Package ‘abseqR’

March 6, 2024

Type Package

Title Reporting and data analysis functionalities for Rep-Seq datasets of antibody libraries

Version 1.20.0

Description AbSeq is a comprehensive bioinformatic pipeline for the analysis of sequencing datasets generated from antibody libraries and abseqR is one of its packages. abseqR empowers the users of abseqPy (https://github.com/malhamdoosh/abseqPy) with plotting and reporting capabilities and allows them to generate interactive HTML reports for the convenience of viewing and sharing with other researchers. Additionally, abseqR extends abseqPy to compare multiple repertoire analyses and perform further downstream analysis on its output.

License GPL-3 | file LICENSE

Encoding UTF-8

LazyData true

Depends R (>= 3.5.0)

Imports ggplot2, RColorBrewer, circlize, reshape2, VennDiagram, plyr, flexdashboard, BiocParallel (>= 1.1.25), png, grid, gridExtra, rmarkdown, knitr, vegan, ggcorrplot, gg dendro, plotly, BiocStyle, stringr, utils, methods, grDevices, stats, tools, graphics

VignetteBuilder knitr

RoxygenNote 6.1.0


SystemRequirements pandoc (>= 1.19.2.1)

URL https://github.com/malhamdoosh/abseqR

BugReports https://github.com/malhamdoosh/abseqR/issues
biocViews  Sequencing, Visualization, ReportWriting, QualityControl, MultipleComparison

Suggests  testthat

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

git_branch  RELEASE_3_18

git_last_commit  6511111

git_last_commit_date  2023-10-24

Repository  Bioconductor 3.18

Date/Publication  2024-03-06

Author  JiaHong Fong [cre, aut], Monther Alhamdoosh [aut]

Maintainer  JiaHong Fong <jiahfong@gmail.com>

R topics documented:

+AbSeqCRep,AbSeqCRep-method ................................................. 4
+AbSeqCRep,AbSeqRep-method ............................................... 5
+AbSeqRep,AbSeqCRep-method ............................................... 6
+AbSeqRep,AbSeqRep-method ............................................... 7
.abundanceAnalysis .......................................................... 8
.abundancePlot .............................................................. 8
.alignQualityHeatMaps ....................................................... 9
.allPrimerNames ............................................................. 9
.aminoAcidBar ............................................................... 10
.aminoAcidPlot .............................................................. 10
.analyzeUpstreamValidity ................................................... 11
.annotAnalysis ............................................................... 11
.asRepertoireAlignLen ....................................................... 12
.asRepertoireBitscore ....................................................... 13
.asRepertoireChain .......................................................... 13
.asRepertoireDir ............................................................. 14
.asRepertoireList ............................................................ 14
.asRepertoireName ........................................................... 15
.asRepertoirePrimer3 ......................................................... 15
.asRepertoirePrimer5 ........................................................ 16
.asRepertoireQueryStart .................................................... 16
.asRepertoireSubjectStart .................................................. 17
.asRepertoireUpstream ....................................................... 17
.boxPlot ................................................................. 18
.calculateDInd .............................................................. 18
.calculateDiversityEstimates ........................................... 19
.canonicalizeTitle ......................................................... 19
.capitalize ................................................................. 20
.checkVert ................................................................. 20
.cloneDistHist ............................................................. 21
<table>
<thead>
<tr>
<th>R topics documented:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>.cloneDistMarginal</td>
<td>21</td>
</tr>
<tr>
<td>.clonotypeAnalysis</td>
<td>22</td>
</tr>
<tr>
<td>.collateReports</td>
<td>22</td>
</tr>
<tr>
<td>.commonPrimerNames</td>
<td>23</td>
</tr>
<tr>
<td>.correlationTest</td>
<td>23</td>
</tr>
<tr>
<td>.distanceMeasure</td>
<td>24</td>
</tr>
<tr>
<td>.diversityAnalysis</td>
<td>24</td>
</tr>
<tr>
<td>.emptyPlot</td>
<td>25</td>
</tr>
<tr>
<td>.findRepertoires</td>
<td>25</td>
</tr>
<tr>
<td>.generateAllSpectratypes</td>
<td>26</td>
</tr>
<tr>
<td>.generateDelayedReport</td>
<td>26</td>
</tr>
<tr>
<td>.generateReport</td>
<td>27</td>
</tr>
<tr>
<td>.getLineTypes</td>
<td>27</td>
</tr>
<tr>
<td>.getTotal</td>
<td>28</td>
</tr>
<tr>
<td>.hmFromMatrix</td>
<td>28</td>
</tr>
<tr>
<td>.inferAnalyzed</td>
<td>29</td>
</tr>
<tr>
<td>.loadMatrixFromDF</td>
<td>29</td>
</tr>
<tr>
<td>.loadSamplesFromString</td>
<td>30</td>
</tr>
<tr>
<td>.pairwiseComparison</td>
<td>30</td>
</tr>
<tr>
<td>.plotCirclize</td>
<td>31</td>
</tr>
<tr>
<td>.plotDist</td>
<td>31</td>
</tr>
<tr>
<td>.plotDiversityCurves</td>
<td>32</td>
</tr>
<tr>
<td>.plotDuplication</td>
<td>33</td>
</tr>
<tr>
<td>.plotErrorDist</td>
<td>33</td>
</tr>
<tr>
<td>.plotIGVErrors</td>
<td>34</td>
</tr>
<tr>
<td>.plotIGVUpstreamLenDist</td>
<td>34</td>
</tr>
<tr>
<td>.plotIGVUpstreamLenDistDetailed</td>
<td>35</td>
</tr>
<tr>
<td>.plotPrimerIGVStatus</td>
<td>36</td>
</tr>
<tr>
<td>.plotPrimerIntegrity</td>
<td>37</td>
</tr>
<tr>
<td>.plotRarefaction</td>
<td>37</td>
</tr>
<tr>
<td>.plotRecapture</td>
<td>38</td>
</tr>
<tr>
<td>.plotSamples</td>
<td>39</td>
</tr>
<tr>
<td>.plotSpectratype</td>
<td>39</td>
</tr>
<tr>
<td>.plotUpstreamLength</td>
<td>40</td>
</tr>
<tr>
<td>.plotUpstreamLengthDist</td>
<td>41</td>
</tr>
<tr>
<td>.primerAnalysis</td>
<td>42</td>
</tr>
<tr>
<td>.prodDistPlot</td>
<td>42</td>
</tr>
<tr>
<td>.productivityAnalysis</td>
<td>43</td>
</tr>
<tr>
<td>.productivityPlot</td>
<td>44</td>
</tr>
<tr>
<td>.readSummary</td>
<td>44</td>
</tr>
<tr>
<td>.regionAnalysis</td>
<td>45</td>
</tr>
<tr>
<td>.reportLBE</td>
<td>45</td>
</tr>
<tr>
<td>.saveAs</td>
<td>46</td>
</tr>
<tr>
<td>.scatterPlot</td>
<td>46</td>
</tr>
<tr>
<td>.scatterPlotComplex</td>
<td>47</td>
</tr>
<tr>
<td>.secretionSignalAnalysis</td>
<td>47</td>
</tr>
<tr>
<td>.substituteStringInFile</td>
<td>48</td>
</tr>
<tr>
<td>.summarySE</td>
<td>49</td>
</tr>
</tbody>
</table>
+.AbSeqCRep,AbSeqCRep-method

Combines 2 AbSeqCRep objects together for comparison

Description

Combines 2 AbSeqCRep objects together for comparison

Usage

## S4 method for signature 'AbSeqCRep,AbSeqCRep'
e1 + e2

Arguments

e1 AbSeqCRep.
e2 AbSeqCRep.

Value

AbSeqCRep object. Calling abseqR’s functions on this object will always result in a comparison.

