Package 'scRepertoire'

April 11, 2023

Title A toolkit for single-cell immune receptor profiling

Version 1.8.0

Description
scRepertoire was built to process data derived from the 10x Genomics Chromium Immune Profiling for both T-cell receptor (TCR) and immunoglobulin (Ig) enrichment workflows and subsequently interacts with the popular Seurat and SingleCellExperiment R packages. It also allows for general analysis of single-cell clonotype information without the use of expression information. The package functions as a wrapper for Startrac and powerTCR R packages.

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 7.2.1

biocViews Software, ImmunoOncology, SingleCell, Classification, Annotation, Sequencing

Depends ggplot2, R (>= 4.0)

Imports stringdist, dplyr, reshape2, ggalluvial, stringr, vegan, powerTCR, SingleCellExperiment, SummarizedExperiment, plyr, parallel, doParallel, methods, utils, rlang, igraph, ggraph, tidygraph, SeuratObject

Suggests knitr, rmarkdown, BiocStyle, circlize, scales, Seurat, scater

VignetteBuilder knitr

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

git_branch RELEASE_3_16

git_last_commit a07f550

git_last_commit_date 2022-11-01

Date/Publication 2023-04-10

Author Nick Borcherding [aut, cre]

Maintainer Nick Borcherding <ncborch@gmail.com>

R topics documented:

`abundanceContig` *Demonstrate the relative abundance of clonotypes by group or sample.*

Description

This function takes the output of combineTCR(), combineBCR(), or expression2List() and displays the number of clonotypes at specific frequencies by sample or group. Visualization can either be a line graph using calculated numbers or if scale = TRUE, the output will be a density plot. Multiple sequencing runs can be group together using the group parameter. If a matrix output for the data is preferred, set exportTable = TRUE.

Usage

```
abundanceContig(
  df,
  cloneCall = "strict",
  chain = "both",
  scale = FALSE,
  group.by = NULL,
  split.by = NULL,
  order = TRUE,
  exportTable = FALSE
)
```

Arguments

df	The product of combineTCR(), combineBCR(), expression2List(), or combine-Expression().
cloneCall	How to call the clonotype - VDJC gene (gene), CDR3 nucleotide (nt), CDR3 amino acid (aa), or VDJC gene + CDR3 nucleotide (strict).
chain	indicate if both or a specific chain should be used - e.g. "both", "TRA", "TRG", "IGH", "IGL"
scale	Converts the graphs into density plots in order to show relative distributions.
group.by	The column header for which you would like to analyze the data.
split.by	If using a single-cell object, the column header to group the new list. NULL will return clusters.
order	Maintain the order of the list when plotting
exportTable	Returns the data frame used for forming the graph to the visualization.

Value

ggplot of the total or relative abundance of clonotypes across quanta

Examples

```
#Making combined contig data
x <- contig_list
combined <- combineTCR(x, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")
abundanceContig(combined, cloneCall = "gene", scale = FALSE)
```

addVariable *Adding variables after the combination of contigs.*

Description

This function adds variables to the product of combineTCR() combineBCR() or expression2List() to be used in later visualizations. For each element, the function will add a column (labled by name) with the variable. The length of the variable paramater needs to match the length of the combined object.

Usage

```
addVariable(df, name = NULL, variables = NULL)
```

Arguments

df	The product of combineTCR() combineBCR() or expression2List().
name	The column header to add.
variables	The exact values to add to each element of the list.

Value

list of contigs with a new column (name).

Examples

```
x <- contig_list
combined <- combineTCR(x, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")
combined <- addVariable(combined, name = "batch", variables = c(1,1,1,1,2,2))
```

alluvialClonotypes *Exploring interaction of clonotypes by seurat or SCE dynamics*

Description

View the proportional contribution of clonotypes by seurat or SCE object meta data after combineExpression(). The visualization is based on the ggalluvial package, which requires the aesthetics to be part of the axes that are visualized. Therefore, alpha, facet, and color should be part of the the axes you wish to view or will add an additional stratum/column to the end of the graph.

Usage

```
alluvialClonotypes(
  sc,
  cloneCall = c("gene", "nt", "aa", "strict"),
  chain = "both",
  y.axes = NULL,
  color = NULL,
  alpha = NULL,
  facet = NULL
)
```

Arguments

sc	The seurat or SCE object to visualize after combineExpression(). For SCE objects, the cluster variable must be in the meta data under "cluster".
cloneCall	How to call the clonotype - VDJC gene (gene), CDR3 nucleotide (nt) or CDR3 amino acid (aa), or VDJC gene + CDR3 nucleotide (strict).
chain	indicate if both or a specific chain should be used - e.g. "both", "TRA", "TRG", "IGH", "IGL"
y.axes	The columns that will separate the proportional visualizations.
color	The column header or clonotype(s) to be highlighted.
alpha	The column header to have gradated opacity.
facet	The column label to separate.

Value

Alluvial ggplot comparing clonotype distribution across selected parameters.

