`outputDir`	The output directory of the generated HTML report.
`colors`	A vector for users to set the colors of (A, T, C, G, else). There are three options for users to choose from. 1. "default": (green, blue, black, red, purple). 2. "cb_friendly": ((0, 0, 0), (199, 199, 199), (0, 114, 178), (213, 94, 0), (204, 121, 167)). 3. Users can set their own colors with a vector with five elements.
`...`	Further generateReportSR-related parameters.

Value

The output absolute path to the SangerRead's HTML file.

Examples

```
data(sangerReadFData)

generateReportSR(sangerReadFData)
```

globalTrimApp *Launch a global trimming controls dashboard*

Description

Opens a Shiny gadget that exposes M1/M2 trimming parameters as sliders/numeric inputs, applies any change to *every* SangerRead in the supplied SangerAlignment via ‘updateQualityParam(SA, ...)‘, and live-previews the consensus length, contig count, and number of reads that survive the new trim.

Unlike ‘launchAppSA()‘ (which exposes per-read trimming) this app operates globally — useful when a whole batch needs the same re-trimming policy.

Usage

```
globalTrimApp(SA)
```

Arguments

`SA`	A SangerAlignment instance.

Value

The (last-applied) SangerAlignment when the user clicks "Done", or NULL if the user cancels.

Examples

```
## Not run:
data(sangerAlignmentData)
SA2 <- globalTrimApp(sangerAlignmentData)

## End(Not run)
```

---

launchApp *Method launchApp*

Description

A method which launches Shiny application of the SangerContig and SangerAlignment instance.

Usage

```
launchApp(object, outputDir = NULL, colors = "default")
```

Arguments

`object`	A SangerContig or SangerAlignment S4 instance.
`outputDir`	The output directory of the saved new SangerContig or SangerAlignment S4 instance.
`colors`	A vector for users to set the colors of (A, T, C, G, else). There are three options for users to choose from. 1. "default": (green, blue, black, red, purple). 2. "cb_friendly": ((0, 0, 0), (199, 199, 199), (0, 114, 178), (213, 94, 0), (204, 121, 167)). 3. Users can set their own colors with a vector with five elements.

Value

A `SangerContig` or `SangerAlignment` object.

Author(s)

Kuan-Hao Chao

Examples

```
data(sangerContigData)
data(sangerAlignmentData)
## Not run:
launchApp(sangerContigData)
launchApp(sangerContigData, colors="cb_friendly")
launchApp(sangerAlignmentData)
launchApp(sangerAlignmentData, colors="cb_friendly")
## End(Not run)
```

`launchAppSA` *Method launchAppSA*

Description

Method launchAppSA

Usage

```
launchAppSA(object, outputDir = NULL, colors = "default")
```

Arguments

`object`	A SangerAlignment S4 instance.
`outputDir`	The output directory of the saved new SangerAlignment S4 instance.
`colors`	A vector for users to set the colors of (A, T, C, G, else). There are three options for users to choose from. 1. "default": (green, blue, black, red, purple). 2. "cb_friendly": ((0, 0, 0), (199, 199, 199), (0, 114, 178), (213, 94, 0), (204, 121, 167)). 3. Users can set their own colors with a vector with five elements.

Value

A `shiny.appobj` object.

Examples

```
data(sangerAlignmentData)
## Not run:
launchAppSA(sangerAlignmentData)
## End(Not run)
```

`launchAppSC` *Method launchAppSC*

Description

Method launchAppSC

Usage

```
launchAppSC(object, outputDir = NULL, colors = "default")
```

Arguments

`object`	A SangerContig S4 instance.
`outputDir`	The output directory of the saved new SangerContig S4 instance.
`colors`	A vector for users to set the colors of (A, T, C, G, else). There are three options for users to choose from. 1. "default": (green, blue, black, red, purple). 2. "cb_friendly": ((0, 0, 0), (199, 199, 199), (0, 114, 178), (213, 94, 0), (204, 121, 167)). 3. Users can set their own colors with a vector with five elements.

Value

A `shiny.appobj` object.

Examples

```
data(sangerContigData)
## Not run:
launchAppSC(sangerContigData)
## End(Not run)
```

`MakeBaseCalls` *Method MakeBaseCalls*

Description

Method MakeBaseCalls

Usage

```
MakeBaseCalls(object, signalRatioCutoff = 0.33)
```

Arguments

`object` A SangerRead S4 instance.

`signalRatioCutoff`
The ratio of the height of a secondary peak to a primary peak. Secondary peaks higher than this ratio are annotated. Those below the ratio are excluded. The default value is `0.33`.

Value

A `SangerRead` instance.

Examples

```
data(sangerReadFData)
MakeBaseCalls(sangerReadFData, signalRatioCutoff = 0.22)
```

`ObjectResults-class` *ObjectResults*

Description

An S4 class storing results related inputs in a SangerRead, SangerContig, and SangerAlignment S4 object.

Slots

creationResult Single logical: TRUE if construction succeeded, FALSE if any input failed validation.

errorMessages Character vector of error messages collected during construction (one per failure).

errorTypes Character vector of machine-readable error tags (e.g. "PARAMETER_RANGE_ERROR"); same length as errorMessages.

warningMessages Character vector of warning messages emitted during construction.

warningTypes Character vector of machine-readable warning tags; same length as warningMessages.

readResultTable A data frame with one row per Sanger read processed, recording per-read creation outcome and any error tag.

printLevel Character indicating which tier ("SangerRead", "SangerContig", or "SangerAlignment") emitted these results.

Author(s)

Kuan-Hao Chao

Examples

```
objectResults <- new("ObjectResults",
                     creationResult   = TRUE,
                     errorMessages    = character(0),
                     errorTypes       = character(0),
                     warningMessages  = character(0),
                     warningTypes     = character(0),
                     readResultTable =  data.frame(),
                     printLevel       = "SangerRead")
```

---

primaryAASeqS1 *Method primaryAASeqS1*

Description

Method primaryAASeqS1

Usage

```
primaryAASeqS1(object)

## S4 method for signature 'SangerRead'
primaryAASeqS1(object)
```

Arguments

object A SangerRead S4 instance.

Value

The frame-1 amino-acid translation of the read's primary sequence as an AAString. Computed lazily — if the slot was populated eagerly at construction time it is returned directly; if the slot is empty (the default when refAminoAcidSeq == "" and lazyAA = TRUE) it is computed via calculateAASeq() on demand.

Examples

```
data(sangerReadFData)
primaryAASeqS1(sangerReadFData)
```

---

primaryAASeqS2 *Method primaryAASeqS2*

Description

Method primaryAASeqS2

Usage

```
primaryAASeqS2(object)

## S4 method for signature 'SangerRead'
primaryAASeqS2(object)
```

Arguments

`object`	A SangerRead S4 instance.

Value

Frame-2 AA translation as `AAString` (lazy).

Examples

```
data(sangerReadFData)
primaryAASeqS2(sangerReadFData)
```

---

primaryAASeqS3 *Method primaryAASeqS3*

Description

Method primaryAASeqS3

Usage

```
primaryAASeqS3(object)

## S4 method for signature 'SangerRead'
primaryAASeqS3(object)
```

Arguments

`object`	A SangerRead S4 instance.

Value

Frame-3 AA translation as `AAString` (lazy).

Examples

```
data(sangerReadFData)
primaryAASeqS3(sangerReadFData)
```

`qualityBasePlot` *Method qualityBasePlot*

Description

Method qualityBasePlot

Usage

```
qualityBasePlot(object)
```

Arguments

`object`	A QualityReport or SangerRead S4 instance

Value

A quality plot.

Examples

```
data(qualityReportData)
data(sangerReadFData)
qualityBasePlot(qualityReportData)
qualityBasePlot(sangerReadFData)
```

`QualityReport-class` *QualityReport*

Description

An S4 class storing quality related inputs and results in a SangerRead S4 object.

Slots

- `TrimmingMethod` The read trimming method for this SangerRead. The value must be `"M1"` (the default) or `'M2'`.
- `M1TrimmingCutoff` The trimming cutoff for the Method 1. If `TrimmingMethod` is `"M1"`, then the default value is `0.0001`. Otherwise, the value must be `NULL`.
- `M2CutoffQualityScore` The trimming cutoff quality score for the Method 2. If `TrimmingMethod` is `'M2'`, then the default value is `20`. Otherwise, the value must be `NULL`. It works with `M2SlidingWindowSize`.
- `M2SlidingWindowSize` The trimming sliding window size for the Method 2. If `TrimmingMethod` is `'M2'`, then the default value is `10`. Otherwise, the value must be `NULL`. It works with `M2CutoffQualityScore`.
- `qualityPhredScores` The Phred quality scores of each base pairs after base calling.
- `qualityBaseScores` The probability of incorrect base call of each base pairs. They are calculated from `qualityPhredScores`.
- `rawSeqLength` The number of nucleotides of raw primary DNA sequence.
- `trimmedSeqLength` The number of nucleotides of trimeed primary DNA sequence.
- `trimmedStartPos` The base pair index of trimming start point from 5' end of the sequence.
- `trimmedFinishPos` The base pair index of trimming finish point from 3' end of the sequence.
- `rawMeanQualityScore` The mean quality score of the primary sequence after base calling. In other words, it is the mean of `qualityPhredScores`.
- `trimmedMeanQualityScore` The mean quality score of the trimmed primary sequence after base calling.
- `rawMinQualityScore` The minimum quality score of the primary sequence after base calling.
- `trimmedMinQualityScore` The minimum quality score of the trimmed primary sequence after base calling.
- `remainingRatio` The remaining sequence length ratio after trimming.

Author(s)

Kuan-Hao Chao

Examples

```
inputFilesPath <- system.file("extdata/", package = "sangeranalyseR")
A_chloroticaFFN <- file.path(inputFilesPath,
                             "Allolobophora_chlorotica",
                             "ACHLO",
                             "Achl_ACHLO006-09_1_F.ab1")
sangerReadF <- new("SangerRead",
                    inputSource           = "ABIF",
                    readFeature           = "Forward Read",
                    readFileName          = A_chloroticaFFN,
                    geneticCode           = GENETIC_CODE,
                    TrimmingMethod        = "M1",
                    M1TrimmingCutoff      = 0.0001,
                    M2CutoffQualityScore  = NULL,
                    M2SlidingWindowSize   = NULL,
                    baseNumPerRow         = 100,
                    heightPerRow          = 200,
```

```
                      signalRatioCutoff     = 0.33,
                      showTrimmed           = TRUE)
"@"(sangerReadF, QualityReport)
```

QualityReport-class-qualityBasePlot
qualityBasePlot

Description

A QualityReport method which creates quality base interactive plot.