See Also

abseqReport returns a list of AbSeqReps

Examples

# Use example data from abseqR as abseqPy's output, substitute this
# with your own abseqPy output directory
abseqPyOutput <- tempdir()
file.copy(system.file("extdata", "ex", package = "abseqR"), abseqPyOutput, recursive=TRUE)
samples <- abseqReport(file.path(abseqPyOutput, "ex"), report = 0)

# The provided example data has PCR1, PCR2, and PCR3 samples contained within
# pcr12 and pcr13 are instances of AbSeqCRep
pcr12 <- samples["PCR1"] + samples["PCR2"]
pcr13 <- samples["PCR1"] + samples["PCR3"]
# all_S is also an instance of AbSeqCRep
all_S <- pcr12 + pcr13

# you can now call the report function on this object
# report(all_S) # uncomment this line to execute report

+AbSeqCRep,AbSeqRep-method

Combines a AbSeqCRep object with a AbSeqRep object together for comparison

Description

Combines a AbSeqCRep object with a AbSeqRep object together for comparison

Usage

## S4 method for signature 'AbSeqCRep,AbSeqRep'
e1 + e2

Arguments

e1 AbSeqCRep.
e2 AbSeqRep.

Value

AbSeqCRep object. Calling abseqR’s functions on this object will always result in a comparison.

See Also

abseqReport returns a list of AbSeqReps

Examples

# Use example data from abseqR as abseqPy's output, substitute this
# with your own abseqPy output directory
abseqPyOutput <- tempdir()
file.copy(system.file("extdata", "ex", package = "abseqR"), abseqPyOutput, recursive=TRUE)
samples <- abseqReport(file.path(abseqPyOutput, "ex"), report = 0)

# The provided example data has PCR1, PCR2, and PCR3 samples contained within
# pcr12 is an instance of AbSeqCRep
pcr12 <- samples[['PCR1']] + samples[['PCR2']]
# pcr3 is instance of AbSeqRep
pcr3 <- samples[['PCR3']] 

# pcr123 is an instance of AbSeqCRep
pcr123 <- pcr12 + pcr3
# you can now call the report function on this object
# report(pcr123)  # uncomment this line to execute report

\textbf{\texttt{\texttt{\texttt{\texttt{\texttt{\texttt{+,AbSeqRep,AbSeqCRep-method}}} }}}}

\textit{Combines a AbSeqRep object with a AbSeqCRep object together for comparison}

\section{Description}
Combines a \texttt{AbSeqRep} object with a \texttt{AbSeqCRep} object together for comparison

\section{Usage}

\begin{verbatim}
## S4 method for signature 'AbSeqRep,AbSeqCRep'
e1 + e2
\end{verbatim}

\section{Arguments}
\begin{itemize}
  \item \texttt{e1} \hspace{1cm} \texttt{AbSeqRep}.
  \item \texttt{e2} \hspace{1cm} \texttt{AbSeqCRep}.
\end{itemize}

\section{Value}
\texttt{AbSeqCRep} object. Calling abseqR’s functions on this object will always result in a comparison.

\section{See Also}
\texttt{abseqReport} returns a list of \texttt{AbSeqReps}

\section{Examples}
\begin{verbatim}
# Use example data from abseqR as abseqPy's output, substitute this # with your own abseqPy output directory
abseqPyOutput <- tempdir()
file.copy(system.file("extdata","ex",package = "abseqR"), abseqPyOutput, recursive=TRUE)
samples <- abseqReport(file.path(abseqPyOutput,"ex"), report = 0)

# The provided example data has PCR1, PCR2, and PCR3 samples contained within # pcr1 is an instance of AbSeqRep
pcr1 <- samples["PCR1"]
# pcr23 is instance of AbSeqCRep
pcr23 <- samples["PCR2"] + samples["PCR3"]

# pcr123 is an instance of AbSeqCRep
pcr123 <- pcr1 + pcr23

# you can now call the report function on this object
# report(pcr123)  # uncomment this line to execute report
\end{verbatim}
Description

Combines 2 AbSeqRep objects together for comparison.

Usage

```r
## S4 method for signature 'AbSeqRep,AbSeqRep'
e1 + e2
```

Arguments

- `e1`: AbSeqRep object.
- `e2`: AbSeqRep object.

Value

AbSeqCRep object. Calling abseqR’s functions on this object will always result in a comparison.

See Also

- `abseqReport` returns a list of AbSeqReps

Examples

```r
# Use example data from abseqR as abseqPy's output, substitute this
# with your own abseqPy output directory
abseqPyOutput <- tempdir()
file.copy(system.file("extdata", "ex", package = "abseqR"), abseqPyOutput, recursive=TRUE)
samples <- abseqReport(file.path(abseqPyOutput, "ex"), report = 0)

# The provided example data has PCR1, PCR2, and PCR3 samples contained within
# pcr1 and pcr2 are instances of AbSeqRep
pcr1 <- samples[['PCR1']]
pcri <- samples[['PCR2']]

# pcr12 is an instance of AbSeqCRep
pcr12 <- pcr1 + pcr2

# you can now call the report function on this object
# report(pcr12)  # uncomment this line to execute report
```
.abundanceAnalysis  Conducts abundance analysis

Description
Conducts abundance analysis

Usage
.abundanceAnalysis(abundanceDirectories, abunOut, sampleNames,
combinedNames, mashedNames, skipDgene = FALSE, .save = TRUE)

Arguments
abundanceDirectories
list type. List of sample directories
abunOut
string type. Output directory
sampleNames
vector type. 1-1 correspondence with abundanceDirectories
combinedNames
string type. Title "combined" sample names
mashedNames
string type. File "mashed" names - avoid special chars
skipDgene
logical type. Skip D gene plots?
.save
logical type. Save ggplot as Rdata

Value
None

.abundancePlot  Abundance distribution

Description
Abundance distribution

Usage
.abundancePlot(files, sampleNames, outputDir, skipDgene = FALSE, .save = TRUE)

Arguments
files
list type. list of files in abundance directory
sampleNames
vector type. 1-1 correspondance to files
outputDir
string type.
skipDgene
logical type. Skip D germline abundance plot if TRUE.
.save
logical type. Save Rdata ggplot item
.alignQualityHeatMaps

Value
None

Description
Plots alignment quality vs:

- mismatches
- gaps
- bitscore
- percent identity
- subject start

Usage
.alignQualityHeatMaps(abundanceDirectory, sampleName)

Arguments
abundanceDirectory
character type. fully qualified path to abundance directory

sampleName
character type. sample name

Value
list of ggplotly heatmaps

.allPrimerNames

Description
Collect primer names from FASTA

Usage
.allPrimerNames(primerFile)

Arguments
primerFile
string type. Path to primer file

Value
vector of primer names as seen in primerFile
.aminoAcidBar  
Plots amino acid composition logo

Description
Plots amino acid composition logo

Usage
.aminoAcidBar(df, scale, region, germ = "")

Arguments
- df  data frame
- scale    logical. scale to proportion?
- region   string. which region is this
- germ      string. V germline family

Value
ggplot2 object

.aminoAcidPlot  
Composition logo plot

Description
Plots 2 kinds: scaled and unscaled composition logos

Usage
.aminoAcidPlot(compositionDirectory, outdir, sampleName, 
regions = c("FR1", "CDR1", "FR2", "CDR2", "FR3", "CDR3", "FR4"),
.save = TRUE)

Arguments
- compositionDirectory  string type.
- outdir                string type.
- sampleName            string type.
- regions               logical type. vector of FR/CDR regions to plot
- .save                 logical type. save ggplot object

Value
none
Description

Plots the validity of upstream sequences

Usage

```r
.analyzeUpstreamValidity(upstreamDirectories, upstreamOut, expectedLength, upstreamLengthRange, sampleNames, combinedNames, mashedNames, .save = TRUE)
```

Arguments

- `upstreamDirectories`: list type. List of sample directories
- `upstreamOut`: string type. Output directory
- `expectedLength`: int type. Expected length of upstream sequences. (i.e. `upstream_end - upstream_start + 1`). If this is infinite, no plots will be generated.
- `upstreamLengthRange`: string type. start_end format
- `sampleNames`: vector type. 1-1 with upstream directories
- `combinedNames`: string type. Title friendly "combined" sample names
- `mashedNames`: string type. File friendly "mashed-up" sample names
- `.save`: logical type. Save Rdata?