Examples

```
#Getting the combined contigs
combined <- combineTCR(contig_list, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")

#Getting a sample of a Seurat object
screp_example <- get(data("screp_example"))
sce <- suppressMessages(Seurat::UpdateSeuratObject(screp_example))
sce <- Seurat::as.SingleCellExperiment(sce)

#Using combineExpresion()
sce <- combineExpression(combined, sce)

#Using alluvialClonotypes()
alluvialClonotypes(sce, cloneCall = "gene",
y.axes = c("Patient", "ident"), color = "ident")
```

`clonalDiversity` *Examine the clonal diversity of samples*

Description

This function calculates traditional measures of diversity - Shannon, inverse Simpson, Chao1 index, abundance-based coverage estimators (ACE), and 1-Pielou's measure of species evenness by sample or group. The function automatically down samples the diversity metrics using 100 boot straps The group parameter can be used to condense the individual samples. If a matrix output for the data is preferred, set exportTable = TRUE.

Usage

```
clonalDiversity(
  df,
  cloneCall = "strict",
  chain = "both",
  group.by = NULL,
  x.axis = NULL,
  split.by = NULL,
  exportTable = FALSE,
  n.boots = 100,
  return.boots = FALSE
)
```

Arguments

`df`	The product of combineTCR(), combineBCR(), expression2List(), or combineExpression().
`cloneCall`	How to call the clonotype - VDJC gene (gene), CDR3 nucleotide (nt), CDR3 amino acid (aa), or VDJC gene + CDR3 nucleotide (strict).
`chain`	indicate if both or a specific chain should be used - e.g. "both", "TRA", "TRG", "IGH", "IGL"
`group.by`	Variable in which to group the diversity calculation
`x.axis`	Additional variable in which to split the x.axis
`split.by`	If using a single-cell object, the column header to group the new list. NULL will return clusters.
`exportTable`	Exports a table of the data into the global environment in addition to the visualization
`n.boots`	number of bootstraps to downsample in order to get mean diversity
`return.boots`	export boot strapped values calculated - will automatically exportTable = TRUE

Value

ggplot of the diversity of clonotype sequences across list

Author(s)

Andrew Malone, Nick Borcherding

Examples

```
#Making combined contig data
x <- contig_list
combined <- combineTCR(x, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")
clonalDiversity(combined, cloneCall = "gene")
```

---

clonalHomeostasis *Examining the clonal homeostasis*

Description

This function calculates the space occupied by clonotype proportions. The grouping of these clonotypes is based on the parameter cloneTypes, at default, cloneTypes will group the clonotypes into bins of Rare = 0 to 0.0001, Small = 0.0001 to 0.001, etc. To adjust the proportions, change the number or labeling of the cloneTypes paramter. If a matrix output for the data is preferred, set exportTable = TRUE.

Usage

```
clonalHomeostasis(
  df,
  cloneTypes = c(Rare = 1e-04, Small = 0.001, Medium = 0.01, Large = 0.1, Hyperexpanded =
    1),
  cloneCall = "strict",
  chain = "both",
  group.by = NULL,
  split.by = NULL,
  exportTable = FALSE
)
```

Arguments

df	The product of combineTCR(), combineBCR(), expression2List(), or combineExpression().
cloneTypes	The cutpoints of the proportions.
cloneCall	How to call the clonotype - VDJC gene (gene), CDR3 nucleotide (nt), CDR3 amino acid (aa), or VDJC gene + CDR3 nucleotide (strict).
chain	indicate if both or a specific chain should be used - e.g. "both", "TRA", "TRG", "IGH", "IGL"
group.by	The column header used for grouping.

split.by	If using a single-cell object, the column header to group the new list. NULL will return clusters.
exportTable	Exports a table of the data into the global environment in addition to the visualization

Value

ggplot of the space occupied by the specific proportion of clonotypes

Examples

```
#Making combined contig data
x <- contig_list
combined <- combineTCR(x, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")
clonalHomeostasis(combined, cloneCall = "gene")
```

---

clonalNetwork *Visualize clonal network along reduced dimensions*

Description

This function generates a network based on clonal proportions of an indicated identity and then superimposes the network onto a single-cell object dimensional reduction plot.

Usage

```
clonalNetwork(
  sc,
  reduction = "umap",
  identity = "ident",
  filter.clones = NULL,
  filter.identity = NULL,
  filter.proportion = NULL,
  filter.graph = FALSE,
  cloneCall = "strict",
  chain = "both",
  exportTable = FALSE
)
```

Arguments

sc	The Seurat or SingleCellExperiment (SCE) after combineExpression().
reduction	The name of the dimensional reduction of the single-cell object
identity	A variable in the meta data to use for the nodes.

filter.clones	Use to select the top n clones (filter.clones = 2000) or n of clones based on the minimum number of all the comparators (filter.clone = "min").
filter.identity	Display the network for a specific level of the indicated identity
filter.proportion	Remove clonotypes from the network below a specific proportion
filter.graph	Remove the reciprocal edges from the half of the graph, allowing for cleaner visualization
cloneCall	How to call the clonotype - VDJC gene (gene), CDR3 nucleotide (nt), CDR3 amino acid (aa), or VDJC gene + CDR3 nucleotide (strict).
chain	indicate if both or a specific chain should be used - e.g. "both", "TRA", "TRG", "IGH", "IGL"
exportTable	Exports a table of the data into the global environment in addition to the visualization

Value

ggplot object

Examples

```
## Not run:
#Getting the combined contigs
combined <- combineTCR(contig_list, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")

#Getting a sample of a Seurat object
screp_example <- get(data("screp_example"))

#Using combineExpresion()
screp_example <- combineExpression(combined, screp_example)

#Using clonalNetwork()
clonalNetwork(screp_example, reduction = "umap",
              identity = "cluster")

## End(Not run)
```

---

clonalOverlap *Examining the clonal overlap between groups or samples*

Description

This functions allows for the calculation and visualizations of the overlap coefficient, morisita, or jaccard index for clonotypes using the product of combineTCR(), combineBCR() or expression2list(). The overlap coefficient is calculated using the intersection of clonotypes divided by the length of the smallest component. If a matrix output for the data is preferred, set exportTable = TRUE.