Usage

```
## S4 method for signature 'QualityReport'
qualityBasePlot(object)
```

Arguments

object	A QualityReport S4 instance.

Value

A quality plot.

Examples

```
data("qualityReportData")

qualityBasePlot(qualityReportData)
```

QualityReport-class-updateQualityParam
updateQualityParam

Description

A QualityReport method which updates quality base interactive plot.

Usage

```
## S4 method for signature 'QualityReport'
updateQualityParam(
  object,
  TrimmingMethod = "M1",
  M1TrimmingCutoff = 1e-04,
  M2CutoffQualityScore = NULL,
  M2SlidingWindowSize = NULL
)
```

Arguments

- `object` — A QualityReport S4 instance.
- `TrimmingMethod` — The read trimming method for this SangerRead. The value must be `"M1"` (the default) or `'M2'`.
- `M1TrimmingCutoff` — The trimming cutoff for the Method 1. If `TrimmingMethod` is `"M1"`, then the default value is `0.0001`. Otherwise, the value must be `NULL`.
- `M2CutoffQualityScore` — The trimming cutoff quality score for the Method 2. If `TrimmingMethod` is `'M2'`, then the default value is `20`. Otherwise, the value must be `NULL`. It works with `M2SlidingWindowSize`.
- `M2SlidingWindowSize` — The trimming sliding window size for the Method 2. If `TrimmingMethod` is `'M2'`, then the default value is `10`. Otherwise, the value must be `NULL`. It works with `M2CutoffQualityScore`.

Value

A QualityReport instance.

Examples

```
data("qualityReportData")
updateQualityParam(qualityReportData,
                   TrimmingMethod         = "M2",
                   M1TrimmingCutoff       = NULL,
                   M2CutoffQualityScore   = 30,
                   M2SlidingWindowSize    = 15)
```

`qualityReportData` *QualityReport instance*

Description

A pre-built `QualityReport` S4 object derived from the bundled ACHLO ABIF fixture, suitable for vignette and example use without re-running the trimming pipeline.

Usage

```
data(qualityReportData)
```

Format

A `QualityReport-class` S4 object containing per-base Phred scores, the trimmed start/finish positions, and the raw / trimmed mean / minimum quality scores.

Author(s)

Kuan-Hao Chao

readTable — *Method readTable*

Description

Method readTable

Usage

```
readTable(object, indentation = 0, ...)
```

Arguments

`object`	A SangerRead, SangerContig, or SangerAlignment S4 instance.
`indentation`	The indentation for different level printing
`...`	Further generateReportSR-related parameters.

Value

None.

Examples

```
data(sangerReadFData)
data(sangerContigData)

readTable(sangerReadFData)
readTable(sangerContigData)
```

SangerAlignment — *SangerAlignment*

Description

the wrapper function for SangerAlignment

Usage

```
SangerAlignment(
  printLevel = "SangerAlignment",
  inputSource = "ABIF",
  processMethod = "REGEX",
  ABIF_Directory = NULL,
  FASTA_File = NULL,
  REGEX_SuffixForward = NULL,
  REGEX_SuffixReverse = NULL,
  CSV_NamesConversion = NULL,
  geneticCode = GENETIC_CODE,
  TrimmingMethod = "M1",
```

```
  M1TrimmingCutoff = 1e-04,
  M2CutoffQualityScore = NULL,
  M2SlidingWindowSize = NULL,
  baseNumPerRow = 100,
  heightPerRow = 200,
  signalRatioCutoff = 0.33,
  showTrimmed = TRUE,
  refAminoAcidSeq = "",
  minReadsNum = 2,
  minReadLength = 20,
  minFractionCall = 0.5,
  maxFractionLost = 0.5,
  acceptStopCodons = TRUE,
  readingFrame = 1,
  processorsNum = 1,
  BPPARAM = NULL,
  lazyAA = TRUE,
  minOverlapFraction = 0,
  minOverlapBases = 0L,
  alignSeqsParams = list(),
  consensusMethod = "strict",
  qualityAware = FALSE
)
```

Arguments

`printLevel` Internal — controls log verbosity when this constructor is called recursively from a parent class. Defaults to "SangerAlignment"; do not set manually.

`inputSource` The input source of the raw file. It must be "ABIF" or "FASTA". The default value is "ABIF".

`processMethod` The method used to group reads into contigs. Either "REGEX" (use REGEX_SuffixForward / REGEX_SuffixReverse) or "CSV" (use CSV_NamesConversion). The default is "REGEX".

`ABIF_Directory` The parent directory of all of the reads contained in ABIF format you wish to analyse. In SangerAlignment, all reads in subdirectories will be scanned recursively.

`FASTA_File` If inputSource is "FASTA", then this value has to be the name of the FASTA file; if inputSource is "ABIF", then this value is "" by default.

`REGEX_SuffixForward`
The suffix of the filenames for forward reads in regular expression, i.e. reads that do not need to be reverse-complemented. For forward reads, it should be "_F.ab1".

`REGEX_SuffixReverse`
The suffix of the filenames for reverse reads in regular expression, i.e. reads that need to be reverse-complemented. For revcerse reads, it should be "_R.ab1".

`CSV_NamesConversion`
The file path to the CSV file that provides read names that follow the naming regulation. If inputSource is "FASTA", then users need to prepare the csv file or make sure the original names inside FASTA file are valid; if inputSource is "ABIF", then this value is NULL by default.

`geneticCode` Named character vector in the same format as `GENETIC_CODE` (the default), which represents the standard genetic code. This is the code with which the function will attempt to translate your DNA sequences. You can get an appropriate vector with the getGeneticCode() function. The default is the standard code.

`TrimmingMethod` TrimmingMethod The read trimming method for this SangerRead. The value must be `"M1"` (the default) or `'M2'`.

`M1TrimmingCutoff`
The trimming cutoff for the Method 1. If `TrimmingMethod` is `"M1"`, then the default value is `0.0001`. Otherwise, the value must be `NULL`.

`M2CutoffQualityScore`
The trimming cutoff quality score for the Method 2. If `TrimmingMethod` is `'M2'`, then the default value is `20`. Otherwise, the value must be `NULL`. It works with `M2SlidingWindowSize`.

`M2SlidingWindowSize`
The trimming sliding window size for the Method 2. If `TrimmingMethod` is `'M2'`, then the default value is `10`. Otherwise, the value must be `NULL`. It works with `M2CutoffQualityScore`.

`baseNumPerRow` It defines maximum base pairs in each row. The default value is `100`.

`heightPerRow` It defines the height of each row in chromatogram. The default value is `200`.

`signalRatioCutoff`
The ratio of the height of a secondary peak to a primary peak. Secondary peaks higher than this ratio are annotated. Those below the ratio are excluded. The default value is `0.33`.

`showTrimmed` The logical value storing whether to show trimmed base pairs in chromatogram. The default value is `TRUE`.

`refAminoAcidSeq`
An amino acid reference sequence supplied as a string or an AAString object. If your sequences are protein-coding DNA seuqences, and you want to have frameshifts automatically detected and corrected, supply a reference amino acid sequence via this argument. If this argument is supplied, the sequences are then kept in frame for the alignment step. Fwd sequences are assumed to come from the sense (i.e. coding, or "+") strand. The default value is `""`.

`minReadsNum` The minimum number of reads required to make a consensus sequence, must be 2 or more. The default value is `2`.

`minReadLength` Reads shorter than this will not be included in the readset. The default `20` means that all reads with length of 20 or more will be included. Note that this is the length of a read after it has been trimmed.

`minFractionCall`
Minimum fraction of the sequences required to call a consensus sequence for SangerContig at any given position (see the ConsensusSequence() function from DECIPHER for more information). Defaults to 0.75 implying that 3/4 of all reads must be present in order to call a consensus.

`maxFractionLost`
Numeric giving the maximum fraction of sequence information that can be lost in the consensus sequence for SangerContig (see the ConsensusSequence() function from DECIPHER for more information). Defaults to 0.5, implying that each consensus base can ignore at most 50 percent of the information at a given position.

acceptStopCodons
: The logical value TRUE or FALSE. TRUE (the defualt): keep all reads, regardless of whether they have stop codons; FALSE: reject reads with stop codons. If FALSE is selected, then the number of stop codons is calculated after attempting to correct frameshift mutations (if applicable).

readingFrame
: 1, 2, or 3. Only used if accept.stop.codons == FALSE. This specifies the reading frame that is used to determine stop codons. If you use a refAminoAcidSeq, then the frame should always be 1, since all reads will be shifted to frame 1 during frameshift correction. Otherwise, you should select the appropriate reading frame.

processorsNum
: The number of processors to use, or NULL (the default) for all available processors.

BPPARAM
: A BiocParallelParam instance that controls how the per-SangerRead construction loop is parallelised. Defaults to NULL, in which case it is derived from processorsNum.

lazyAA
: Logical (default TRUE). When TRUE and refAminoAcidSeq == "", the per-read 3-frame amino-acid translation is skipped at construction time and computed on demand via primaryAASeqS1/S2/S3().

minOverlapFraction
: Numeric in [0, 1] (default 0.0). When > 0, after read alignment the smallest pairwise non-gap overlap is computed; if it falls below minOverlapFraction * shorter_read_length, a LOW_OVERLAP_WARN is logged. Use this to detect spurious merges of poorly-overlapping forward/reverse reads (issues #94, #66).

minOverlapBases
: Integer (default 0L). Like minOverlapFraction but expressed in absolute base pairs; the warning fires if the smallest pairwise overlap is below this value. Whichever of the two thresholds is larger applies.

alignSeqsParams
: A named list (default list()) of additional arguments forwarded to DECIPHER::AlignSeqs (or AlignTranslation when refAminoAcidSeq != ""). Useful for tuning alignment behaviour on minimal-overlap 16S reads (e.g. list(iterations = 1L, refinements = 1L)).

consensusMethod
: One of "strict" (default; uses DECIPHER's ConsensusSequence with IUPAC ambiguity codes), "majority" (per-column plurality vote, no ambiguity codes), or "quality_weighted" (per-column vote weighted by source-read Phred scores). Issues #87, #48.

qualityAware
: Logical shorthand (default FALSE); when TRUE, equivalent to consensusMethod = "quality_weighted". Issue #48.

Value

A SangerAlignment instance.