Value

None

Description

Annotation analysis

Usage

```r
.annotAnalysis(annotDirectories, annotOut, sampleNames, mashedNames, .save = TRUE)
```
### Arguments

**annotDirectories**
- list type. List of sample directories

**annotOut**
- string type. Output directory

**sampleNames**
- vector type. 1-1 with annotDirectories

**mashedNames**
- string type. File output "mashed" sample names

**.save**
- logical type. Saves ggplot object

### Value

**none**

### Description

*Accessor for alignlen slot*

### Usage

```
.asRepertoireAlignLen(object, collapse = " - ")
```

### Arguments

**object**
- AbSeqRep object

**collapse**
- character type, collapse the range using this string.

### Value

character type. If collapse is a string, then the ranges are represented as 'start - end' in a string, if collapse is NULL, returns a character vector of length two, denoting the start and end value respectively.
.asRepertoireBitscore

Accessor for bitscore slot

Description

Accessor for bitscore slot

Usage

.asRepertoireBitscore(object, collapse = " - ")

Arguments

object
AbSeqRep object

collapse
character type, collapse the range using this string.

Value

character type. If collapse is a string, then the ranges are represented as ‘start - end’ in a string, if collapse is NULL, returns a character vector of length two, denoting the start and end value respectively.

.asRepertoireChain

Accessor for chain slot

Description

Accessor for chain slot

Usage

.asRepertoireChain(object)

Arguments

object
AbSeqRep object

Value

character type, the chain type of this sample
Description

Accessor for the outdir slot

Usage

.asRepertoireDir(object)

Arguments

object AbSeqRep object

Value

character type, the output directory of this object

Description

Accessor for AbSeqCRep’s list of AbSeqRep objects

Usage

.asRepertoireList(object)

Arguments

object AbSeqCRep object

Value

list type, list of AbSeqRep objects that together, compose this AbSeqCRep object.
### .asRepertoireName

**Accessor for the name slot**

**Description**

Accessor for the name slot

**Usage**

`asRepertoireName(object)`

**Arguments**

- `object`: AbSeqRep object

**Value**

character type, the sample name of this object.

### .asRepertoirePrimer3

**Accessor for the primer3end slot**

**Description**

Accessor for the primer3end slot

**Usage**

`asRepertoirePrimer3(object)`

**Arguments**

- `object`: AbSeqRep object

**Value**

character type, the FASTA file name for primer 3’ end sequences
.asRepertoirePrimer5  
*Accessor for the primer5end slot*

**Description**

Accessor for the primer5end slot

**Usage**

```
.asRepertoirePrimer5(object)
```

**Arguments**

- `object`  
  AbSeqRep object

**Value**

character type, the FASTA file name for primer 5' end sequences

---

.asRepertoireQueryStart

*Accessor for qstart slot*

**Description**

Accessor for qstart slot

**Usage**

```
.asRepertoireQueryStart(object, collapse = " - ")
```

**Arguments**

- `object`  
  AbSeqRep object
- `collapse`  
  character type, collapse the range using this string.

**Value**

character type. If collapse is a string, then the ranges are represented as `start - end` in a string, if collapse is NULL, returns a character vector of length two, denoting the start and end value respectively.
.asRepertoireSubjectStart

**Accessor for sstart slot**

**Description**

Accessor for sstart slot

**Usage**

.asRepertoireSubjectStart(object, collapse = " - ")

**Arguments**

- **object**: AbSeqRep object
- **collapse**: character type, collapse the range using this string.

**Value**

character type. If collapse is a string, then the ranges are represented as 'start - end' in a string, if collapse is NULL, returns a character vector of length two, denoting the start and end value respectively.

---

.asRepertoireUpstream  **Accessor for the upstream slot**

**Description**

Accessor for the upstream slot

**Usage**

.asRepertoireUpstream(object)

**Arguments**

- **object**: AbSeqRep object

**Value**

character type
.boxPlot

Creates a box plot

Description

Creates a box plot

Usage

.boxPlot(dataframes, sampleNames, plotTitle, xlabel = '', ylabel = '',
        subs = '')

Arguments

dataframes        list type. List of sample dataframes
sampleNames       vector type. 1-1 with dataframes
plotTitle         string type
xlabel            string type
ylabel            string type
subs              string type

Value

ggplot2 object

.calculatedInd

Calculates the "standard" diversity indices

Description

Calculates the "standard" diversity indices

Usage

.calculatedInd(df)

Arguments

df               clontype dataframe. Vegan format: +---------------+
                 | S.1| S.2| S.3 |
                 | S.4| ... | (each species should have its own column) +---------------+
                 | v1 | v2 | v3 | ... | (each species' count values are placed in the corresponding
column) +---------------+
.calculateDiversityEstimates

Calculates Lower Bound Estimates for unseen species and Common Diversity Indices from clonotype tables

Description

Employ common techniques to calculate LBE for unseen species and commonly used diversity indices

Usage

`.calculateDiversityEstimates(diversityDirectories, diversityOut, sampleNames)`

Arguments

- `diversityDirectories` list type. List of directories to diversity dir
- `diversityOut` string type. Output directory
- `sampleNames` vector type. 1-1 with diversityDirectories sample names

Value

None

.canonicalizeTitle

Convert file names to human friendly text

Description

Convert file names to human friendly text

Usage

`.canonicalizeTitle(str)`
Arguments
str string type

Value
string

`.capitalize`  
*Helper function to capitalize the first letter of str*

Description
Helper function to capitalize the first letter of str

Usage
`.capitalize(str)`

Arguments
str string type

Value
string, str capitalized

`.checkVert`  
*Checks if abseqPy has a metadata line that suggests the orientation*

Description
Checks if abseqPy has a metadata line that suggests the orientation

Usage
`.checkVert(filename)`

Arguments
filename csv filename

Value
True if CSV metadata says "plot vertically"
.cloneDistHist  
Marginal histogram of clonotypes (blue for shared, grey for total). The y axis is scaled by sqrt (but it doesn’t really matter anyway, since we’re stripping away the y-ticks)

Description
Marginal histogram of clonotypes (blue for shared, grey for total). The y axis is scaled by sqrt (but it doesn’t really matter anyway, since we’re stripping away the y-ticks)

Usage
```
.cloneDistHist(df.original, otherClones, lim.min, flip)
```

Arguments
- `df.original`: dataframe with all clones
- `otherClones`: clones from the other dataframe
- `lim.min`: x-axis minimum limit
- `flip`: logical type

Value
ggplot2 object

---

.cloneDistMarginal  
Marginal density graph of clonotypes (blue for shared, grey for total, purple for exclusive clones)

Description
Marginal density graph of clonotypes (blue for shared, grey for total, purple for exclusive clones)

Usage
```
.cloneDistMarginal(df.original, otherClones, lim.min, flip)
```

Arguments
- `df.original`: dataframe with all clones
- `otherClones`: clones from the other dataframe
- `lim.min`: x-axis minimum limit
- `flip`: logical type

Value
ggplot2 object
.clonotypeAnalysis  Comprehensive clonotype analyses

Description
Comprehensive clonotype analyses

Usage
.clonotypeAnalysis(diversityDirectories, clonotypeOut, sampleNames, mashedNames, .save = TRUE)

Arguments

- diversityDirectories: list type. List of directories to diversity dir
- clonotypeOut: string type. Output directory
- sampleNames: vector type. 1-1 with diversityDirectories
- mashedNames: string type. Prefix for output files using "mashed-up"
- .save: logical type. Save ggplot object?

Value
Nothing

.collateReports  Collate all HTML reports into a single directory and create an entry index.html file that redirects to all collated HTML files

Description
Collate all HTML reports into a single directory and create an entry index.html file that redirects to all collated HTML files

Usage
.collateReports(reports, individualSamples, outputDirectory)

Arguments

- reports: list/vector type. Collection of strings that are path(s) to <sample>_report.html
- individualSamples: list type. list of AbSeqRep objects. Used to extract filtering information and % read counts.
- outputDirectory: string type. Where should the report be placed.
.commonPrimerNames

Value
Nothing

Description
Collect the intersection of all primer names within a collection of primer files

Usage
.commonPrimerNames(primerFiles)

Arguments
primerFiles list / vector type. Collection of primer files

Value
vector type. Vector of primerNames that are present in ALL primerFiles. NULL if there’s no intersection at all

.correlationTest

Description
Conducts pearson and spearman correlation analysis on dataframe

Usage
.correlationTest(df)

Arguments
df dataframe with at least the following 2 columns: +--------+ | prop.x | prop.y | +--------+ | .... | .... | +--------+ where prop.x and prop.y are normalized counts (i.e. frequencies) of the clones They may contain 0 in a column to denote it being missing from sample x or y.