Usage

```
clonalOverlap(
  df,
  cloneCall = "strict",
  method = c("overlap", "morisita", "jaccard", "raw"),
  chain = "both",
  split.by = NULL,
  exportTable = FALSE
)
```

Arguments

df	The product of combineTCR(), combineBCR(), expression2List(), or combineExpression().
cloneCall	How to call the clonotype - VDJC gene (gene), CDR3 nucleotide (nt), CDR3 amino acid (aa), or VDJC gene + CDR3 nucleotide (strict).
method	The method to calculate the overlap, either the "overlap" coefficient, "morisita", "jaccard" indices, or "raw" for the base numbers.
chain	indicate if both or a specific chain should be used - e.g. "both", "TRA", "TRG", "IGH", "IGL"
split.by	If using a single-cell object, the column header to group the new list. NULL will return clusters.
exportTable	Returns the data frame used for forming the graph

Value

ggplot of the clonotypic overlap between elements of a list

Examples

```
#Making combined contig data
x <- contig_list
combined <- combineTCR(x, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")

clonalOverlap(combined, cloneCall = "gene", method = "overlap")
```

---

clonalOverlay *Visualize distribution of clonal frequency overlaid on dimensional reduction plots*

Description

This function allows the user to visualize the clonal expansion by overlaying the cells with specific clonal frequency onto the dimensional reduction plots in Seurat. Credit to the idea goes to Drs Andreatta and Carmona and their work with ProjectTIL.

Usage

```
clonalOverlay(
  sc,
  reduction = NULL,
  freq.cutpoint = 30,
  bins = 25,
  facet = NULL
)
```

Arguments

sc	The seurat or SCE object to visualize after combineExpression().
reduction	The dimensional reduction to visualize
freq.cutpoint	The overlay cutpoint to include, this corresponds to the Frequency variable in the single-cell objecter
bins	The number of contours to the overlay
facet	meta data variable to facet the comparison

Value

ggplot object

Author(s)

Francesco Mazziotta, Nick Borcherding

Examples

```
#Getting the combined contigs
combined <- combineTCR(contig_list, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")

#Getting a sample of a Seurat object
screp_example <- get(data("screp_example"))
sce <- suppressMessages(Seurat::UpdateSeuratObject(screp_example))

#Using combineExpresion()
sce <- combineExpression(combined, sce)

#Using clonalOverlay()
clonalOverlay(sce, freq.cutpoint = 0.3, bins = 5)
```

clonalProportion *Examining the clonal space occupied by specific clonotypes*

Description

This function calculates the relative clonal space occupied by the clonotypes. The grouping of these clonotypes is based on the parameter split, at default, split will group the clonotypes into bins of 1:10, 11:100, 101:1001, etc. To adjust the clonotypes selected, change the numbers in the variable split. If a matrix output for the data is preferred, set exportTable = TRUE.

Usage

```
clonalProportion(
  df,
  split = c(10, 100, 1000, 10000, 30000, 1e+05),
  cloneCall = "strict",
  chain = "both",
  group.by = NULL,
  split.by = NULL,
  exportTable = FALSE
)
```

Arguments

df	The product of combineTCR(), combineBCR(), expression2List(), or combineExpression().
split	The cutpoints for the specific clonotypes.
cloneCall	How to call the clonotype - VDJC gene (gene), CDR3 nucleotide (nt), CDR3 amino acid (aa), or VDJC gene + CDR3 nucleotide (strict).
chain	indicate if both or a specific chain should be used - e.g. "both", "TRA", "TRG", "IGH", "IGL"
group.by	The column header used for grouping.
split.by	If using a single-cell object, the column header to group the new list. NULL will return clusters.
exportTable	Exports a table of the data into the global environment in addition to the visualization

Value

ggplot of the space occupied by the specific rank of clonotypes

Examples

```
#Making combined contig data
x <- contig_list
combined <- combineTCR(x, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")
clonalProportion(combined, cloneCall = "gene")
```

---

clonesizeDistribution *Hierarchical clustering of clonotypes on clonotype size and Jensen-Shannon divergence*

Description

This function produces a hierarchical clustering of clonotypes by sample using the Jensen-Shannon distance and discrete gamma-GPD spliced threshold model in powerTCR R package. Please read and cite PMID: 30485278 if using the function for analyses.

Usage

```
clonesizeDistribution(
  df,
  cloneCall = "strict",
  chain = "both",
  method = "ward.D2",
  threshold = 1,
  group.by = NULL,
  split.by = NULL,
  exportTable = FALSE
)
```

Arguments

df	The product of combineTCR(), combineBCR(), expression2List(), or combineExpression().
cloneCall	How to call the clonotype - VDJC gene (gene), CDR3 nucleotide (nt), CDR3 amino acid (aa), or VDJC gene + CDR3 nucleotide (strict).
chain	indicate if both or a specific chain should be used - e.g. "both", "TRA", "TRG", "IGH", "IGL"
method	The clustering parameter for the dendrogram.
threshold	Numerical vector containing the thresholds the grid search was performed over.
group.by	The column header used for grouping.
split.by	If using a single-cell object, the column header to group the new list. NULL will return clusters.
exportTable	Returns the data frame used for forming the graph.