Author(s)

Kuan-Hao Chao

Examples

```
rawDataDir <- system.file("extdata", package = "sangeranalyseR")
```

```
parentDir <- file.path(rawDataDir, "Allolobophora_chlorotica", "RBNII")
REGEX_SuffixForward <- "_[0-9]*_F.ab1$"
REGEX_SuffixReverse <- "_[0-9]*_R.ab1$"
sangerAlignment <- SangerAlignment(
                       inputSource            = "ABIF",
                       ABIF_Directory       = parentDir,
                       REGEX_SuffixForward   = REGEX_SuffixForward,
                       REGEX_SuffixReverse   = REGEX_SuffixReverse,
               refAminoAcidSeq = "SRQWLFSTNHKDIGTLYFIFGAWAGMVGTSLSILIRAELGHPGALIGDDQIYNVIVTAHAFIMIFFMVMPIMI
                       TrimmingMethod        = "M1",
                       M1TrimmingCutoff      = 0.0001,
                       M2CutoffQualityScore  = NULL,
                       M2SlidingWindowSize   = NULL,
                       baseNumPerRow         = 100,
                       heightPerRow          = 200,
                       signalRatioCutoff     = 0.33,
                       showTrimmed           = TRUE,
                       processorsNum         = 2)
```

`SangerAlignment-class` *SangerAlignment*

Description

An S4 class containing SangerContigs lists and contigs alignment results which corresponds to a final alignment in Sanger sequencing.

Slots

`objectResults` This is the object that stores all information of the creation result.

`inputSource` The input source of the raw file. It must be `"ABIF"` or `"FASTA"`. The default value is `"ABIF"`.

`processMethod` The method to create a contig from reads. The value is `"REGEX"` or `"CSV"`. The default value is `"REGEX"`.

`ABIF_Directory` If `inputSource` is `"ABIF"`, then this value is the path of a parent directory storing all reads in ABIF format you want to analyse. If `inputSource` is `"FASTA"`, then this value has to be `NULL` by default.

`FASTA_File` If `inputSource` is `"FASTA"`, then this value has to be the path to a valid FASTA file ; if `inputSource` is `"ABIF"`, then this value has to be `NULL` by default.

`REGEX_SuffixForward` The suffix of the filenames for forward reads in regular expression, i.e. reads that do not need to be reverse-complemented. For forward reads, it should be `"_F.ab1"`.

`REGEX_SuffixReverse` The suffix of the filenames for reverse reads in regular expression, i.e. reads that need to be reverse-complemented. For revcerse reads, it should be `"_R.ab1"`.

`CSV_NamesConversion` The file path to the CSV file that provides read names, directions, and their contig groups. If `processMethod` is `"CSV"`, then this value has to be the path to a valid CSV file; if `processMethod` is `"REGEX"`, then this value has to be `NULL` by default.

`geneticCode` Named character vector in the same format as `GENETIC_CODE` (the default), which represents the standard genetic code. This is the code with which the function will attempt to translate your DNA sequences. You can get an appropriate vector with the getGeneticCode() function. The default is the standard code.

`refAminoAcidSeq` An amino acid reference sequence supplied as a string or an AAString object. If your sequences are protein-coding DNA seuqences, and you want to have frameshifts automatically detected and corrected, supply a reference amino acid sequence via this argument. If this argument is supplied, the sequences are then kept in frame for the alignment step. Fwd sequences are assumed to come from the sense (i.e. coding, or "+") strand. The default value is "".

`contigList` A list storing all SangerContigs S4 instances.

`contigsConsensus` The consensus read of all SangerContig S4 instances in DNAString object.

`contigsAlignment` The alignment of all SangerContig S4 instances with the called consensus sequence in DNAStringSet object. Users can use BrowseSeqs() to view the alignment.

`contigsTree` A phylo instance returned by bionj function in ape package. It can be used to draw the tree.

Author(s)

Kuan-Hao Chao

Examples

```
## Simple example
rawDataDir <- system.file("extdata", package = "sangeranalyseR")
parentDir <- file.path(rawDataDir, 'Allolobophora_chlorotica', 'ACHLO')
my_aligned_contigs <- new("SangerAlignment",
                          ABIF_Directory     = parentDir,
                          REGEX_SuffixForward = "_[0-9]*_F.ab1$",
                          REGEX_SuffixReverse = "_[0-9]*_R.ab1$")

rawDataDir <- system.file("extdata", package = "sangeranalyseR")
parentDir <- file.path(rawDataDir, 'Allolobophora_chlorotica', 'ACHLO')
CSV_NamesConversion <- file.path(rawDataDir, "ab1", "SangerAlignment", "names_conversion.csv")
sangerAlignment <- new("SangerAlignment",
                       processMethod          = "CSV",
                       ABIF_Directory         = parentDir,
                       CSV_NamesConversion    = CSV_NamesConversion)

## Input From ABIF file format (Regex)
REGEX_SuffixForward <- "_[0-9]*_F.ab1$"
REGEX_SuffixReverse <- "_[0-9]*_R.ab1$"
sangerAlignment <- new("SangerAlignment",
                       printLevel            = "SangerAlignment",
                       inputSource           = "ABIF",
                       processMethod         = "REGEX",
                       FASTA_File            = NULL,
                       CSV_NamesConversion   = NULL,
                       ABIF_Directory        = parentDir,
                       REGEX_SuffixForward   = REGEX_SuffixForward,
                       REGEX_SuffixReverse   = REGEX_SuffixReverse,
                       TrimmingMethod        = "M1",
                       M1TrimmingCutoff      = 0.0001,
                       M2CutoffQualityScore  = NULL,
                       M2SlidingWindowSize   = NULL,
                       baseNumPerRow         = 100,
                       heightPerRow          = 200,
                       signalRatioCutoff     = 0.33,
                       showTrimmed           = TRUE,
```

```
                refAminoAcidSeq = "SRQWLFSTNHKDIGTLYFIFGAWAGMVGTSLSILIRAELGHPGALIGDDQIYNVIVTAHAFIMIFFMVMPIMI
                        minReadsNum           = 2,
                        minReadLength         = 20,
                        minFractionCall       = 0.5,
                        maxFractionLost       = 0.5,
                        geneticCode           = GENETIC_CODE,
                        acceptStopCodons      = TRUE,
                        readingFrame          = 1,
                        processorsNum         = 2)

## Input From ABIF file format (Csv three column)
rawDataDir <- system.file("extdata", package = "sangeranalyseR")
parentDir <- file.path(rawDataDir, 'Allolobophora_chlorotica', 'ACHLO')
CSV_NamesConversion <- file.path(rawDataDir, "ab1", "SangerAlignment",
"names_conversion_all.csv")
sangerAlignment <- new("SangerAlignment",
                        inputSource           = "ABIF",
                        processMethod         = "CSV",
                        ABIF_Directory        = parentDir,
                        CSV_NamesConversion   = CSV_NamesConversion,
                refAminoAcidSeq = "SRQWLFSTNHKDIGTLYFIFGAWAGMVGTSLSILIRAELGHPGALIGDDQIYNVIVTAHAFIMIFFMVMPIMI
                        TrimmingMethod        = "M1",
                        M1TrimmingCutoff      = 0.0001,
                        M2CutoffQualityScore  = NULL,
                        M2SlidingWindowSize   = NULL,
                        baseNumPerRow         = 100,
                        heightPerRow          = 200,
                        signalRatioCutoff     = 0.33,
                        showTrimmed           = TRUE,
                        processorsNum         = 2)

## Input From FASTA file format (No Csv - Regex)
rawDataDir <- system.file("extdata", package = "sangeranalyseR")
fastaFN <- file.path(rawDataDir, "fasta",
                     "SangerAlignment", "Sanger_all_reads.fa")
REGEX_SuffixForwardFa <- "_[0-9]*_F$"
REGEX_SuffixReverseFa <- "_[0-9]*_R$"
sangerAlignmentFa <- new("SangerAlignment",
                          inputSource           = "FASTA",
                          processMethod         = "REGEX",
                          FASTA_File            = fastaFN,
                          REGEX_SuffixForward   = REGEX_SuffixForwardFa,
                          REGEX_SuffixReverse   = REGEX_SuffixReverseFa,
                 refAminoAcidSeq = "SRQWLFSTNHKDIGTLYFIFGAWAGMVGTSLSILIRAELGHPGALIGDDQIYNVIVTAHAFIMIFFMVMPIM
                          processorsNum         = 2)

## Input From FASTA file format (Csv three column method)
rawDataDir <- system.file("extdata", package = "sangeranalyseR")
fastaFN <- file.path(rawDataDir, "fasta",
                     "SangerAlignment", "Sanger_all_reads.fa")
CSV_NamesConversion <- file.path(rawDataDir, "fasta",
                                 "SangerAlignment", "names_conversion.csv")
sangerAlignmentFa <- new("SangerAlignment",
                          inputSource           = "FASTA",
                          processMethod         = "CSV",
                          FASTA_File            = fastaFN,
                          CSV_NamesConversion   = CSV_NamesConversion,
```

```
                  refAminoAcidSeq = "SRQWLFSTNHKDIGTLYFIFGAWAGMVGTSLSILIRAELGHPGALIGDDQIYNVIVTAHAFIMIFFMVMPIM
                         processorsNum         = 2)
```

SangerAlignment-class-generateReportSA
generateReportSA

Description

A SangerAlignment method which generates final reports of the SangerContig instance.

Usage

```
## S4 method for signature 'SangerAlignment'
generateReportSA(
  object,
  outputDir,
  includeSangerContig = TRUE,
  includeSangerRead = TRUE,
  colors
)
```

Arguments

object	A SangerAlignment S4 instance.
outputDir	The output directory of the generated HTML report.
includeSangerContig	The parameter that decides whether to include SangerContig level report. The value is TRUE or FALSE and the default is TRUE.
includeSangerRead	The parameter that decides whether to include SangerRead level report. The value is TRUE or FALSE and the default is TRUE.
colors	A vector for users to set the colors of (A, T, C, G, else). There are three options for users to choose from. 1. "default": (green, blue, black, red, purple). 2. "cb_friendly": ((0, 0, 0), (199, 199, 199), (0, 114, 178), (213, 94, 0), (204, 121, 167)). 3. Users can set their own colors with a vector with five elements.

Value

The output absolute path to the SangerAlignment's HTML file.

Examples

```
data("sangerAlignmentData")

generateReportSA(sangerAlignmentData)
generateReportSA(sangerAlignmentData, colors="cb_friendly")
```

`SangerAlignment-class-launchAppSA`
launchAppSA

Description

A SangerAlignment method which launches Shiny app for SangerAlignment instance.