Value
named list of pearson, pearson.p, spearman, spearman.p
.distanceMeasure

**Description**
Computes the distance between pairwise samples

**Usage**
```
.distanceMeasure(df)
```

**Arguments**
- `df` dataframe with at least the following 2 columns: 
  - prop.x
  - prop.y

...where prop.x and prop.y are normalized counts (i.e. frequencies) of the clones. They may contain 0 in a column to denote it being missing from sample x or y.

**Value**
named list of bray.curtis, jaccard, and morisita.horn

---

.diversityAnalysis

**Title**
Diversity analysis

**Description**
Title Diversity analysis

**Usage**
```
.diversityAnalysis(diversityDirectories, diversityOut, sampleNames, 
mashedNames, .save = TRUE)
```

**Arguments**
- `diversityDirectories` list type. List of directories to diversity dir
- `diversityOut` string type. Output directory
- `sampleNames` vector type. 1-1 with diversityDirectories
- `mashedNames` string type. Prefix for output files using "mashed-up" sample names
- `.save` logical type. Save ggplot object?

**Value**
None
.emptyPlot

Description

Creates and returns an empty plot

Usage

.emptyPlot()

Value

empty ggplot2 object

.findRepertoires

Description

Given a directory = <abseqPy_outputdir>/RESULT_DIR/, returns the directories (repositories) in 'directory'. That is, will not return any sample_vs_sample directories. This is done by asserting that a 'repository' must have an (analysis.params) file, and a summary.txt file.

Usage

.findRepertoires(directory)

Arguments

directory string. Path up until <abseqPy_outputdir>/RESULT_DIR/

Value

vector of strings that are samples in 'directory', note, this is NOT a full path, but just the sample/reertoire name itself
.generateAllSpectratypes

Generates all FR/CDR spectratypes

Description
Generates all FR/CDR spectratypes

Usage
.generateAllSpectratypes(diversityDirectories, diversityOut, sampleNames, mashedNames, .save = TRUE)

Arguments

diversityDirectories
  list type. List of directories to diversity dir

diversityOut
  string type. Output directory

sampleNames
  vector type. 1-1 with diversityDirectories

mashedNames
  string type. Prefix for output files using "mashed-up" sample names

.save
  logical type. Save ggplot object?

Value
Nothing

.generateDelayedReport

Generates report for all samples in `compare`

Description
This function is needed because we are delaying the generation of reports until after all threads/processes have joined. There’s currently an issue with rmarkdown::render() in parallel execution, see: https://github.com/rstudio/rmarkdown/issues/499

Usage
.generateDelayedReport(root, compare, interactivePlot)

Arguments

root
  string, project root directory.

compare
  vector of strings, each string is a comparison defined by the user (assumes that this value has been checked).

interactivePlot
  logical, whether or not to plot interactive plotly plots.
`.generateReport`  

**Value**

a named list of samples, each an AbSeqRep object found in "root"

---

`.generateReport` *Generates HTML report from AbSeqRep and AbSeqCRep objects*

---

**Description**

Generates HTML report from AbSeqRep and AbSeqCRep objects

**Usage**

```r
.generateReport(object, root, outputDir, interactivePlot = TRUE, 
.indexHTML = "#")
```

**Arguments**

- `object` AbSeqCRep type.
- `root` string type. Root directory of the sample(s)
- `outputDir` string type. The path where the HTML will be generated
- `interactivePlot` logical type. Interactive or not
- `.indexHTML` character type. The back button will redirect to this link. This is typically used to redirect users back to index.html page

**Value**

path (including HTML name) where the report (HTML file) was saved to

---

`.getLineTypes`  

*Helper function to return line types by importance based on provided CD/Fs regions*

---

**Description**

In the aesthetics of diversity plots (rarefaction, recapture, and duplication), the line types should emphasise the most important antibody region, they're ranked in ascending order of: "FR4", "FR1", "FR2", "FR3", "CDR1", "CDR2", "CDR3", "V".

**Usage**

```
.getLineTypes(regions)
```
Arguments

regions a list/vector of strings (regions)

Value

vector of strings, each corresponding to the appropriate line type for regions.

---

**getTotal**  
*Get total number of samples (n)*

Description

Often enough, the CSV values supplied do not contain raw counts but percentages (so this value will let us know exactly the sample size).

Usage

```
.getTotal(filename)
```

Arguments

filename csv filename

Value

string, sample size.

---

**hmFromMatrix**  
*Plots a plotly heatmap from provided matrix*

Description

Plots a plotly heatmap from provided matrix

Usage

```
.hmFromMatrix(m, title, xlabel = "", ylabel = "")
```

Arguments

m matrix type  
title character type  
xlabel character type  
ylabel character type

Value

list with keys: static and interactive (ggplot2 object and plotly object respectively)
.inferAnalyzed

Returns all samples found under sampleDirectory

Description

Returns all samples found under sampleDirectory

Usage

.inferAnalyzed(sampleDirectory)

Arguments

sampleDirectory
  string, path to sample directory.

Value

un-normalized path to all samples under sampleDirectory

.loadMatrixFromDF

Given a dataframe with the columns "from", "to", and value.var, return a symmetric matrix (with diagonal values = diag). I.e. a call to isSymmetric(return_value_of_this_function) will always be TRUE.

Description

Given a dataframe with the columns "from", "to", and value.var, return a symmetric matrix (with diagonal values = diag). I.e. a call to isSymmetric(return_value_of_this_function) will always be TRUE.

Usage

.loadMatrixFromDF(dataframe, value.var, diag, unidirectional = TRUE)

Arguments

dataframe
  dataframe with 3 required columns, namely:
  +----------------------------+
  |from | to | value.var | ... |
  +----------------------------+
  where value.var is the string provided in the function parameter

value.var
  the column to use as the matrix value

diag
  what should the diagonal values be if the dataframe doesn’t provide them

unidirectional
  logical type. If the dataframe provided has the reverse pairs (i.e. a from-to pair AND a to-from pair with the same values in the value.var column), then this should be FALSE. Otherwise, this function will flip the from-to columns to generate a symmetric dataframe (and hence, a symmetric matrix).
Value

a symmetric matrix with rownames(mat) == colnames(mat) The diagonal values are filled with diag if the dataframe itself doesn’t have diagonal data

.pairwiseComparison

Conduct all vs all pairwise comparison analyses

Description

Conduct all vs all pairwise comparison analyses

Usage

.pairwiseComparison(dataframes, sampleNames, outputPath, .save = TRUE)

Arguments

dataframes list of dataframes
sampleNames 1-1 vector corresponding to dataframes
outputPath string
.save logical
Value
nothing

.plotCirclize V-J association plot

Description
V-J association plot

Usage
.plotCirclize(sampleName, path, outputdir)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sampleName</td>
<td>string type</td>
</tr>
<tr>
<td>path</td>
<td>string type. Path to _vjassoc.csv</td>
</tr>
<tr>
<td>outputdir</td>
<td>string type</td>
</tr>
</tbody>
</table>

Value
None

.plotDist Bar plotter

Description
Plots barplot for all sample in dataframes. If length(sampleNames) == 1, then the bars will also have y-values (or x if horizontal plot) labels on them. Use ‘perc’ to control if the values are percentages.

Usage
.plotDist(dataframes, sampleNames, plotTitle, vert = TRUE, xlabel = "", ylabel = "", perc = TRUE, subs = "", sorted = TRUE, cutoff = 15, legendPos = "right")
.plotDiversityCurves

Arguments

- **dataframes**: list type. List of dataframes
- **sampleNames**: vector type. 1-1 correspondence to dataframes.
- **plotTitle**: string type.
- **vert**: boolean type. True if the plot should be vertical
- **xlabel**: string type
- **ylabel**: string type
- **perc**: boolean type. True if data’s axis is a percentage proportion (instead of 0-1) only used if length(sampleNames) == 1
- **subs**: string type
- **sorted**: boolean type. True if bar plot should be sorted in descending order
- **cutoff**: int type. Number of maximum ticks to show (x on vert plots, y on hori plots).
- **legendPos**: string type. Where to position legend (see ggplot’s theme())

Value
ggplot2 object

.plotDiversityCurves *Plots rarefaction, recapture, and de-dup plots for specified region*

Description

Plots rarefaction, recapture, and de-dup plots for specified region

Usage

.plotDiversityCurves(region, diversityDirectories, sampleNames, mashedNames, diversityOut, .save = TRUE)

Arguments

- **region**: string type. One of: "cdr", "cdr_v", and "fr". "cdr" means CDR1-3, "cdr_v" means CDR3 and V only, and finally "fr" means FR1-4.
- **diversityDirectories**: list type. List of directories to diversity dir
- **sampleNames**: vector type. 1-1 with diversityDirectories
- **mashedNames**: string type. Prefix for output files using "mashed-up"
- **diversityOut**: string type. Output directory sample names
- **.save**: logical type. Save ggplot object?