Value

ggplot dendrogram of the clone size distribution

Examples

```
#Making combined contig data
x <- contig_list
combined <- combineTCR(x, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")
clonesizeDistribution(combined, cloneCall = "strict", method="ward.D2")
```

`clonotypeBias` *Examine clonotype bias*

Description

Clonotype bias method was developed and outlined from a single-cell manuscript characterizing CD4 responses to acute and chronic infection. The metric seeks to quantify how individual clones are skewed towards a specific cellular compartment or cluster. A clonotype bias of 1 indicates that a clonotype is composed of cells from a single compartment or cluster, while a clonotype bias of 0 matches the background subtype distribution.

Usage

```
clonotypeBias(
  df,
  cloneCall = "strict",
  split.by = NULL,
  group.by = NULL,
  n.boots = 20,
  min.expand = 10,
  exportTable = FALSE
)
```

Arguments

`df`	The product of combineTCR(), combineBCR(), expression2List(), or combine-Expression().
`cloneCall`	How to call the clonotype - VDJC gene (gene), CDR3 nucleotide (nt), CDR3 amino acid (aa), or VDJC gene + CDR3 nucleotide (strict).
`split.by`	The column header used for calculating the baseline frequencies. For example, "Type" for tumor vs peripheral blood comparison
`group.by`	The column header used for comparisons of bias.
`n.boots`	number of bootstraps to downsample
`min.expand`	clonotype frequency cut off for the purpose of comparison
`exportTable`	Returns the data frame used for forming the graph

Value

Returns ggplot of the clonotype bias

Examples

```
## Not run:
Getting the combined contigs
combined <- combineTCR(contig_list, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")

#Getting a sample of a Seurat object
screp_example <- get(data("screp_example"))
sce <- suppressMessages(Seurat::UpdateSeuratObject(screp_example))

#Using combineExpresion()
sce <- combineExpression(combined, sce)

#Using occupiedscRepertoire()
clonotypeBias(sce, cloneCall = "aa", split.by = "Patient", group.by = "cluster",
n.boots = 20, min.expand =10)

## End(Not run)
```

`clusterTCR` *Clustering T cell receptors*

Description

This function uses edit distances of either the nucleotide or amino acid sequences of the CDR3 to cluster similar TCRs together. The distance clustering will then be amended to the end of the list of combined contigs. The cluster will appear as CHAIN.num if a unique sequence or CHAIN:LD.num if clustered together. This function will only two chains recovered, multiple chains will automatically be reduced. This function also underlies the combineBCR() function and therefore not needed for B cells. This may take some time to calculate the distances and cluster.

Usage

```
clusterTCR(
  df,
  chain = NULL,
  sequence = NULL,
  threshold = 0.85,
  group.by = NULL
)
```

Arguments

df	The product of combineTCR(), expression2List(), or combineExpression().
chain	The TCR to cluster - TRA, TRB, TRG, TRD
sequence	Clustering based on either "aa" or "nt"
threshold	The normalized edit distance to consider. The higher the number the more similarity of sequence will be used for clustering.
group.by	The column header used for to calculate the cluster

Value

List of clonotypes for individual cell barcodes

Examples

```
combined <- combineTCR(contig_list, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")

sub_combined <- clusterTCR(combined[[2]], chain = "TRA", sequence = "aa")
```

---

combineBCR *Combining the list of B Cell Receptor contigs*

Description

This function consolidates a list of BCR sequencing results to the level of the individual cell barcodes. Using the samples and ID parameters, the function will add the strings as prefixes to prevent issues with repeated barcodes. The resulting new barcodes will need to match the seurat or SCE object in order to use, combineExpression. Unlike combineTCR(), combineBCR produces a column CTstrict of an index of nucleotide sequence and the corresponding v-gene. This index automatically caluclates the Levenshtein distance between sequences of the same length and will index sequences with <= 0.15 normalized Levenshtein distance with the same ID. After which, clonotype clusters are called using the igraph component() function. Clonotype that are clustered across multiple sequences will then be labeled with "LD" with the CTstrict header.

Usage

```
combineBCR(
  df,
  samples = NULL,
  ID = NULL,
  threshold = 0.85,
  removeNA = FALSE,
  removeMulti = FALSE
)
```

Arguments

df	List of filtered contig annotations from 10x Genomics.
samples	The labels of samples (required).
ID	The additional sample labeling (optional).
threshold	The normalized edit distance to consider. The higher the number the more similarity of sequence will be used for clustering.
removeNA	This will remove any chain without values.
removeMulti	This will remove barcodes with greater than 2 chains.

Value

List of clonotypes for individual cell barcodes

Examples

```
#Data derived from the 10x Genomics intratumoral NSCLC B cells
BCR <- read.csv("https://ncborcherding.github.io/vignettes/b_contigs.csv")
combined <- combineBCR(BCR, samples = "Patient1",
ID = "Time1", threshold = 0.85)
```

---

combineExpression *Adding clonotype information to a Seurat or SCE object*

Description

This function adds the immune receptor information to the Seurat or SCE object to the meta data. By default this function also calculates the frequencies of the clonotypes by sequencing run (group.by = "none"). To change how the frequencies are calculated, select a column header for the group.by variable. Importantly, before using combineExpression() ensure the barcodes of the seurat or SCE object match the barcodes in the output of the combinedContig() call. Check changeNames() to change the prefix of the Seurat object. If combining more than one immune receptor type, barcodes with both receptors will be removed during the combination process.