Usage

```
## S4 method for signature 'SangerAlignment'
launchAppSA(object, outputDir = NULL, colors = "default")
```

Arguments

`object`	A SangerAlignment S4 instance.
`outputDir`	The output directory of the saved new SangerContig S4 instance.
`colors`	A vector for users to set the colors of (A, T, C, G, else). There are three options for users to choose from. 1. "default": (green, blue, black, red, purple). 2. "cb_friendly": ((0, 0, 0), (199, 199, 199), (0, 114, 178), (213, 94, 0), (204, 121, 167)). 3. Users can set their own colors with a vector with five elements.

Value

A `shiny.appobj` object.

Examples

```
data("sangerAlignmentData")
RShinySA <- launchAppSA(sangerAlignmentData)
RShinySA <- launchAppSA(sangerAlignmentData, colors="cb_friendly")
```

`SangerAlignment-class-updateQualityParam`
updateQualityParam

Description

A SangerAlignment method which updates QualityReport parameter for each the SangerRead instance inside SangerAlignment.

Usage

```
## S4 method for signature 'SangerAlignment'
updateQualityParam(
  object,
  TrimmingMethod = "M1",
  M1TrimmingCutoff = 1e-04,
  M2CutoffQualityScore = NULL,
  M2SlidingWindowSize = NULL,
  processorsNum = NULL
)
```

Arguments

- `object` A SangerAlignment S4 instance.
- `TrimmingMethod` The read trimming method for this SangerRead. The value must be `"M1"` (the default) or `'M2'`.
- `M1TrimmingCutoff` The trimming cutoff for the Method 1. If `TrimmingMethod` is `"M1"`, then the default value is `0.0001`. Otherwise, the value must be `NULL`.
- `M2CutoffQualityScore` The trimming cutoff quality score for the Method 2. If `TrimmingMethod` is `'M2'`, then the default value is `20`. Otherwise, the value must be `NULL`. It works with `M2SlidingWindowSize`.
- `M2SlidingWindowSize` The trimming sliding window size for the Method 2. If `TrimmingMethod` is `'M2'`, then the default value is `10`. Otherwise, the value must be `NULL`. It works with `M2CutoffQualityScore`.
- `processorsNum` The number of processors to use, or NULL (the default) for all available processors.

Value

A SangerAlignment instance.

Examples

```
data("sangerAlignmentData")

updateQualityParam(sangerAlignmentData,
                   TrimmingMethod         = "M2",
                   M1TrimmingCutoff       = NULL,
                   M2CutoffQualityScore   = 40,
                   M2SlidingWindowSize    = 15)
```

SangerAlignment-class-writeFastaSA

writeFastaSA

Description

A SangerAlignment method which writes sequences into Fasta files.

Usage

```
## S4 method for signature 'SangerAlignment'
writeFastaSA(
  object,
  outputDir = NULL,
  compress = FALSE,
  compression_level = NA,
  selection = "all"
)
```

Arguments

`object`	A SangerAlignment S4 instance.
`outputDir`	The output directory of generated FASTA files.
`compress`	Like for the `save` function in base R, must be `TRUE` or `FALSE` (the default), or a single string specifying whether writing to the file is to use compression. The only type of compression supported at the moment is "gzip". This parameter will be passed to `writeXStringSet` function in Biostrings package.
`compression_level`	This parameter will be passed to `writeXStringSet` function in Biostrings package.
`selection`	This value can be `all`, `contigs_alignment`, `contigs_unalignment` or `all_reads`. It generates reads and contigs FASTA files.

Value

The output directory of FASTA files.

Examples

```
data("sangerAlignmentData")
writeFastaSA(sangerAlignmentData)
```

`sangerAlignmentData` *SangerAlignment instance*

Description

A pre-built `SangerAlignment` S4 object aggregating four contigs from the ACHLO fixture.

Usage

```
data(sangerAlignmentData)
```

Format

A `SangerAlignment-class` S4 object containing `contigList`, `contigsAlignment`, `contigsConsensus`, and a `contigsTree` `phylo` instance.

Author(s)

Kuan-Hao Chao

`sangeranalyseR` *sangeranalyseR-package*

Description

sangeranalyseR-package

SangerContig *SangerContig*

Description

the wrapper function for SangerContig

Usage

```
SangerContig(
  printLevel = "SangerContig",
  inputSource = "ABIF",
  processMethod = "REGEX",
  ABIF_Directory = NULL,
  FASTA_File = NULL,
  REGEX_SuffixForward = NULL,
  REGEX_SuffixReverse = NULL,
  CSV_NamesConversion = NULL,
  contigName = NULL,
  geneticCode = GENETIC_CODE,
  TrimmingMethod = "M1",
  M1TrimmingCutoff = 1e-04,
  M2CutoffQualityScore = NULL,
  M2SlidingWindowSize = NULL,
  baseNumPerRow = 100,
  heightPerRow = 200,
  signalRatioCutoff = 0.33,
  showTrimmed = TRUE,
  refAminoAcidSeq = "",
  minReadsNum = 2,
  minReadLength = 20,
  minFractionCall = 0.5,
  maxFractionLost = 0.5,
  acceptStopCodons = TRUE,
  readingFrame = 1,
  processorsNum = 1,
  BPPARAM = NULL,
  lazyAA = TRUE,
  minOverlapFraction = 0,
  minOverlapBases = 0L,
  alignSeqsParams = list(),
  consensusMethod = "strict",
  qualityAware = FALSE
)
```

Arguments

`printLevel`	Internal — controls log verbosity when this constructor is called recursively from a parent class. Defaults to `"SangerContig"`; do not set manually.
`inputSource`	The input source of the raw file. It must be `"ABIF"` or `"FASTA"`. The default value is `"ABIF"`.

- `processMethod` Either `"REGEX"` or `"CSV"`. Default `"REGEX"`.
- `ABIF_Directory` The parent directory of all of the reads contained in ABIF format you wish to analyse. In SangerContig, all reads must be in the first layer in this directory.
- `FASTA_File` If `inputSource` is `"FASTA"`, then this value has to be the name of the FASTA file; if `inputSource` is `"ABIF"`, then this value is `""` by default.
- `REGEX_SuffixForward` The suffix of the filenames for forward reads in regular expression, i.e. reads that do not need to be reverse-complemented. For forward reads, it should be `"_F.ab1"`.
- `REGEX_SuffixReverse` The suffix of the filenames for reverse reads in regular expression, i.e. reads that need to be reverse-complemented. For revcerse reads, it should be `"_R.ab1"`.
- `CSV_NamesConversion` The file path to the CSV file that provides read names that follow the naming regulation. If `inputSource` is `"FASTA"`, then users need to prepare the csv file or make sure the original names inside FASTA file are valid; if `inputSource` is `"ABIF"`, then this value is `NULL` by default.
- `contigName` The contig name of all the reads in `ABIF_Directory`.
- `geneticCode` Named character vector in the same format as `GENETIC_CODE` (the default), which represents the standard genetic code. This is the code with which the function will attempt to translate your DNA sequences. You can get an appropriate vector with the getGeneticCode() function. The default is the standard code.
- `TrimmingMethod` TrimmingMethod The read trimming method for this SangerRead. The value must be `"M1"` (the default) or `'M2'`.
- `M1TrimmingCutoff` The trimming cutoff for the Method 1. If `TrimmingMethod` is `"M1"`, then the default value is `0.0001`. Otherwise, the value must be `NULL`.
- `M2CutoffQualityScore` The trimming cutoff quality score for the Method 2. If `TrimmingMethod` is `'M2'`, then the default value is `20`. Otherwise, the value must be `NULL`. It works with `M2SlidingWindowSize`.
- `M2SlidingWindowSize` The trimming sliding window size for the Method 2. If `TrimmingMethod` is `'M2'`, then the default value is `10`. Otherwise, the value must be `NULL`. It works with `M2CutoffQualityScore`.
- `baseNumPerRow` It defines maximum base pairs in each row. The default value is `100`.
- `heightPerRow` It defines the height of each row in chromatogram. The default value is `200`.
- `signalRatioCutoff` The ratio of the height of a secondary peak to a primary peak. Secondary peaks higher than this ratio are annotated. Those below the ratio are excluded. The default value is `0.33`.
- `showTrimmed` The logical value storing whether to show trimmed base pairs in chromatogram. The default value is `TRUE`.
- `refAminoAcidSeq` An amino acid reference sequence supplied as a string or an AAString object. If your sequences are protein-coding DNA seuqences, and you want to have frameshifts automatically detected and corrected, supply a reference amino acid sequence via this argument. If this argument is supplied, the sequences are then

kept in frame for the alignment step. Fwd sequences are assumed to come from the sense (i.e. coding, or "+") strand. The default value is "".

`minReadsNum` The minimum number of reads required to make a consensus sequence, must be 2 or more. The default value is 2.

`minReadLength` Reads shorter than this will not be included in the readset. The default 20 means that all reads with length of 20 or more will be included. Note that this is the length of a read after it has been trimmed.

`minFractionCall` Minimum fraction of the sequences required to call a consensus sequence for SangerContig at any given position (see the ConsensusSequence() function from DECIPHER for more information). Defaults to 0.75 implying that 3/4 of all reads must be present in order to call a consensus.

`maxFractionLost` Numeric giving the maximum fraction of sequence information that can be lost in the consensus sequence for SangerContig (see the ConsensusSequence() function from DECIPHER for more information). Defaults to 0.5, implying that each consensus base can ignore at most 50 percent of the information at a given position.

`acceptStopCodons` The logical value TRUE or FALSE. TRUE (the defualt): keep all reads, regardless of whether they have stop codons; FALSE: reject reads with stop codons. If FALSE is selected, then the number of stop codons is calculated after attempting to correct frameshift mutations (if applicable).

`readingFrame` 1, 2, or 3. Only used if `accept.stop.codons == FALSE`. This specifies the reading frame that is used to determine stop codons. If you use a `refAminoAcidSeq`, then the frame should always be 1, since all reads will be shifted to frame 1 during frameshift correction. Otherwise, you should select the appropriate reading frame.

`processorsNum` The number of processors to use, or NULL (the default) for all available processors.

`BPPARAM` A `BiocParallelParam` instance for the per-read parallel loop. Default NULL (derived from `processorsNum`).