Value

Nothing
.plotDuplication

**Description**

Bins singletons, doubletons, and higher order clonotypes into a line plot.

**Usage**

```
.plotDuplication(files, sampleNames, regions = c("CDR3", "V"))
```

**Arguments**

- `files` list type. List of strings to _cdr_v_duplication.csv pathname
- `sampleNames` vector type. Vector of strings each representing sample names
- `regions` vector type. Which regions to include in the plot. Default = c("CDR3", "V")

**Value**

`ggplot2` object

---

.plotErrorDist

**Description**

Plots the distribution of indels (gaps), indels in out-of-frame sequences, and the distribution of mismatches for CDRs, FRs, IGV, IGD, and IGJ.

**Usage**

```
.plotErrorDist(productivityDirectories, prodOut, sampleNames, combinedNames, mashedNames, .save = TRUE)
```

**Arguments**

- `productivityDirectories` list type. List of directories
- `prodOut` string type. Output directory
- `sampleNames` vector type. 1-1 with productivity directories
- `combinedNames` string type. Title friendly "combined" sample names
- `mashedNames` string type. File friendly "mashed-up" sample names
- `save` logical type. Save Rdata?
.plotIGVErrors

*Plots the error distribution for IGV germlines*

**Description**
Plots the distribution of in-frame unproductive, out-of-frame unproductive, and productive IGV germlines.

**Usage**
```
.plotIGVErrors(productivityDirectories, prodOut, sampleNames, combinedNames, mashedNames, .save = TRUE)
```

**Arguments**
- `productivityDirectories`: list type. List of directories
- `prodOut`: string type. Output directory
- `sampleNames`: vector type. 1-1 with productivity directories
- `combinedNames`: string type. Title friendly "combined" sample names
- `mashedNames`: string type. File friendly "mashed-up" sample names
- `save`: logical type. Save Rdata?

**Value**
None

---

.plotIGVUpstreamLenDist

*Plot IGV family distribution for a given upstreamLengthRange*

**Description**
Given an upstream length range, plot the distributions of IGV family without showing their actual lengths. If their actual lengths matter, refer to .plotIGVUpstreamLenDistDetailed.

**Usage**
```
.plotIGVUpstreamLenDist(upstreamDirectories, upstreamOut, upstreamLengthRange, lengthType, sampleNames, combinedNames, mashedNames, .save = TRUE)
```

**Value**
None
Arguments

- **upstreamDirectories**
  - list type. List of sample directories

- **upstreamOut**
  - string type. Output directory

- **upstreamLengthRange**
  - The range of upstream sequences to be included in this plot. This is usually determined by abseqPy and the format should be as follows: "min_max", e.g.: 1_15 means range(1, 15) inclusive. string type.

- **lengthType**
  - string type. "" (the empty string) denotes everything and ".short" denotes a short sequence. abseqPy dictates this because it’s used for locating the files.

- **sampleNames**
  - vector type. 1-1 with upstream directories

- **combinedNames**
  - string type. Title friendly "combined" sample names

- **mashedNames**
  - string type. File friendly "mashed-up" sample names

- **save**
  - logical type. Save Rdata?

Value

None

**.plotIGVUpstreamLenDistDetailed**

*Plots the detailed length distribution for IGV families*

Description

A boxplot for each IGV families showing the IQR of upstream lengths. In contrast to **.plotIGVUpstreamLenDist**, which only shows the distribution of IGV families over upstreamLengthRange.

Usage

```r
.plotIGVUpstreamLenDistDetailed(upstreamDirectories, upstreamOut, upstreamLengthRange, lengthType, sampleNames, combinedNames, mashedNames, .save = TRUE)
```

Arguments

- **upstreamDirectories**
  - list type. List of sample directories

- **upstreamOut**
  - string type. Output directory

- **upstreamLengthRange**
  - The range of upstream sequences to be included in this plot. This is usually determined by abseqPy and the format should be as follows: "min_max", e.g.: 1_15 means range(1, 15) inclusive. string type.
.plotPrimerIGVStatus

lengthType  string type. "" (the empty string) denotes everything and "_short" denotes a short sequence. abseqPy dictates this because it's used for locating the files.
sampleNames  vector type. 1-1 with upstream directories
combinedNames  string type. Title friendly "combined" sample names
mashedNames  string type. File friendly "mashed-up" sample names
.save  logical type. Save Rdata?

Value
None

.plotPrimerIGVStatus  Plots, for a given category and pend, the primer IGV indelled distribution in a bar plot

Description
Plots the abundance of indelled primers relative to IGV germlines

Usage
.plotPrimerIGVStatus(primer, pend, category, primerDirectories, sampleNames, primerOut, combinedNames, mashedNames, .save = TRUE)

Arguments
primer  string, primer name
pend  string, either 3 or 5 (primer end)
category  string, either "all", "productive", or "outframe"
primerDirectories  string type. Path to primer analysis directory
sampleNames  vector type. 1-1 with primerDirectories
primerOut  string type. output directory
combinedNames  string type. Title friendly "combined" sample names
mashedNames  string type. File friendly "mashed-up" sample names
.save  logical type. Save Rdata?

Value
None
.plotPrimerIntegrity

Plots the distribution of primer integrity for a given category and 5' or 3' pend

Description

Plots the distribution of primer integrity for a given category and 5' or 3' pend

Usage

.plotPrimerIntegrity(primerIntegrity, pend, category, primerDirectories, sampleNames, primerOut, combinedNames, mashedNames, .save = TRUE)

Arguments

primerIntegrity  string. One of "stopcodon", "integrity", "indelled", "indel_pos"
pend              string, either 3 or 5 (primer end)
category          string, either "all", "productive", or "outframe"
primerDirectories string type. Path to primer analysis directory
sampleNames       vector type. 1-1 with primerDirectories
primerOut         string type. output directory
combinedNames     string type. Title friendly "combined" sample names
mashedNames       string type. File friendly "mashed-up" sample names
.save             logical type. Save Rdata?

Value

None

.plotRarefaction

Rarefaction plot

Description

Plots the number of unique clonotypes (on the y-axis) drawn from a sample size on the x axis. The number of unique clonotypes is averaged over 5 repeated rounds.

Usage

.plotRarefaction(files, sampleNames, regions = c("CDR3", "V"))
**Arguments**

files  
list type. A list of files consisting of path to samples

sampleNames  
vector type. A vector of strings, each being the name of samples in files

regions  
vector type. A vector of strings, regions to be included. Defaults to c("CDR3", "V")

**Value**

ggplot2 object

---

**Description**

Plots the percent of recapture clonotypes (on the y-axis) drawn from a repeated (with replacement) sample size on the x axis. The percentage of recaptured clonotypes is averaged over 5 recapture rounds.