Usage

```
combineExpression(
  df,
  sc,
  cloneCall = "strict",
  chain = "both",
  group.by = "none",
  proportion = TRUE,
  filterNA = FALSE,
 cloneTypes = c(Rare = 1e-04, Small = 0.001, Medium = 0.01, Large = 0.1, Hyperexpanded =
    1),
  addLabel = FALSE
)
```

Arguments

df	The product of CombineTCR() or CombineBCR() or a list of both c(CombineTCR(), combineBCR())
sc	The seurat or SingleCellExperiment (SCE) object to attach
cloneCall	How to call the clonotype - VDJC gene (gene), CDR3 nucleotide (nt) CDR3 amino acid (aa), or VDJC gene + CDR3 nucleotide (strict).
chain	indicate if both or a specific chain should be used - e.g. "both", "TRA", "TRG", "IGH", "IGL"
group.by	The column label in the combined contig object in which clonotype frequency will be calculated.
proportion	Whether to use the total frequency (FALSE) or the proportion (TRUE) of the clonotype based on the group.by variable.
filterNA	Method to subset seurat object of barcodes without clonotype information
cloneTypes	The bins for the grouping based on frequency
addLabel	This will add a label to the frequency header, allowing the user to try multiple group.by variables or recalculate frequencies after subseting the data.

Value

seurat or SingleCellExperiment object with attached clonotype information

Examples

```
#Getting the combined contigs
combined <- combineTCR(contig_list, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")

#Getting a sample of a Seurat object
screp_example <- get(data("screp_example"))
sce <- suppressMessages(Seurat::UpdateSeuratObject(screp_example))
sce <- Seurat::as.SingleCellExperiment(sce)

#Using combineExpresion()
sce <- combineExpression(combined, sce)
```

---

combineTCR *Combining the list of T Cell Receptor contigs*

Description

This function consolidates a list of TCR sequencing results to the level of the individual cell barcodes. Using the samples and ID parameters, the function will add the strings as prefixes to prevent issues with repeated barcodes. The resulting new barcodes will need to match the Seurat or SCE object in order to use, combineExpression. Several levels of filtering exist - remove or filterMulti are parameters that control how the function deals with barcodes with multiple chains recovered.

Usage

```
combineTCR(
  df,
  samples = NULL,
  ID = NULL,
  cells = "T-AB",
  removeNA = FALSE,
  removeMulti = FALSE,
  filterMulti = FALSE
)
```

Arguments

df	List of filtered contig annotations from 10x Genomics.
samples	The labels of samples (required).
ID	The additional sample labeling (optional).
cells	The type of T cell - T cell-AB or T cell-GD. Only 1 T cell type can be called at once.
removeNA	This will remove any chain without values.
removeMulti	This will remove barcodes with greater than 2 chains.
filterMulti	This option will allow for the selection of the 2 corresponding chains with the highest expression for a single barcode.

Value

List of clonotypes for individual cell barcodes

Examples

```
combineTCR(contig_list,
           samples = rep(c("PX", "PY", "PZ"), each=2),
           ID = rep(c("P", "T"), 3),
           cells ="T-AB")
```

---

combineTRUST4 *Combining the list of T or B cell Receptors from TRUST4 pipeline*

Description

This function consolidates a list of TCR/BCR sequencing results to the level of the individual cell barcodes using the same approach as combineTCR and combineBCR. Using the samples and ID parameters, the function will add the strings as prefixes to prevent issues with repeated barcodes. The resulting new barcodes will need to match the seurat or SCE object in order to use, combineExpression. Several levels of filtering exist - removeMulti are parameters that control how the function deals with barcodes with multiple chains recovered. Please read more and cite the TRUST4 pipeline if using this function.

Usage

```
combineTRUST4(
  df,
  samples = NULL,
  ID = NULL,
  cells = c("T-AB", "T-GD", "B"),
  removeNA = FALSE,
  threshold = 0.85
)
```

Arguments

df	List of Contig outputs from TRUST4
samples	The labels of samples (required).
ID	The additional sample labeling (optional).
cells	The type of cell - T cell-AB or T cell-GD, or B cell
removeNA	This will remove any chain without values.
threshold	If combining B cells - the normalized edit distance to consider. The higher the number the more similarity of sequence will be used for clustering.

Value

List of clonotypes for individual cell barcodes

Examples

```
## Not run:
combineTRUST4(contig_list, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")

## End(Not run)
```

---

compareClonotypes *Demonstrate the difference in clonal proportion between clonotypes*

Description

This function produces an alluvial or area graph of the proportion of the indicated clonotypes for all or selected samples. Clonotypes can be selected using the clonotypes parameter with the specific sequence of interest or using the number parameter with the top n clonotypes by proportion to be visualized.

Usage

```
compareClonotypes(
  df,
  cloneCall = "strict",
  chain = "both",
  samples = NULL,
  clonotypes = NULL,
  numbers = NULL,
  split.by = NULL,
  graph = "alluvial",
  exportTable = FALSE
)
```

Arguments

df	The product of combineTCR(), combineBCR(), expression2List(), or combine-Expression().
cloneCall	How to call the clonotype - VDJC gene (gene), CDR3 nucleotide (nt), CDR3 amino acid (aa), or VDJC gene + CDR3 nucleotide (strict).
chain	indicate if both or a specific chain should be used - e.g. "both", "TRA", "TRG", "IGH", "IGL"
samples	The specific samples to isolate for visualization.
clonotypes	The specific sequences of interest.
numbers	The top number clonotype sequences per group
split.by	If using a single-cell object, the column header to group the new list. NULL will return clusters.
graph	The type of graph produced, either "alluvial" or "area".
exportTable	Returns the data frame used for forming the graph.