`lazyAA` Logical (default TRUE). Skip eager 3-frame AA translation when no `refAminoAcidSeq` is supplied; use the `primaryAASeqS1/S2/S3()` accessors on demand instead.

`minOverlapFraction` Numeric in [0, 1] (default `0.0`). Triggers a `LOW_OVERLAP_WARN` when the smallest pairwise non-gap overlap is below `minOverlapFraction * shorter_read_length`. See SangerAlignment for full discussion.

`minOverlapBases` Integer (default `0L`). Absolute-base-pair threshold variant of `minOverlapFraction`.

`alignSeqsParams` A named list (default `list()`) of additional arguments forwarded to `DECIPHER::AlignSeqs`.

`consensusMethod` One of `"strict"` (default), `"majority"`, or `"quality_weighted"`. See SangerAlignment for full discussion.

`qualityAware` Logical shorthand for `consensusMethod = "quality_weighted"`. Issue #48.

Value

A SangerContig instance.

Author(s)

Kuan-Hao Chao

Examples

```
rawDataDir <- system.file("extdata", package = "sangeranalyseR")
parentDir <- file.path(rawDataDir, "Allolobophora_chlorotica", "ACHLO")
contigName <- "Achl_ACHLO006-09"
REGEX_SuffixForward <- "_F.ab1"
REGEX_SuffixReverse <- "_R.ab1"
sangerContig <- SangerContig(
                     inputSource           = "ABIF",
                     ABIF_Directory       = parentDir,
                     contigName            = contigName,
                     REGEX_SuffixForward   = REGEX_SuffixForward,
                     REGEX_SuffixReverse   = REGEX_SuffixReverse,
              refAminoAcidSeq = "SRQWLFSTNHKDIGTLYFIFGAWAGMVGTSLSILIRAELGHPGALIGDDQIYNVIVTAHAFIMIFFMVMPIMIG
                     TrimmingMethod        = "M2",
                     M1TrimmingCutoff      = NULL,
                     M2CutoffQualityScore  = 20,
                     M2SlidingWindowSize   = 10,
                     baseNumPerRow         = 100,
                     heightPerRow          = 200,
                     signalRatioCutoff     = 0.33,
                     showTrimmed           = TRUE,
                     processorsNum         = 2)
```

SangerContig-class   *SangerContig*

Description

An S4 class containing forward and reverse SangerRead lists and alignment, consensus read results which corresponds to a contig in Sanger sequencing.

Slots

`objectResults` This is the object that stores all information of the creation result.

`inputSource` The input source of the raw file. It must be `"ABIF"` or `"FASTA"`. The default value is `"ABIF"`.

`processMethod` The method to create a contig from reads. The value is `"REGEX"` or `"CSV"`. The default value is `"REGEX"`.

`ABIF_Directory` If `inputSource` is `"ABIF"`, then this value is the path of a parent directory storing all reads in ABIF format you want to analyse. If `inputSource` is `"FASTA"`, then this value has to be `NULL` by default.

`FASTA_File` If `inputSource` is `"FASTA"`, then this value has to be the path to a valid FASTA file ; if `inputSource` is `"ABIF"`, then this value has to be `NULL` by default.

`REGEX_SuffixForward` The suffix of the filenames for forward reads in regular expression, i.e. reads that do not need to be reverse-complemented.

`REGEX_SuffixReverse` The suffix of the filenames for reverse reads in regular expression, i.e. reads that need to be reverse-complemented.

CSV_NamesConversion The file path to the CSV file that provides read names, directions, and their contig groups. If processMethod is "CSV", then this value has to be the path to a valid CSV file; if processMethod is "REGEX", then this value has to be NULL by default.

contigName The contig name of all the reads in ABIF_Directory.

geneticCode Named character vector in the same format as GENETIC_CODE (the default), which represents the standard genetic code. This is the code with which the function will attempt to translate your DNA sequences. You can get an appropriate vector with the getGeneticCode() function. The default is the standard code.

forwardReadList The list of SangerRead S4 instances which are all forward reads.

reverseReadList The list of SangerRead S4 instances which are all reverse reads.

minReadsNum The minimum number of reads required to make a consensus sequence, must be 2 or more. The default value is 2.

minReadLength Reads shorter than this will not be included in the readset. The default 20 means that all reads with length of 20 or more will be included. Note that this is the length of a read after it has been trimmed.

refAminoAcidSeq An amino acid reference sequence supplied as a string or an AAString object. If your sequences are protein-coding DNA seuqences, and you want to have frameshifts automatically detected and corrected, supply a reference amino acid sequence via this argument. If this argument is supplied, the sequences are then kept in frame for the alignment step. Fwd sequences are assumed to come from the sense (i.e. coding, or "+") strand. The default value is "".

minFractionCall Minimum fraction of the sequences required to call a consensus sequence for SangerContig at any given position (see the ConsensusSequence() function from DECIPHER for more information). Defaults to 0.75 implying that 3/4 of all reads must be present in order to call a consensus.

maxFractionLost Numeric giving the maximum fraction of sequence information that can be lost in the consensus sequence for SangerContig (see the ConsensusSequence() function from DECIPHER for more information). Defaults to 0.5, implying that each consensus base can ignore at most 50 percent of the information at a given position.

acceptStopCodons The logical value TRUE or FALSE. TRUE (the defualt): keep all reads, regardless of whether they have stop codons; FALSE: reject reads with stop codons. If FALSE is selected, then the number of stop codons is calculated after attempting to correct frameshift mutations (if applicable).

readingFrame 1, 2, or 3. Only used if accept.stop.codons == FALSE. This specifies the reading frame that is used to determine stop codons. If you use a refAminoAcidSeq, then the frame should always be 1, since all reads will be shifted to frame 1 during frameshift correction. Otherwise, you should select the appropriate reading frame.

contigSeq The consensus read of all SangerRead S4 instances in DNAString object.

alignment The alignment of all SangerRead S4 instances with the called consensus sequence in DNAStringSet object. Users can use BrowseSeqs() to view the alignment.

differencesDF A data frame of the number of pairwise differences between each read and the consensus sequence, as well as the number of bases in each input read that did not contribute to the consensus sequence. It can assist in detecting incorrect reads, or reads with a lot of errors.

distanceMatrix A distance matrix of genetic distances (corrected with the JC model) between all of the input reads.

dendrogram A list storing cluster groups in a data frame and a dendrogram object depicting the distance.matrix. Users can use plot() to see the dendrogram.

`indelsDF` If users specified a reference sequence via `refAminoAcidSeq`, then this will be a data frame describing the number of indels and deletions that were made to each of the input reads in order to correct frameshift mutations.

`stopCodonsDF` If users specified a reference sequence via `refAminoAcidSeq`, then this will be a data frame describing the number of stop codons in each read.

`secondaryPeakDF` A data frame with one row for each column in the alignment that contained more than one secondary peak. The data frame has three columns: the column number of the alignment; the number of secondary peaks in that column; and the bases (with IUPAC ambiguity codes representing secondary peak calls) in that column represented as a string.

Author(s)

Kuan-Hao Chao

Examples

```
## Simple example
rawDataDir <- system.file("extdata", package = "sangeranalyseR")
parentDir <- file.path(rawDataDir, "Allolobophora_chlorotica", "RBNII")
contigName <- "Achl_RBNII384-13"
REGEX_SuffixForward <- "_[0-9]*_F.ab1$"
REGEX_SuffixReverse <- "_[0-9]*_R.ab1$"
sangerContig <- new("SangerContig",
                     ABIF_Directory       = parentDir,
                     contigName            = contigName,
                     REGEX_SuffixForward   = REGEX_SuffixForward,
                     REGEX_SuffixReverse   = REGEX_SuffixReverse)

## forward / reverse reads match error
## Input From ABIF file format (Regex)
rawDataDir <- system.file("extdata", package = "sangeranalyseR")
parentDir <- file.path(rawDataDir, "Allolobophora_chlorotica", "ACHLO")
contigName <- "Achl_ACHLO006-09"
REGEX_SuffixForward <- "_[0-9]*_F.ab1$"
REGEX_SuffixReverse <- "_[0-9]*_R.ab1$"
sangerContig <- new("SangerContig",
                     inputSource           = "ABIF",
                     processMethod         = "REGEX",
                     ABIF_Directory       = parentDir,
                     contigName            = contigName,
                     REGEX_SuffixForward   = REGEX_SuffixForward,
                     REGEX_SuffixReverse   = REGEX_SuffixReverse,
              refAminoAcidSeq = "SRQWLFSTNHKDIGTLYFIFGAWAGMVGTSLSILIRAELGHPGALIGDDQIYNVIVTAHAFIMIFFMVMPIMIG
                     TrimmingMethod        = "M1",
                     M1TrimmingCutoff      = 0.0001,
                     baseNumPerRow         = 100,
                     heightPerRow          = 200,
                     signalRatioCutoff     = 0.33,
                     showTrimmed           = TRUE,
                     minReadsNum           = 2,
                     processorsNum         = 2)

## Input From ABIF file format (Csv three column method)
rawDataDir <- system.file("extdata", package = "sangeranalyseR")
parentDir <- file.path(rawDataDir, "Allolobophora_chlorotica", "RBNII")
CSV_NamesConversion <- file.path(rawDataDir, "ab1", "SangerContig", "names_conversion_2.csv")
```

```
sangerContig <- new("SangerContig",
                     inputSource           = "ABIF",
                     processMethod         = "CSV",
                     ABIF_Directory        = parentDir,
                     CSV_NamesConversion   = CSV_NamesConversion,
                     contigName            = "Achl_RBNII384-13",
              refAminoAcidSeq = "SRQWLFSTNHKDIGTLYFIFGAWAGMVGTSLSILIRAELGHPGALIGDDQIYNVIVTAHAFIMIFFMVMPIMIG
                     TrimmingMethod        = "M1",
                     M1TrimmingCutoff      = 0.000001,
                     baseNumPerRow         = 100,
                     heightPerRow          = 200,
                     signalRatioCutoff     = 0.33,
                     showTrimmed           = TRUE,
                     processorsNum         = 2)


## Input From FASTA file format (Regex)
rawDataDir <- system.file("extdata", package = "sangeranalyseR")
fastaFN <- file.path(rawDataDir, "fasta",
                     "SangerContig", "Achl_ACHLO006-09.fa")
contigName <- "Achl_ACHLO006-09"
REGEX_SuffixForwardFa <- "_[0-9]*_F$"
REGEX_SuffixReverseFa <- "_[0-9]*_R$"
sangerContigFa <- new("SangerContig",
                      inputSource           = "FASTA",
                      processMethod         = "REGEX",
                      FASTA_File         = fastaFN,
                      contigName            = contigName,
                      REGEX_SuffixForward   = REGEX_SuffixForwardFa,
                      REGEX_SuffixReverse   = REGEX_SuffixReverseFa,
               refAminoAcidSeq     = "SRQWLFSTNHKDIGTLYFIFGAWAGMVGTSLSILIRAELGHPGALIGDDQIYNVIVTAHAFIMIFFMVMP
                      processorsNum         = 2)

## Input From FASTA file format (Csv - Csv three column method)
rawDataDir <- system.file("extdata", package = "sangeranalyseR")
fastaFN <- file.path(rawDataDir, "fasta",
                     "SangerContig", "Achl_ACHLO006-09.fa")
CSV_NamesConversion <- file.path(rawDataDir, "fasta", "SangerContig", "names_conversion_1.csv")
sangerContigFa <- new("SangerContig",
                      inputSource           = "FASTA",
                      processMethod         = "CSV",
                      FASTA_File         = fastaFN,
                      CSV_NamesConversion    = CSV_NamesConversion,
                      contigName            = "Achl_ACHLO006-09",
               refAminoAcidSeq     = "SRQWLFSTNHKDIGTLYFIFGAWAGMVGTSLSILIRAELGHPGALIGDDQIYNVIVTAHAFIMIFFMVMP
                      processorsNum         = 2)
```

SangerContig-class-generateReportSC
generateReportSC

Description

A SangerContig method which generates final reports of the SangerContig instance.