**Usage**

`.plotRecapture(files, sampleNames, regions = c("CDR3", "V"))`

**Arguments**

files  
list type. List of _cdr_v_recapture.csv.gz files.

sampleNames  
vector type. A vector of strings each representing the name of samples in files.

regions  
vector type. A vector of strings, regions to be included in the plot. defaults to c("CDR3", "V")

**Value**

ggplot2 object
.plotSamples

Monolith AbSeq Plot function - the "driver" program

Description

Monolith AbSeq Plot function - the "driver" program

Usage

.plotSamples(sampleNames, directories, analysis, outputDir, primer5Files, primer3Files, upstreamRanges, skipDgene = FALSE)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sampleNames</td>
<td>vector type. Vector of sample names in strings</td>
</tr>
<tr>
<td>directories</td>
<td>vector type. Vector of directories in strings, must be 1-1 with sampleNames</td>
</tr>
<tr>
<td>analysis</td>
<td>vector / list type. What analysis to plot for. If sampleNames or directories is &gt; 1 (i.e. AbSeqCRep), then make sure that it’s an intersection of all analysis conducted by the repertoires, otherwise, it wouldn’t make sense</td>
</tr>
<tr>
<td>outputDir</td>
<td>string type. Where to dump the output</td>
</tr>
<tr>
<td>primer5Files</td>
<td>vector / list type. Collection of strings that the sample used for primer5 analysis. If sample N doesn’t have a primer 5 file, leave it as anything but a valid file path.</td>
</tr>
<tr>
<td>primer3Files</td>
<td>vector / list type. Collection of strings that the sample used for primer 3 analysis. If sample N doesn’t have a primer 3 file, leave it as anything but a valid file path.</td>
</tr>
<tr>
<td>upstreamRanges</td>
<td>list type. Collection of &quot;None&quot;s or vector denoting upstreamStart and upstreamEnd for each sample.</td>
</tr>
<tr>
<td>skipDgene</td>
<td>logical type. Whether or not to skip D gene distribution plot</td>
</tr>
</tbody>
</table>

Value

none

.plotSpectratype

Spectratype plotter

Description

Plots length distribution

Usage

.plotSpectratype(dataframes, sampleNames, region, title = "Spectratype", subtitle = "", xlabel = "Length(AA)", ylabel = "Distribution", showLabel = FALSE)
Arguments

- **dataframes**: list type. List of dataframes.
- **sampleNames**: vector type. 1-1 correspondence with dataframes.
- **region**: string type. Region that will be displayed in the plot title. This specifies which region this spectratype belongs to. If not supplied, a default (start, end) range will be displayed instead.
- **title**: string type. Ignored if region is specified.
- **subtitle**: string type.
- **xlabel**: string type.
- **ylabel**: string type.
- **showLabel**: bool type. Show geom_text? - Ignored if samples > 1

Value

ggplot2 object

Description

Plot upstream distribution

Usage

```r
.plotUpstreamLength(upstreamDirectories, upstreamOut, expectedLength, upstreamLengthRange, sampleNames, combinedNames, mashedNames, .save = TRUE)
```

Arguments

- **upstreamDirectories**: list type. List of sample directories.
- **upstreamOut**: string type. Output directory.
- **expectedLength**: int type. Expected length of upstream sequences. (i.e. upstream_end - upstream_start + 1).
- **upstreamLengthRange**: string type. start_end format.
- **sampleNames**: vector type. 1-1 with upstream directories.
- **combinedNames**: string type. Title friendly "combined" sample names.
- **mashedNames**: string type. File friendly "mashed-up" sample names.
- **.save**: logical type. Save Rdata?

Value

None
.plotUpstreamLengthDist

Plot upstream sequence length distribution for upstream sequences (5'UTR or secretion signal) for a given upstreamLengthRange

Description

Given an upstream length range, plot the distribution of upstream sequence lengths.

Usage

.plotUpstreamLengthDist(upstreamDirectories, upstreamOut, upstreamLengthRange, lengthType, sampleNames, combinedNames, mashedNames, .save)

Arguments

upstreamDirectories
   list type. List of sample directories

upstreamOut
   string type. Output directory

upstreamLengthRange
   The range of upstream sequences to be included in this plot. This is usually determined by abseqPy and the format should be as follows: "min_max", e.g.: 1_15 means range(1, 15) inclusive. string type.

lengthType
   string type. "" (the empty string) denotes everything and "_short" denotes a short sequence. abseqPy dictates this because it's used for locating the files.

sampleNames
   vector type. 1-1 with upstream directories

combinedNames
   string type. Title friendly "combined" sample names

mashedNames
   string type. File friendly "mashed-up" sample names

.save
   logical type. Save Rdata?

Value

None
.primerAnalysis  

Conducts primer specificity analysis

Description

Conducts primer specificity analysis

Usage

.primerAnalysis(primerDirectories, primer5Files, primer3Files, primerOut, sampleNames, combinedNames, mashedNames, .save = TRUE)

Arguments

- **primerDirectories**: string type. Path to primer analysis directory
- **primer5Files**: vector/list type. 5' end primer files
- **primer3Files**: vector/list type. 3' end primer files
- **primerOut**: string type. output directory
- **sampleNames**: vector type. 1-1 with primerDirectories
- **combinedNames**: string type. Title friendly "combined" sample names
- **mashedNames**: string type. File friendly "mashed-up" sample names
- **.save**: logical type. Save Rdata?

Value

None

.prodDistPlot  

Plots a distribution plot for different productivity analysis files

Description

A wrapper for plotDist

Usage

.prodDistPlot(productivityDirectories, sampleNames, title, reg, outputFileName, region, .save = TRUE)
Arguments

productivityDirectories  
vector type. directories where all productivity csv files lives (usually <sample-name>/productivity/)
sampleNames  
vector type.
title  
string type.
reg  
string type. Regular expression to find the right files for this particular distribution plot
outputFileName  
string type. Vector of file names to save in the order of regions
region  
string type. Most of the dist plots are regional based. use "" if no regions are involved
.save  
logical type. Save Rdata?

Value

None

.productivityAnalysis  
Conducts productivity analysis

Description

Conducts productivity analysis

Usage

.productivityAnalysis(productivityDirectories, prodOut, sampleNames, combinedNames, mashedNames, .save = TRUE)

Arguments

productivityDirectories  
list type. List of directories
prodOut  
string type. Output directory
sampleNames  
vector type. 1-1 with productivity directories
combinedNames  
string type. Title friendly "combined" sample names
mashedNames  
string type. File friendly "mashed-up" sample names
.save  
logical type. Save Rdata

Value

None
.productivityPlot

Description

Shows the percentage of 1. productivity, 2. non-functional + reason for being unproductive, i.e. "Stop Codon" or "Out of frame" or "Stop & Out"

Usage

.productivityPlot(dataframes, sampleNames)

Arguments

dataframes list type. List of sample dataframes
sampleNames vector type. 1-1 with dataframes

Value

ggplot2 object

.readSummary

Description

Return value specified by key from AbSeq’s summary file

Usage

.readSummary(sampleRoot, key)

Arguments

sampleRoot sample’s root directory. For example, /path/to/<outputdir>/reports/<sample_name>.
key character type. Possible values are (though they might change)
  • RawReads
  • AnnotatedReads
  • FilteredReads
  • ProductiveReads

Value

value associated with key from summary file. "NA" (in string) if the field is not available refer to util.R for the key values
.regionAnalysis

Title Shows varying regions for a given clonotype defined by its CDR3

Description
Title Shows varying regions for a given clonotype defined by its CDR3

Usage
.regionAnalysis(path, sampleName, top = 15)

Arguments

path  
string type. Path to diversity folder where <sampleName>_clonotype_diversity_region_analysis.csv.gz is located

sampleName  
string type

top  
int type. Top N number of clones to analyze

Value

ggplot2 object

.reportLBE

Reports abundance-based (Lower bound) diversity estimates using the Vegan package

Description
Reports abundance-based (Lower bound) diversity estimates using the Vegan package

Usage
.reportLBE(df)

Arguments
df  
clonotype dataframe. Vegan format: +—————————+ | S.1| S.2| S.3 | S.4 | ... | (each species should have its own column) +—————————+ | v1 | v2 | v3 | ... | (each species’ count values are placed in the corresponding column) +——+ | v1 | v2 | .... | +

Value
dataframe with the format: +—————————————————-+ | S.obs | S.chao1 | se.chao1 | S.ACE | se.ACE | s.jack1 | s.jack2 | +—————————————————-+ | v1 | v2 | ... | +
.saveAs  
_Saves ggplot object as a Rdata file._

**Description**

It’s a convinient function that does the check and saves at the same time, for brevity within other areas of the code (to eliminate repeated if checks).

**Usage**

```
.saveAs(.save, filename, plot)
```

**Arguments**

- `.save` logical type. Whether or not we should save.
- `filename` string.
- `plot` ggplot object.