Value

ggplot of the proportion of total sequencing read of selecting clonotypes

Examples

```
#Making combined contig data
x <- contig_list
combined <- combineTCR(x, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")
compareClonotypes(combined, numbers = 10,
samples = c("PX_P", "PX_T"), cloneCall="aa")
```

`contig_list` *A data set of T cell contigs as a list outputed from the filter_contig_annotation files.*

Description

A data set of T cell contigs as a list outputed from the filter_contig_annotation files.

`createHTOContigList` *Generate a contig list from a multiplexed experiment*

Description

Multiplexing single-cell sequencing runs is an efficient method for quantifying multiple samples or conditions simultaneously. Unfortunately, the hashing information is not stored in the TCR sequence data. In order preprocess and form a contig list for downstream analysis in scRepertoire, createHTOContigList() take the filtered contig annotation output and the single-cell RNA object to create the list. If using an integrated single-cell object, it is recommended to split the object by sequencing run and remove extra prefixes and suffixes on the barcode before using createHTOContigList(). Alternatively, the variable multi.run can be used to separate a list of contigs by a meta data variable. This may have issues with the repeated barcodes.

Usage

```
createHTOContigList(contig, sc, group.by = NULL, multi.run = NULL)
```

Arguments

`contig`	The filtered contig annotation file from multiplexed experiment
`sc`	The Seurat or SCE object.
`group.by`	One or more meta data headers to create the contig list based on. If more than one header listed, the function combines them into a single variable.
`multi.run`	If using integrated single-cell object, the meta data variable that indicates the sequencing run.

Value

Returns a list of contigs corresponding to the multiplexed Seurat or Single-Cell Experiment object

Examples

```
## Not run:
filtered.contig <- read.csv(".../Sample/outs/filtered_contig_annotations.csv")
contig.list <- createHTOContigList(contig = filtered.contig,
sc = Seurat.Obj, group.by = "HTO_maxID")

## End(Not run)
```

`expression2List` *Allows users to take the meta data in Seurat/SCE and place it into a list that will work with all the functions*

Description

Allows users to perform more fundamental measures of clonotype analysis using the meta data from the Seurat or SCE object. For Seurat objects the active identity is automatically added as "cluster". Remaining grouping parameters or SCE or Seurat objects must appear in the meta data.

Usage

```
expression2List(sc, split.by)
```

Arguments

`sc`	object after combineExpression().
`split.by`	The column header to group the new list. NULL will return clusters.

Value

list derived from the meta data of single-cell object with elements divided by the group parameter

Examples

```
#Getting the combined contigs
combined <- combineTCR(contig_list, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")

#Getting a sample of a Seurat object
screp_example <- get(data("screp_example"))
screp_example <- combineExpression(combined, screp_example)

#Using expression2List
newList <- expression2List(screp_example, split.by = "seurat_clusters")
```

`getCirclize` *Generate data frame to be used with circlize R package to visualize clonotypes as a chord diagram.*

Description

This function will take the meta data from the product of combineExpression()and generate a relational data frame to be used for a chord diagram. Each cord will represent the number of clonotype unqiue and shared across the multiple group.by variable.

Usage

```
getCirclize(sc, cloneCall = "strict", group.by = NULL, proportion = FALSE)
```

Arguments

`sc`	object after combineExpression().
`cloneCall`	How to call the clonotype - VDJC gene (gene), CDR3 nucleotide (nt), CDR3 amino acid (aa), or VDJC gene + CDR3 nucleotide (strict).
`group.by`	The group header for which you would like to analyze the data.
`proportion`	Binary will calculate relationship unique clonotypes (proportion = FALSE) or a ratio of the group.by variable (proportion = TRUE)

Value

data frame of shared clonotypes between groups

Author(s)

Dillon Corvino, Nick Borcherding

Examples

```
#Getting the combined contigs
combined <- combineTCR(contig_list, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")

#Getting a sample of a Seurat object
screp_example <- get(data("screp_example"))
screp_example <- combineExpression(combined, screp_example)

#Getting data frame output for Circilize
circles <- getCirclize(screp_example, group.by = "seurat_clusters")
```

highlightClonotypes *Highlighting specific clonotypes in Seurat*

Description

Use a specific clonotype sequence to highlight on top of the dimensional reduction in seurat object.

Usage

```
highlightClonotypes(
  sc,
  cloneCall = c("gene", "nt", "aa", "strict"),
  sequence = NULL
)
```

Arguments

sc	The Seurat object to attach
cloneCall	How to call the clonotype - VDJC gene (gene), CDR3 nucleotide (nt), CDR3 amino acid (aa), or VDJC gene + CDR3 nucleotide (strict).
sequence	The specific sequence or sequence to highlight

Value

Seurat object with new meta data column for indicated clones

Examples

```
#' #Getting the combined contigs
combined <- combineTCR(contig_list, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")

#Getting a sample of a Seurat object
screp_example <- get(data("screp_example"))

#Using combineExpresion()
screp_example <- combineExpression(combined, screp_example )

#Using highlightClonotype()
screp_example  <- highlightClonotypes(screp_example, cloneCall= "aa",
sequence = c("CAVNGGSQGNLIF_CSAEREDTDTQYF"))
```

`lengthContig` *Demonstrate the distribution of lengths filtered contigs.*

Description

This function takes the output of combineTCR(), combineBCR(), or expression2List() and displays either the nucleotide (nt) or amino acid (aa) sequence length. The sequence length visualized can be selected using the chains parameter, either the combined clonotype (both chains) or across all single chains. Visualization can either be a histogram or if scale = TRUE, the output will be a density plot. Multiple sequencing runs can be group together using the group parameter. If a matrix output for the data is preferred, set exportTable = TRUE.