Usage

```
## S4 method for signature 'SangerContig'
generateReportSC(
  object,
  outputDir,
  includeSangerRead = TRUE,
  colors,
  navigationAlignmentFN = NULL
)
```

Arguments

`object`	A SangerContig S4 instance.
`outputDir`	The output directory of the generated HTML report.
`includeSangerRead`	The parameter that decides whether to include SangerRead level report. The value is `TRUE` or `FALSE` and the default is `TRUE`.
`colors`	A vector for users to set the colors of (A, T, C, G, else). There are three options for users to choose from. 1. "default": (green, blue, black, red, purple). 2. "cb_friendly": ((0, 0, 0), (199, 199, 199), (0, 114, 178), (213, 94, 0), (204, 121, 167)). 3. Users can set their own colors with a vector with five elements.
`navigationAlignmentFN`	The internal parameter passed to HTML report. Users should not modify this parameter on their own.

Value

The output absolute path to the SangerContig's HTML file.

Examples

```
data("sangerContigData")

generateReportSC(sangerContigData)
generateReportSC(sangerContigData, colors="cb_friendly")
```

SangerContig-class-launchAppSC *launchAppSC*

Description

A SangerContig method which launches Shiny app for SangerContig instance.

Usage

```
## S4 method for signature 'SangerContig'
launchAppSC(object, outputDir = NULL, colors = "default")
```

Arguments

`object`	A SangerContig S4 instance.
`outputDir`	The output directory of the saved new SangerContig S4 instance.
`colors`	A vector for users to set the colors of (A, T, C, G, else). There are three options for users to choose from. 1. "default": (green, blue, black, red, purple). 2. "cb_friendly": ((0, 0, 0), (199, 199, 199), (0, 114, 178), (213, 94, 0), (204, 121, 167)). 3. Users can set their own colors with a vector with five elements.

Value

A `shiny.appobj` object.

Examples

```
data("sangerContigData")
RShinySC <- launchAppSC(sangerContigData)
RShinySC <- launchAppSC(sangerContigData, colors="cb_friendly")
```

SangerContig-class-readTable

readTable

Description

A SangerContig method which generates summary table for SangerContig instance

Usage

```
## S4 method for signature 'SangerContig'
readTable(object, indentation = 0)
```

Arguments

`object`	A SangerContig S4 instance.
`indentation`	The indentation for different level printing.

Value

None

Examples

```
data(sangerReadFData)
data(sangerContigData)

readTable(sangerReadFData)
readTable(sangerContigData)
```

SangerContig-class-updateQualityParam

updateQualityParam

Description

A SangerContig method which updates QualityReport parameter for each the SangerRead instance inside SangerContig.

Usage

```
## S4 method for signature 'SangerContig'
updateQualityParam(
  object,
  TrimmingMethod = "M1",
  M1TrimmingCutoff = 1e-04,
  M2CutoffQualityScore = NULL,
  M2SlidingWindowSize = NULL,
  processorsNum = NULL
)
```

Arguments

`object` A SangerContig S4 instance.

`TrimmingMethod` The read trimming method for this SangerRead. The value must be `"M1"` (the default) or `'M2'`.

`M1TrimmingCutoff` The trimming cutoff for the Method 1. If `TrimmingMethod` is `"M1"`, then the default value is `0.0001`. Otherwise, the value must be `NULL`.

`M2CutoffQualityScore` The trimming cutoff quality score for the Method 2. If `TrimmingMethod` is `'M2'`, then the default value is `20`. Otherwise, the value must be `NULL`. It works with `M2SlidingWindowSize`.

`M2SlidingWindowSize` The trimming sliding window size for the Method 2. If `TrimmingMethod` is `'M2'`, then the default value is `10`. Otherwise, the value must be `NULL`. It works with `M2CutoffQualityScore`.

`processorsNum` The number of processors to use, or NULL (the default) for all available processors.

Value

A SangerContig instance.

Examples

```
data("sangerContigData")

updateQualityParam(sangerContigData,
                   TrimmingMethod         = "M2",
```

```
M1TrimmingCutoff       = NULL,
M2CutoffQualityScore   = 40,
M2SlidingWindowSize    = 15)
```

SangerContig-class-writeFastaSC
writeFastaSC

Description

A SangerContig method which writes sequences into Fasta files.

Usage

```
## S4 method for signature 'SangerContig'
writeFastaSC(
  object,
  outputDir = NULL,
  compress = FALSE,
  compression_level = NA,
  selection = "all"
)
```

Arguments

`object`	A SangerContig S4 instance.
`outputDir`	The output directory of generated FASTA files.
`compress`	Like for the `save` function in base R, must be `TRUE` or `FALSE` (the default), or a single string specifying whether writing to the file is to use compression. The only type of compression supported at the moment is "gzip". This parameter will be passed to `writeXStringSet` function in Biostrings package.
`compression_level`	This parameter will be passed to `writeXStringSet` function in Biostrings package.
`selection`	This value can be `all`, `reads_alignment`, `reads_unalignment` or `contig`. It generates reads and the contig FASTA files.

Value

The output directory of FASTA files.

Examples

```
data("sangerContigData")
writeFastaSC(sangerContigData)
```

`sangerContigData` *SangerContig instance*

Description

A pre-built `SangerContig` S4 object containing one forward + one reverse `SangerRead` from the ACHLO fixture, plus the assembled contig consensus.

Usage

```
data(sangerContigData)
```

Format

A `SangerContig-class` S4 object.

Author(s)

Kuan-Hao Chao

`SangerRead` *SangerRead*

Description

the wrapper function for SangerRead

Usage

```
SangerRead(
  printLevel = "SangerRead",
  inputSource = "ABIF",
  readFeature = "",
  readFileName = "",
  fastaReadName = NULL,
  geneticCode = GENETIC_CODE,
  TrimmingMethod = "M1",
  M1TrimmingCutoff = 1e-04,
  M2CutoffQualityScore = NULL,
  M2SlidingWindowSize = NULL,
  baseNumPerRow = 100,
  heightPerRow = 200,
  signalRatioCutoff = 0.33,
  showTrimmed = TRUE,
  lazyAA = TRUE
)
```

Arguments

- `printLevel` Internal — controls log verbosity when this constructor is called recursively from a parent class. Defaults to `"SangerRead"`; do not set manually.
- `inputSource` The input source of the raw file. It must be `"ABIF"` or `"FASTA"`. The default value is `"ABIF"`.
- `readFeature` The direction of the Sanger read. The value must be `"Forward Read"` or `"Reverse Read"`.
- `readFileName` The filename of the target ABIF file.
- `fastaReadName` If `inputSource` is `"FASTA"`, then this value has to be the name of the read inside the FASTA file; if `inputSource` is `"ABIF"`, then this value is `""` by default.
- `geneticCode` Named character vector in the same format as `GENETIC_CODE` (the default), which represents the standard genetic code. This is the code with which the function will attempt to translate your DNA sequences. You can get an appropriate vector with the getGeneticCode() function. The default is the standard code.
- `TrimmingMethod` TrimmingMethod The read trimming method for this SangerRead. The value must be `"M1"` (the default) or `"M2"`. M1 is the modified Mott's trimming algorithm that can also be found in Phred/Phrap and Biopython. M2 is like trimmomatic's sliding window method.
- `M1TrimmingCutoff` The trimming cutoff for the Method 1. If `TrimmingMethod` is `"M1"`, then the default value is `0.0001`. Otherwise, the value must be `NULL`.
- `M2CutoffQualityScore` The trimming cutoff quality score for the Method 2. If `TrimmingMethod` is `'M2'`, then the default value is `20`. Otherwise, the value must be `NULL`. It works with `M2SlidingWindowSize`.
- `M2SlidingWindowSize` The trimming sliding window size for the Method 2. If `TrimmingMethod` is `'M2'`, then the default value is `10`. Otherwise, the value must be `NULL`. It works with `M2CutoffQualityScore`.
- `baseNumPerRow` It defines maximum base pairs in each row. The default value is `100`.
- `heightPerRow` It defines the height of each row in chromatogram. The default value is `200`.
- `signalRatioCutoff` The ratio of the height of a secondary peak to a primary peak. Secondary peaks higher than this ratio are annotated. Those below the ratio are excluded. The default value is `0.33`.
- `showTrimmed` The logical value storing whether to show trimmed base pairs in chromatogram. The default value is `TRUE`.
- `lazyAA` Logical (default `TRUE`). Skip eager 3-frame AA translation; use `primaryAASeqS1/S2/S3()` accessors instead.

Value

A SangerRead instance.