**Value**

nothing

---

.scatterPlot  
_Title Creates a scatter plot_

**Description**

Title Creates a scatter plot

**Usage**

```
.scatterPlot(df1, df2, name1, name2, cloneClass)
```

**Arguments**

- `df1` dataframe for sample 1
- `df2` dataframe for sample 2
- `name1` string type. Sample 1 name
- `name2` string type. Sample 2 name
- `cloneClass` string type. What region was used to classify clonotypes - appears in title. For example, CDR3 or V region

**Value**

ggplot2 object
.scatterPlotComplex

Description

Creates a complex scatter plot

Usage

.scatterPlotComplex(df.union, df1, df2, name1, name2, cloneClass)

Arguments

df.union: a 'lossless' dataframe created by intersecting sample1 and sample2's dataframes. It should contain NAs where clones that appear in one sample doesn’t appear in the other. For example:

+----------------------------------+
| Clonotype | prop.x | prop.y | Count.x |
| Count.y | +----------------------------------+
| ABCDEF | NA | 0.01 | NA |
| 210 | ...... | +|

df1: dataframe for sample 1
df2: dataframe for sample 2
name1: string type, Sample 1 name
name2: string type, Sample 2 name
cloneClass: string type. What region was used to classify clonotypes - appears in title. For example, CDR3 or V region

Value

ggplot2 object

.secretionSignalAnalysis

Description

Secretion signal analysis

Usage

.secretionSignalAnalysis(secDirectories, secOut, sampleNames, combinedNames, mashedNames, upstreamRanges, .save = TRUE)
Arguments

- **secDirectories** list type. Secretion signal directories where files are located
- **secOut** string type. Where to dump output
- **sampleNames** vector type. 1-1 with secDirectories
- **combinedNames** string type. Title friendly string
- **mashedNames** string type. File name friendly string
- **upstreamRanges** list type. Upstream ranges for each sample. If length(secDirectories) > 1, the plots will only be generated for upstream ranges that are present in ALL samples. (i.e. the intersection)
- **.save** logical type, save Rdata?

Value

none

---

**.substituteStringInFile**  
*Substitutes the first occurrence of 'key' with 'value' in 'filename'*

Description

Substitutes the first occurrence of 'key' with 'value' in 'filename'

Usage

```
.substituteStringInFile(filename, key, value, fixed = FALSE)
```

Arguments

- **filename** character type
- **key** character type
- **value** character type
- **fixed** logical type

Value

None
**.summarySE**  
*Summary of dataframe*

**Description**

Gives count, mean, standard deviation, standard error of the mean, and confidence interval (default 95%).

adapted from http://www.cookbook-r.com/Graphs/Plotting_means_and_error_bars_(ggplot2)/#Helper functions

**Usage**

`.summarySE(data = NULL, measurevar, groupvars = NULL, na.rm = FALSE, conf.interval = 0.95, .drop = TRUE)`

**Arguments**

- `data` : a data frame.  
- `measurevar` : the name of a column that contains the variable to be summarized  
- `groupvars` : a vector containing names of columns that contain grouping variables  
- `na.rm` : a boolean that indicates whether to ignore NA's  
- `conf.interval` : the percent range of the confidence interval (default is 95%)  
- `.drop` : logical.

**Value**

dataframe

---

**.topNDist**  
*Title Clonotype table*

**Description**

Title Clonotype table

**Usage**

`.topNDist(dataframes, sampleNames, top = 10)`

**Arguments**

- `dataframes` : list type. List of dataframes.  
- `sampleNames` : vector type. vector of strings representing sample names should have one-to-one correspondence with dataframes  
- `top` : int type. Top N clonotypes to plot
**.UTR5Analysis**

**Description**

Generates all the required plots for 5' UTR analysis. This includes upstream length distributions and upstream sequence validity.

**Usage**

```r
.UTR5Analysis(utr5Directories, utr5Out, sampleNames, combinedNames, 
mashedNames, upstreamRanges, .save = TRUE)
```

**Arguments**

- `utr5Directories`: list type. 5'UTR directories where files are located
- `utr5Out`: string type. Where to dump output
- `sampleNames`: vector type. 1-1 with utr5Directories
- `combinedNames`: string type. Title friendly string
- `mashedNames`: string type. File name friendly string
- `upstreamRanges`: list type. Upstream ranges for each sample. If length(utr5Directories) > 1, the plots will only be generated for upstream ranges that are present in ALL samples. (i.e the intersection)
- `.save`: logical type, save Rdata?

**Value**

None
.vennIntersection

Title Creates Venn diagram for clonotype intersection

Description
Title Creates Venn diagram for clonotype intersection

Usage
.vennIntersection(dataframes, sampleNames, outFile, top = Inf)

Arguments

dataframes list type. List of sample dataframes. Only accepts 2 - 5 samples. Warning message will be generated for anything outside of the range

sampleNames vector type. 1-1 with dataframes

outFile string type. Filename to be saved as

top int type. Top N cutoff, defaults to ALL clones if not specified

Value
Nothing

AbSeqCRep-class
S4 class - AbSeqCompositeRepertoire analysis object

Description
AbSeqCRep is a collection of AbSeqRep S4 objects. This S4 class contains multiple samples(repertoires) and it can be "combined" with other samples by using the + operator to create an extended AbSeqCRep object. This value, in turn, can be used as the first argument to report which generates a comparison between all samples included in the + operation.

Users do not manually construct this class, but rather indirectly obtain this class object as a return value from the + operation between two AbSeqRep objects, which are in turn, obtained indirectly from abseqReport and report functions. It is also possible to obtain this class object by + (adding) AbSeqCRep objects.

Value
AbSeqCRep

Slots

repertoires list of AbSeqRep objects.
See Also

AbSeqRep

Examples

```r
# Use example data from abseqR as abseqPy's output, substitute this
# with your own abseqPy output directory
abseqPyOutput <- tempdir()
file.copy(system.file("extdata", "ex", package = "abseqR"), abseqPyOutput, recursive=TRUE)
samples <- abseqReport(file.path(abseqPyOutput, "ex"), report = 0)

# The provided example data has PCR1, PCR2, and PCR3 samples contained within
# pcr12 and pcr13 are instances of AbSeqCRep
pcr12 <- samples["PCR1"] + samples["PCR2"]
pcr13 <- samples["PCR1"] + samples["PCR3"]

# all_S is also an instance of AbSeqCRep
all_S <- pcr12 + pcr13
```

---

### AbSeqRep-class

*S4 class - AbSeqRepertoire analysis object*

**Description**

The AbSeqRep object contains all metadata associated with the AbSeq (python backend) run conducted on it. This S4 class represents a single sample(repertoire) and it can be "combined" with other samples by using the + operator to create an AbSeqCRep object. This value, in turn, can be used as the first argument to `report` which generates a comparison between all samples included in the + operation.

Users do not manually construct this class, but rather indirectly obtain this class object as a return value from the `abseqReport` and `report` functions.

**Value**

AbSeqRep

**Slots**

- `f1` character. Path to FASTA/FASTQ file 1.
- `f2` character. Path to FASTA/FASTQ file 2.
- `chain` character. Type of chain, possible values:
  - `hv`
  - `lv`
  - `kv`
  - `klv`

  each representing Heavy, Lambda and Kappa respectively.
task character. Type of analysis conducted, possible values:
  • all
  • annotate
  • abundance
  • diversity
  • productivity
  • fastqc
  • primer
  • 5utr
  • rsasimple
  • seqlen
  • secretion
  • seqlenclass

name character. Name of analysis.

bitscore numeric. Part of filtering criteria: V gene bitscore filter value.

qstart numeric. Part of filtering criteria: V gene query start filter value.

sstart numeric. Part of filtering criteria: V gene subject start filter value.

alignlen numeric. Part of filtering criteria: V gene alignment length filter value.

clonelimit numeric. Number of clones to export into csv file. This is only relevant in -t all or -t diversity where clonotypes are exported into <outdir>/<name>/diversity/clonotypes

detailedComposition logical. Plots composition logo by IGHV families if set to true, otherwise, plots logos by FR/CDRs.

log character. Path to log file.

merger character. Merger used to merge paired-end reads.

fmt character. File format of file1 and (if present) file2. Possible values are FASTA or FASTQ.

sites character. Path to restriction sites txt file. This option is only used if -t rsasimple

primer5end ANY. Path to 5' end primer FASTA file.

primer3end ANY. Path to 3' end primer FASTA file.

trim5 numeric. Number of nucleotides to trim at the 5' end;

trim3 numeric. Number of nucleotides to trim at the 3' end;

outdir character. Path to output directory

primer5endoffset numeric. Number of nucleotides to offset before aligning 5' end primers in primer5end FASTA file.

threads numeric. Number of threads to run.

upstream character. Index (range) of upstream nucleotides to analyze. This option is only used if -t 5utr or -t secretion.

seqtype character. Sequence type, possible values are either dna or protein.

database character. Path to IgBLAST database.

actualqstart numeric. Query sequence’s starting index (indexing starts from 1). This value overrides the inferred query start position by AbSeq.
fr4cut logical. The end of FR4 is marked as the end of the sequence if set to TRUE, otherwise the end of the sequence is either the end of the read itself, or trimmed to --trim3 <num>.

domainSystem character. Domain system to use in IgBLAST, possible values are either imgt or kabat.