Usage

```
lengthContig(
  df,
  cloneCall = "aa",
  chain = "both",
  group.by = NULL,
  split.by = NULL,
  order = TRUE,
  scale = FALSE,
  exportTable = FALSE
)
```

Arguments

`df`	The product of combineTCR(), combineBCR(), expression2List(), or combineExpression().
`cloneCall`	How to call the clonotype - CDR3 nucleotide (nt), CDR3 amino acid (aa).
`chain`	indicate if both or a specific chain should be used - e.g. "both", "TRA", "TRG", "IGH", "IGL"
`group.by`	The group header for which you would like to analyze the data.
`split.by`	If using a single-cell object, the column header to group the new list. NULL will return clusters.
`order`	Maintain the order of the list when plotting
`scale`	Converts the graphs into density plots in order to show relative distributions.
`exportTable`	Returns the data frame used for forming the graph.

Value

ggplot of the discrete or relative length distributions of clonotype sequences

Examples

```
#Making combined contig data
x <- contig_list
combined <- combineTCR(x, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")
lengthContig(combined, cloneCall="aa", chain = "both")
```

`loadContigs` *Loading the contigs derived from single-cell sequencing*

Description

This function generates a contig list and formats the data to allow for function with `combineTCR` or `combineTCR`. If using data derived from filtered outputs of 10X Genomics, there is no need to use this function as the data is already compatible. The function assumes if listing multiple directories, there are distinct outputs with unmodified file names in them.

Usage

```
loadContigs(dir, format = "10X")
```

Arguments

`dir`	The directory or directories with single-cell contig data
`format`	The format of the single-cell contig, currently supporting: "10X", "AIRR", "TRUST4", and "WAT3R"

Value

List of contigs for further processing in scRepertoire

Examples

```
## Not run:
dir <- c("./Sample1/outs/", "./Sample2/outs/", "./Sample3/outs/")
contig.list <- loadContigs(dir, format = "10X")

## End(Not run)
```

`occupiedscRepertoire` *Visualize the number of single cells with clonotype frequencies by cluster*

Description

View the count of clonotypes frequency group in seurat or SCE object meta data after combineExpression(). The visualization will take the new meta data variable "cloneType" and plot the number of cells with each designation using a secondary variable, like cluster. Credit to the idea goes to Drs. Carmona and Andreatta and their work with ProjectTIL.

Usage

```
occupiedscRepertoire(
  sc,
  x.axis = "ident",
  label = TRUE,
  facet.by = NULL,
  proportion = FALSE,
  na.include = FALSE,
  exportTable = FALSE
)
```

Arguments

`sc`	The Seurat or SCE object to visualize after combineExpression().
`x.axis`	The variable in the meta data to graph along the x.axis
`label`	Include the number of clonotype in each category by x.axis variable
`facet.by`	The column header used for faceting the graph
`proportion`	Convert the stacked bars into relative proportion
`na.include`	Visualize NA values or not.
`exportTable`	Exports a table of the data into the global environment in addition to the visualization

Value

Stacked bar plot of counts of cells by clonotype frequency group

Examples

```
#Getting the combined contigs
combined <- combineTCR(contig_list, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")

#Getting a sample of a Seurat object
screp_example <- get(data("screp_example"))
```

```
sce <- suppressMessages(Seurat::UpdateSeuratObject(screp_example))

#Using combineExpresion()
sce <- combineExpression(combined, sce)

#Using occupiedscRepertoire()
occupiedscRepertoire(sce, x.axis = "ident")
table <- occupiedscRepertoire(sce, x.axis = "ident", exportTable = TRUE)
```

---

quantContig *Quantify the unique clonotypes in the filtered contigs.*

Description

This function takes the output from combineTCR(), combineBCR(), or expression2List() and quantifies unique clonotypes. The unique clonotypes can be either reported as a raw output or scaled to the total number of clonotypes recovered using the scale parameter. Multiple sequencing runs can be group together using the group parameter. If a matrix output for the data is preferred, set exportTable = TRUE.

Usage

```
quantContig(
  df,
  cloneCall = "strict",
  chain = "both",
  scale = FALSE,
  group.by = NULL,
  split.by = NULL,
  order = TRUE,
  exportTable = FALSE
)
```

Arguments

df	The product of combineTCR(), combineBCR(), expression2List(), or combineExpression().
cloneCall	How to call the clonotype - VDJC gene (gene), CDR3 nucleotide (nt), CDR3 amino acid (aa), or VDJC gene + CDR3 nucleotide (strict).
chain	indicate if both or a specific chain should be used - e.g. "both", "TRA", "TRG", "IGH", "IGL"
scale	Converts the graphs into percentage of unique clonotypes.
group.by	The column header used for grouping.
split.by	If using a single-cell object, the column header to group the new list. NULL will return clusters.
order	Maintain the order of the list when plotting
exportTable	Returns the data frame used for forming the graph

Value

ggplot of the total or relative unique clonotypes

Examples

```
#Making combined contig data
x <- contig_list
combined <- combineTCR(x, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")
quantContig(combined, cloneCall="strict", scale = TRUE)
```

---

scatterClonotype *Scatter plot comparing the expansion of two samples*

Description

This function produces a scatter plot directly comparing the specific clonotypes between two samples. The clonotypes will be categorized by counts into singlets or multiplets, either exclusive or shared between the selected samples. Visualization inspired by the work of Wu, T, et al 2020.