Author(s)

Kuan-Hao Chao

Examples

```
inputFilesPath <- system.file("extdata/", package = "sangeranalyseR")
A_chloroticaFdFN <- file.path(inputFilesPath,
                              "Allolobophora_chlorotica",
                              "ACHLO",
                              "Achl_ACHLO006-09_1_F.ab1")
sangerRead <- SangerRead(
                   printLevel           = "SangerRead",
                   inputSource          = "ABIF",
                   readFeature          = "Forward Read",
                   readFileName         = A_chloroticaFdFN,
                   geneticCode          = GENETIC_CODE,
                   TrimmingMethod       = "M1",
                   M1TrimmingCutoff     = 0.0001,
                   M2CutoffQualityScore = NULL,
                   M2SlidingWindowSize  = NULL,
                   baseNumPerRow        = 100,
                   heightPerRow         = 200,
                   signalRatioCutoff    = 0.33,
                   showTrimmed          = TRUE)
```

`SangerRead-class` *SangerRead*

Description

An S4 class extending sangerseq S4 class which corresponds to a single ABIF file in Sanger sequencing.

Slots

`objectResults` This is the object that stores all information of the creation result.

`inputSource` The input source of the raw file. It must be `"ABIF"` or `"FASTA"`. The default value is `"ABIF"`.

`readFeature` The direction of the Sanger read. The value must be `"Forward Read"` or `"Reverse Read"`.

`readFileName` The filename of the target input file.

`fastaReadName` If `inputSource` is `"FASTA"`, then this value has to be the name of the read inside the FASTA file; if `inputSource` is `"ABIF"`, then this value is `NULL` by default.

`geneticCode` Named character vector in the same format as `GENETIC_CODE` (the default), which represents the standard genetic code. This is the code with which the function will attempt to translate your DNA sequences. You can get an appropriate vector with the getGeneticCode() function. The default is the standard code.

`abifRawData` An S4 class containing all fields in the ABIF file. It is the abif class defined in sangerseqR package.

`QualityReport` A S4 class containing quality trimming related inputs and trimming results.

`ChromatogramParam` A S4 class containing chromatogram inputs.

`primaryAASeqS1` A polypeptide translated from primary DNA sequence starting from the first nucleic acid.

primaryAASeqS2 A polypeptide translated from primary DNA sequence starting from the second nucleic acid.

primaryAASeqS3 A polypeptide translated from primary DNA sequence starting from the third nucleic acid.

primarySeqRaw The raw primary sequence from sangerseq class in sangerseqR package before base calling.

secondarySeqRaw The raw secondary sequence from sangerseq class in sangerseqR package before base calling.

peakPosMatrixRaw The raw peak position matrix from sangerseq class in sangerseqR package before base calling.

peakAmpMatrixRaw The raw peak amplitude matrix from sangerseq class in sangerseqR package before base calling.

Author(s)

Kuan-Hao Chao

Examples

```
## Simple example
inputFilesPath <- system.file("extdata/", package = "sangeranalyseR")
A_chloroticaFFN <- file.path(inputFilesPath,
                             "Allolobophora_chlorotica",
                             "ACHLO",
                             "Achl_ACHLO006-09_1_F.ab1")
sangerReadF <- new("SangerRead",
                    readFeature           = "Forward Read",
                    readFileName          = A_chloroticaFFN)

## Input From ABIF file format
# Forward Read
A_chloroticaFFN <- file.path(inputFilesPath,
                             "Allolobophora_chlorotica",
                             "ACHLO",
                             "Achl_ACHLO006-09_1_F.ab1")
sangerReadF <- new("SangerRead",
                    printLevel            = "SangerRead",
                    inputSource           = "ABIF",
                    readFeature           = "Forward Read",
                    readFileName          = A_chloroticaFFN,
                    fastaReadName         = NULL,
                    geneticCode           = GENETIC_CODE,
                    TrimmingMethod        = "M1",
                    M1TrimmingCutoff      = 0.0001,
                    M2CutoffQualityScore  = NULL,
                    M2SlidingWindowSize   = NULL,
                    baseNumPerRow         = 100,
                    heightPerRow          = 200,
                    signalRatioCutoff     = 0.33,
                    showTrimmed           = TRUE)

# Reverse Read
A_chloroticaRFN <- file.path(inputFilesPath,
                             "Allolobophora_chlorotica",
```

```
                             "ACHLO",
                             "Achl_ACHLO006-09_2_R.ab1")
sangerReadR <- new("SangerRead",
                    inputSource           = "ABIF",
                    readFeature           = "Reverse Read",
                    readFileName          = A_chloroticaRFN,
                    geneticCode           = GENETIC_CODE,
                    TrimmingMethod        = "M1",
                    M1TrimmingCutoff      = 0.0001,
                    M2CutoffQualityScore  = NULL,
                    M2SlidingWindowSize   = NULL,
                    baseNumPerRow         = 100,
                    heightPerRow          = 200,
                    signalRatioCutoff     = 0.33,
                    showTrimmed           = TRUE)


## Input From FASTA file format
# Forward Read
inputFilesPath <- system.file("extdata/", package = "sangeranalyseR")
A_chloroticaFFNfa <- file.path(inputFilesPath,
                               "fasta",
                               "SangerRead",
                               "Achl_ACHLO006-09_1_F.fa")
readNameFfa <- "Achl_ACHLO006-09_1_F"
sangerReadFfa <- new("SangerRead",
                     inputSource        = "FASTA",
                     readFeature        = "Forward Read",
                     readFileName       = A_chloroticaFFNfa,
                     fastaReadName      = readNameFfa,
                     geneticCode        = GENETIC_CODE)
# Reverse Read
A_chloroticaRFNfa <- file.path(inputFilesPath,
                               "fasta",
                               "SangerRead",
                               "Achl_ACHLO006-09_2_R.fa")
readNameRfa <- "Achl_ACHLO006-09_2_R"
sangerReadRfa <- new("SangerRead",
                     inputSource   = "FASTA",
                     readFeature   = "Reverse Read",
                     readFileName  = A_chloroticaRFNfa,
                     fastaReadName = readNameRfa,
                     geneticCode   = GENETIC_CODE)
```

SangerRead-class-generateReportSR *generateReportSR*

Description

A SangerRead method which generates final reports of the SangerRead instance.

Usage

```
## S4 method for signature 'SangerRead'
generateReportSR(
  object,
  outputDir,
  colors,
  navigationContigFN = NULL,
  navigationAlignmentFN = NULL
)
```

Arguments

`object` A SangerRead S4 instance.

`outputDir` The output directory of the generated HTML report.

`colors` A vector for users to set the colors of (A, T, C, G, else). There are three options for users to choose from. 1. "default": (green, blue, black, red, purple). 2. "cb_friendly": ((0, 0, 0), (199, 199, 199), (0, 114, 178), (213, 94, 0), (204, 121, 167)). 3. Users can set their own colors with a vector with five elements.

`navigationContigFN` The internal parameter passed to HTML report. Users should not modify this parameter on their own.

`navigationAlignmentFN` The internal parameter passed to HTML report. Users should not modify this parameter on their own.

Value

The output absolute path to the SangerRead's HTML file.

Examples

```
data("sangerReadFData")

generateReportSR(sangerReadFData, "~/Documents")
generateReportSR(sangerReadFData, colors="cb_friendly")
```

SangerRead-class-MakeBaseCalls

MakeBaseCalls

Description

A SangerRead method which does base calling on SangerRead instance

Usage

```
## S4 method for signature 'SangerRead'
MakeBaseCalls(object, signalRatioCutoff = 0.33)
```

Arguments

`object` A SangerRead S4 instance.

`signalRatioCutoff`
: The ratio of the height of a secondary peak to a primary peak. Secondary peaks higher than this ratio are annotated. Those below the ratio are excluded. The default value is `0.33`.

Value

A `SangerRead` instance.

Examples

```
data("sangerReadFData")
newSangerReadFData <- MakeBaseCalls(sangerReadFData, signalRatioCutoff = 0.22)
```

SangerRead-class-qualityBasePlot
qualityBasePlot

Description

A SangerRead method which creates quality base interactive plot.

Usage

```
## S4 method for signature 'SangerRead'
qualityBasePlot(object)
```

Arguments

`object` A SangerRead S4 instance.

Value

A quality plot.

Examples

```
data("sangerReadFData")

qualityBasePlot(sangerReadFData)
```

SangerRead-class-readTable

readTable

Description

A SangerRead method which generates summary table for SangerRead instance

Usage

```
## S4 method for signature 'SangerRead'
readTable(object, indentation = 0)
```

Arguments

object	A SangerRead S4 instance.
indentation	The indentation for different level printing.

Value

None

Examples

```
data(sangerReadFData)
data(sangerContigData)

readTable(sangerReadFData)
readTable(sangerContigData)
```

SangerRead-class-updateQualityParam

updateQualityParam

Description

A SangerRead method which updates QualityReport parameter inside the SangerRead.

Usage

```
## S4 method for signature 'SangerRead'
updateQualityParam(
  object,
  TrimmingMethod = "M1",
  M1TrimmingCutoff = 1e-04,
  M2CutoffQualityScore = NULL,
  M2SlidingWindowSize = NULL
)
```

Arguments

`object` A SangerRead S4 instance.

`TrimmingMethod` The read trimming method for this SangerRead. The value must be "M1" (the default) or 'M2'.

`M1TrimmingCutoff` The trimming cutoff for the Method 1. If `TrimmingMethod` is "M1", then the default value is `0.0001`. Otherwise, the value must be `NULL`.

`M2CutoffQualityScore` The trimming cutoff quality score for the Method 2. If `TrimmingMethod` is 'M2', then the default value is `20`. Otherwise, the value must be `NULL`. It works with `M2SlidingWindowSize`.

`M2SlidingWindowSize` The trimming sliding window size for the Method 2. If `TrimmingMethod` is 'M2', then the default value is `10`. Otherwise, the value must be `NULL`. It works with `M2CutoffQualityScore`.

Value

A SangerRead instance.

Examples

```
data("sangerReadFData")
updateQualityParam(sangerReadFData,
                   TrimmingMethod         = "M2",
                   M1TrimmingCutoff       = NULL,
                   M2CutoffQualityScore   = 40,
                   M2SlidingWindowSize    = 15)
```

SangerRead-class-writeFastaSR

writeFastaSR

Description

A SangerRead method which writes the sequence into Fasta files.

Usage

```
## S4 method for signature 'SangerRead'
writeFastaSR(
  object,
  outputDir = NULL,
  compress = FALSE,
  compression_level = NA
)
```

Arguments

- `object` — A SangerRead S4 instance.
- `outputDir` — The output directory of the generated FASTA file.
- `compress` — Like for the `save` function in base R, must be `TRUE` or `FALSE` (the default), or a single string specifying whether writing to the file is to use compression. The only type of compression supported at the moment is "gzip". This parameter will be passed to `writeXStringSet` function in Biostrings package.
- `compression_level` — This parameter will be passed to `writeXStringSet` function in Biostrings package.