See Also

abseqReport returns a list of AbSeqRep objects.

Examples

# this class is not directly constructed by users, but as a return value from the abseqReport method.
# Use example data from abseqR as abseqPy's output, substitute this with your own abseqPy output directory
abseqPyOutput <- tempdir()
file.copy(system.file("extdata", "ex", package = "abseqR"), abseqPyOutput, recursive=TRUE)
samples <- abseqReport(file.path(abseqPyOutput, "ex"), report = 0)

abseqReport Visualize all analysis conducted by abseqPy

Description

Plots all samples in the output directory supplied to abseqPy's --outdir or -o argument. Users can optionally specify which samples in directory should be compared. Doing so generates additional plots for clonotype comparison and the usual plots will also conveniently include these samples using additional aesthetics.

Calling this function with a valid directory will always return a named list of objects; these individual objects can be combined using the + operator to form a new comparison, in which the report function accepts as its first parameter.

Usage

abseqReport(directory, report, compare, BPPARAM)

Arguments

directory string type. directory as specified in -o or --outdir in abseqPy. This tells AbSeq where to look for abseqPy's output.

report (optional) integer type. The possible values are:
  • 0 - does nothing (returns named list of AbSeqRep objects)
  • 1 - generates plots for csv files
  • 2 - generates a report that collates all plots
  • 3 - generates interactive plots in report (default)
each higher value also does what the previous values do. For example, report = 2 will return a named list of `AbSeqRep` objects, plot csv files, and generate a (non-interactive)HTML report that collates all the plots together.

**compare**

(optional) vector of strings. From the samples in found in directory directory, they can be selected and compared against each other. For example, to compare "sample1" with "sample2" and "sample3" with "sample4", compare should be c("sample1",sample2", "sample3",sample4"). An error will be thrown if the samples specified in this parameter are not found in directory.

**BPPARAM**

(optional) BiocParallel backend. Configures the parallel implementation. Refer to `BiocParallel` for more information. By default, use all available cores.

**Value**

named list. List of `AbSeqRep` objects. The names of the list elements are taken directly from the repertoire object itself. This return value is consistent with the return value of `report`.

**See Also**

`AbSeqRep`  
`report`. Analogous function, but takes input from an `AbSeqRep` or `AbSeqCRep` object instead.

**Examples**

```r
# Use example data from abseqR as abseqPy's output, substitute this
# with your own abseqPy output directory
abseqPyOutput <- tempdir()
file.copy(system.file("extdata", "ex", package = "abseqR"), abseqPyOutput, recursive=TRUE)

### 1. The `report` parameter usage example:

# report = 0; don't plot, don't collate a HTML report, don't show anything interactive
samples <- abseqReport(file.path(abseqPyOutput, "ex"), report = 0)
# samples is now a named list of AbSeqRep objects

# report = 1; just plot pngs; don't collate a HTML report; nothing interactive
# samples <- abseqReport(file.path(abseqPyOutput, "ex"), report = 1)
# samples is now a named list of AbSeqRep objects

# report = 2; plot pngs; collate a HTML report; HTML report will NOT be interactive
# samples <- abseqReport(file.path(abseqPyOutput, "ex"), report = 2)
# samples is now a named list of AbSeqRep objects

# report = 3 (default); plot pngs; collate a HTML report; HTML report will be interactive
# samples <- abseqReport(file.path(abseqPyOutput, "ex"), report = 3)
# samples is now a named list of AbSeqRep objects

### 2. Using the return value of abseqReport:

# NOTE, often, this is used to load multiple samples from different directories
# using abseqReport (with report = 0), then the samples are added together
# before calling the report function. This is most useful when the samples
```
# live in different abseqPy output directory.

# Note that the provided example data has PCR1, PCR2, and PCR3 samples contained within the directory
stopifnot(names(samples) == c("PCR1", "PCR2", "PCR3"))

# as a hypothetical example, say we found something interesting in PCR1 and PCR3, and we want to isolate them:
# we want to explicitly compare PCR1 with PCR3
pcr13 <- samples[["PCR1"]]+samples[["PCR3"]]

# see abseqR::report for more information.
# abseqR::report(pcr13) # uncomment this line to run

### BPPARAM usage:

# 4 core machine, use all cores - use whatever value that suits you
nproc <- 4
# samples <- abseqReport(file.path(abseqPyOutput, "ex"),
# BPPARAM = BiocParallel::MulticoreParam(nproc))

# run sequentially - no multiprocessing
# samples <- abseqReport(file.path(abseqPyOutput, "ex"),
# BPPARAM = BiocParallel::SerialParam())

# see https://bioconductor.org/packages/release/bioc/html/BiocParallel.html
# for more information about how to use BPPARAM and BiocParallel in general.

---

**report**

Plots *AbSeqRep* or *AbSeqCRep* object to the specified directory

**Description**

Plots all samples in the object argument and saves the analysis in outputDir. Users can optionally specify which samples in object should be compared. Doing so generates additional plots for clonotype comparison and the usual plots will also conveniently include these samples using additional aesthetics.

This method is analogous to `abseqReport`. The only difference is that this method accepts *AbSeqRep* or *AbSeqCRep* objects as its first parameter, and the outputDir specifies where to store the result.

**Usage**

```r
report(object, outputDir, report = 3)
```

## S4 method for signature 'AbSeqRep'
report(object, outputDir, report = 3)
## S4 method for signature 'AbSeqCRep'

`report(object, outputDir, report = 3)`

### Arguments

- **object**: AbSeqRep or AbSeqCRep object to plot.
- **outputDir**: string type. Directory where analysis will be saved to.
- **report**: (optional) integer type. The possible values are:
  - 0 - does nothing (returns named list of `AbSeqRep` objects)
  - 1 - generates plots for csv files
  - 2 - generates a report that collates all plots
  - 3 - generates interactive plots in report (default)

Each value also does what the previous values do. For example, `report = 2` will return a named list of `AbSeqRep` objects, plot csv files, and generate a (non-interactive) HTML report that collates all the plots together.

### Value

named list. List of `AbSeqRep` objects. The names of the list elements are taken directly from the repertoire object itself. This return value is consistent with the return value of `abseqReport`.

### See Also

- `abseqReport`. Analogous function, but takes input from a string that signifies the output directory of abseqPy as the first argument instead.
- `AbSeqRep`
- `AbSeqCRep`

### Examples

```r
# Use example data from abseqR as abseqPy's output, substitute this
# with your own abseqPy output directory
abseqPyOutput <- tempdir()
file.copy(system.file("extdata", "ex", package = "abseqR"), abseqPyOutput, recursive=TRUE)
samples <- abseqReport(file.path(abseqPyOutput, "ex"), report = 0)

# The provided example data has PCR1, PCR2, and PCR3 samples contained within
# We can use the + operator to combine samples, thus requesting the
# report function to compare them:
pcr12 <- samples[["PCR1"]] + samples[["PCR2"]]

# generate plots and report for this new comparison
# report(pcr12, "PCR1_vs_PCR2")

# generate plots only
# report(pcr12, "PCR1_vs_PCR2", report = 1)

# generate plots, and a non-interactive report
```
# report(pcr12, "PCR1_vs_PCR2", report = 2)

# generate plots, and an interactive report
# report(pcr12, "PCR1_vs_PCR2", report = 3)  # this is the default
.summarySE, 49
.topNDist, 49
.vennIntersection, 51

AbSeqCRep, 4–7, 14, 27, 51, 52, 55–57
AbSeqCRep (AbSeqCRep-class), 51
AbSeqCRep-class, 51
AbSeqRep, 5–7, 14, 27, 51, 52, 54–57
AbSeqRep (AbSeqRep-class), 52
AbSeqRep-class, 52
abseqReport, 4–7, 51, 52, 54, 55, 56
report, AbSeqCRep-method (report), 56
report, AbSeqRep-method (report), 56