Value

The output absolute path to the FASTA file.

Examples

```
data("sangerReadFData")
writeFastaSR(sangerReadFData)
```

`sangerReadFData` *SangerRead instance*

Description

A pre-built `SangerRead` S4 object for one forward ABIF read from the ACHLO fixture.

Usage

```
data(sangerReadFData)
```

Format

A `SangerRead-class` S4 object with populated `primarySeq`, `secondarySeq`, `traceMatrix`, and nested `QualityReport` / `ChromatogramParam`.

Author(s)

Kuan-Hao Chao

updateQualityParam *Method updateQualityParam*

Description

Method updateQualityParam

Usage

```
updateQualityParam(
  object,
  TrimmingMethod = "M1",
  M1TrimmingCutoff = 1e-04,
  M2CutoffQualityScore = NULL,
  M2SlidingWindowSize = NULL,
  ...
)
```

Arguments

- `object` A QualityReport, SangerRead, SangerContig, or SangerAlignment S4 instance.
- `TrimmingMethod` The read trimming method for this SangerRead. The value must be `"M1"` (the default) or `'M2'`.
- `M1TrimmingCutoff` The trimming cutoff for the Method 1. If `TrimmingMethod` is `"M1"`, then the default value is `0.0001`. Otherwise, the value must be `NULL`.
- `M2CutoffQualityScore` The trimming cutoff quality score for the Method 2. If `TrimmingMethod` is `'M2'`, then the default value is `20`. Otherwise, the value must be `NULL`. It works with `M2SlidingWindowSize`.
- `M2SlidingWindowSize` The trimming sliding window size for the Method 2. If `TrimmingMethod` is `'M2'`, then the default value is `10`. Otherwise, the value must be `NULL`. It works with `M2CutoffQualityScore`.
- `...` Further updateQualityParam-related parameters.

Value

A `QualityReport`, `SangerRead`, `SangerContig`, or `SangerAlignment` instance.

Examples

```
data(qualityReportData)
data(sangerReadFData)
data(sangerContigData)
data(sangerAlignmentData)

updateQualityParam(qualityReportData,
                   TrimmingMethod         = "M2",
                   M1TrimmingCutoff       = NULL,
                   M2CutoffQualityScore   = 40,
```

```
                     M2SlidingWindowSize    = 15)
updateQualityParam(sangerReadFData,
                     TrimmingMethod         = "M2",
                     M1TrimmingCutoff       = NULL,
                     M2CutoffQualityScore   = 40,
                     M2SlidingWindowSize    = 15)
updateQualityParam(sangerContigData,
                     TrimmingMethod         = "M2",
                     M1TrimmingCutoff       = NULL,
                     M2CutoffQualityScore   = 40,
                     M2SlidingWindowSize    = 15)
updateQualityParam(sangerAlignmentData,
                     TrimmingMethod         = "M2",
                     M1TrimmingCutoff       = NULL,
                     M2CutoffQualityScore   = 40,
                     M2SlidingWindowSize    = 15)
```

---

writeFasta *Method writeFasta*

Description

A method which writes FASTA files of the SangerRead, SangerContig, and SangerAlignment instance.

Usage

```
writeFasta(
  object,
  outputDir = NULL,
  compress = FALSE,
  compression_level = NA,
  selection = "all"
)
```

Arguments

`object`	A SangerRead, SangerContig, or SangerAlignment S4 instance.
`outputDir`	The output directory of generated FASTA files.
`compress`	Like for the `save` function in base R, must be `TRUE` or `FALSE` (the default), or a single string specifying whether writing to the file is to use compression. The only type of compression supported at the moment is "gzip". This parameter will be passed to `writeXStringSet` function in Biostrings package.
`compression_level`	This parameter will be passed to `writeXStringSet` function in Biostrings package.
`selection`	This parameter will be passed to `writeFastaSC` or `writeFastaSA`.

Value

A `SangerRead`, `SangerContig`, or `SangerAlignment` object.

Author(s)

Kuan-Hao Chao

Examples

```
data(sangerReadFData)
data(sangerContigData)
data(sangerAlignmentData)

writeFasta(sangerReadFData)
writeFasta(sangerContigData)
writeFasta(sangerAlignmentData)
```

---

writeFastaSA *Method writeFastaSA*

Description

Method writeFastaSA

Usage

```
writeFastaSA(
  object,
  outputDir = NULL,
  compress = FALSE,
  compression_level = NA,
  selection = "all"
)
```

Arguments

object	A SangerAlignment S4 instance.
outputDir	The output directory of generated FASTA files.
compress	Like for the save function in base R, must be TRUE or FALSE (the default), or a single string specifying whether writing to the file is to use compression. The only type of compression supported at the moment is "gzip". This parameter will be passed to writeXStringSet function in Biostrings package.
compression_level	This parameter will be passed to writeXStringSet function in Biostrings package.
selection	This value can be all, contigs_alignment, contigs_unalignment or all_reads. It generates reads and contigs FASTA files.

Value

The output directory of FASTA files.

Examples

```
data(sangerAlignmentData)
writeFastaSA(sangerAlignmentData)
```

`writeFastaSC` *Method writeFastaSC*

Description

Method writeFastaSC

Usage

```
writeFastaSC(
  object,
  outputDir = NULL,
  compress = FALSE,
  compression_level = NA,
  selection = "all"
)
```

Arguments

`object`	A SangerContig S4 instance.
`outputDir`	The output directory of generated FASTA files.
`compress`	Like for the `save` function in base R, must be `TRUE` or `FALSE` (the default), or a single string specifying whether writing to the file is to use compression. The only type of compression supported at the moment is "gzip". This parameter will be passed to `writeXStringSet` function in Biostrings package.
`compression_level`	This parameter will be passed to `writeXStringSet` function in Biostrings package.
`selection`	This value can be `all`, `reads_alignment`, `reads_unalignment` or `contig`. It generates reads and the contig FASTA files.

Value

The output directory of FASTA files.

Examples

```
data(sangerContigData)
writeFastaSC(sangerContigData)
```

`writeFastaSR` *Method writeFastaSR*

Description

Method writeFastaSR

Usage

```
writeFastaSR(
  object,
  outputDir = NULL,
  compress = FALSE,
  compression_level = NA
)
```

Arguments

- `object` — A SangerRead S4 instance.
- `outputDir` — The output directory of the generated FASTA file.
- `compress` — Like for the `save` function in base R, must be `TRUE` or `FALSE` (the default), or a single string specifying whether writing to the file is to use compression. The only type of compression supported at the moment is "gzip". This parameter will be passed to `writeXStringSet` function in Biostrings package.
- `compression_level` — This parameter will be passed to `writeXStringSet` function in Biostrings package.

Value

The output absolute path to the FASTA file.

Examples

```
data(sangerReadFData)
writeFastaSR(sangerReadFData)
```

Index

∗ **datasets**
qualityReportData, 18
sangerAlignmentData, 29
sangerContigData, 41
sangerReadFData, 50

chromatogram_overwrite, 4, *5*
chromatogram_plotly, *4*, 5
ChromatogramParam-class, 3

generateReport, 6
generateReportSA, 7
generateReportSA,SangerAlignment-method (SangerAlignment-class-generateReportSA), 26
generateReportSC, 8
generateReportSC,SangerContig-method (SangerContig-class-generateReportSC), 36
generateReportSR, 9
generateReportSR,SangerRead-method (SangerRead-class-generateReportSR), 45
globalTrimApp, 9

launchApp, 10
launchAppSA, 11
launchAppSA,SangerAlignment-method (SangerAlignment-class-launchAppSA), 27
launchAppSC, 11
launchAppSC,SangerContig-method (SangerContig-class-launchAppSC), 37

MakeBaseCalls, 12
MakeBaseCalls,SangerRead-method (SangerRead-class-MakeBaseCalls), 46

ObjectResults-class, 12

primaryAASeqS1, 13
primaryAASeqS1,SangerRead-method (primaryAASeqS1), 13
primaryAASeqS2, 14
primaryAASeqS2,SangerRead-method (primaryAASeqS2), 14
primaryAASeqS3, 14
primaryAASeqS3,SangerRead-method (primaryAASeqS3), 14

qualityBasePlot, 15
qualityBasePlot,QualityReport-method (QualityReport-class-qualityBasePlot), 17
qualityBasePlot,SangerRead-method (SangerRead-class-qualityBasePlot), 47
QualityReport-class, 15
QualityReport-class-qualityBasePlot, 17
QualityReport-class-updateQualityParam, 17
qualityReportData, 18

readTable, 19
readTable,SangerContig-method (SangerContig-class-readTable), 38
readTable,SangerRead-method (SangerRead-class-readTable), 48

SangerAlignment, 19
SangerAlignment-class, 23
SangerAlignment-class-generateReportSA, 26
SangerAlignment-class-launchAppSA, 27
SangerAlignment-class-updateQualityParam, 27
SangerAlignment-class-writeFastaSA, 28
sangerAlignmentData, 29
sangeranalyseR, 29
SangerContig, 30
SangerContig-class, 33
SangerContig-class-generateReportSC, 36
SangerContig-class-launchAppSC, 37

SangerContig-class-readTable, 38
SangerContig-class-updateQualityParam,
39
SangerContig-class-writeFastaSC, 40
sangerContigData, 41
SangerRead, 41
SangerRead-class, 43
SangerRead-class-generateReportSR, 45
SangerRead-class-MakeBaseCalls, 46
SangerRead-class-qualityBasePlot, 47
SangerRead-class-readTable, 48
SangerRead-class-updateQualityParam,
48
SangerRead-class-writeFastaSR, 49
sangerReadFData, 50

updateQualityParam, 51
updateQualityParam,QualityReport-method
(QualityReport-class-updateQualityParam),
17
updateQualityParam,SangerAlignment-method
(SangerAlignment-class-updateQualityParam),
27
updateQualityParam,SangerContig-method
(SangerContig-class-updateQualityParam),
39
updateQualityParam,SangerRead-method
(SangerRead-class-updateQualityParam),
48

writeFasta, 52
writeFastaSA, 53
writeFastaSA,SangerAlignment-method
(SangerAlignment-class-writeFastaSA),
28
writeFastaSC, 54
writeFastaSC,SangerContig-method
(SangerContig-class-writeFastaSC),
40
writeFastaSR, 54
writeFastaSR,SangerRead-method
(SangerRead-class-writeFastaSR),
49