Usage

```
scatterClonotype(
  df,
  cloneCall = "strict",
  x.axis = NULL,
  y.axis = NULL,
  chain = "both",
  dot.size = "total",
  split.by = NULL,
  graph = "proportion",
  exportTable = FALSE
)
```

Arguments

df	The product of combineTCR(), combineBCR(), expression2List(), or combineExpression().
cloneCall	How to call the clonotype - VDJC gene (gene), CDR3 nucleotide (nt), CDR3 amino acid (aa), or VDJC gene + CDR3 nucleotide (strict).
x.axis	name of the list element to appear on the x.axis
y.axis	name of the list element to appear on the y.axis
chain	indicate if both or a specific chain should be used - e.g. "both", "TRA", "TRG", "IGH", "IGL"
dot.size	either total or the name of the list element to use for size of dots

split.by	If using a single-cell object, the column header to group the new list. NULL will return clusters.
graph	graph either proportion or raw clonotype count
exportTable	Returns the data frame used for forming the graph.

Value

ggplot of the relative clonotype numbers

Examples

```
#Making combined contig data
x <- contig_list
combined <- combineTCR(x, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")
scatterClonotype(combined, x.axis = "PY_P", y.axis = "PY_T",
graph = "proportion")
```

---

screp_example *A seurat object of 1000 single T cells derived from 3 clear cell renal carcinoma patients.*

Description

A seurat object of 1000 single T cells derived from 3 clear cell renal carcinoma patients.

Startrac *The Startrac Class*

Description

The Startrac object store the data for tcr-based T cell dynamics analyis. The slots contained in Startrac object are listed below:

Value

method definition for runing startrac

Slots

`aid` character. aid of the object, used for identification of the object. For example, patient id. default: "AID"

`cell.data` data.frame. Each line for a cell, and these columns as required: 'Cell_Name', 'clone.id', 'patient', 'majorCluster', 'loc'

`cell.perm.data` object. list of 'Startrac" objects constructed from permutated cell data

`clonotype.data` data.frame. Each line for a clonotype; contain the clonotype level indexes information

`cluster.data` data.frame. Each line for a cluster; contain the cluster level indexes information

`pIndex.migr` data.frame. Each line for a cluster; pairwise migration index between the two locations indicated in the column name.

`pIndex.tran` data.frame. Each line for a cluster; pairwise transition index betwwen the two major clusters indicated by the row name and column name.

`cluster.sig.data` data.frame. Each line for a cluster; contains the p values of cluster indices.

`pIndex.sig.migr` data.frame. Each line for a cluster; contains the p values of pairwise migration indices.

`pIndex.sig.tran` data.frame. Each line for a cluster; contains the p values of pairwise transition indices.

`clonotype.dist.loc` matrix. Each line for a clonotype and describe the cells distribution among the locations.

`clonotype.dist.cluster` matrix. Each line for a clonotype and describe the cells distribution among the clusters.

`clust.size` array. Number of cells of each major cluster.

`patient.size` array. Number of cells of each patient.

`clone.size` array. Number of cells of each clone.

`clone2patient` array. Mapping from patient id to clone id.

`StartracDiversity` *Diversity indices for single-cell RNA-seq*

Description

This function utilizes the Startrac R package derived from PMID: 30479382 Required to run the function, the "type" variable needs to include the difference in where the cells were derived. The output of this function will produce 3 indices: expa (clonal expansion), migra (cross-tissue migration), and trans (state transition). In order to understand the underlying analyses of the outputs please read and cite the linked manuscript.

Usage

```
StartracDiversity(
  sc,
  type = "Type",
  sample = NULL,
  by = "overall",
  exportTable = FALSE
)
```

Arguments

sc	The seurat or SCE object to visualize after combineExpression(). For SCE objects, the cluster variable must be in the meta data under "cluster".
type	The column header in the meta data that gives the where the cells were derived from, not the patient sample IDs
sample	The column header corresponding to individual samples or patients.
by	Method to subset the indices by either overall (across all samples) or by specific group
exportTable	Returns the data frame used for forming the graph

Value

ggplot object of Startrac diversity metrics

Examples

```
## Not run:
#Getting the combined contigs
combined <- combineTCR(contig_list, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")

#Getting a sample of a Seurat object
screp_example <- get(data("screp_example"))
screp_example <- combineExpression(combined, screp_example)

#Using occupiedscRepertoire()
StartracDiversity(screp_example, type = "Type", sample = "Patient", by = "overall")

## End(Not run)
```

---

stripBarcode *Removing any additional prefixes to the barcodes of filtered contigs.*

Description

Removing any additional prefixes to the barcodes of filtered contigs.

Usage

```
stripBarcode(contigs, column = 1, connector = "_", num_connects = 3)
```

Arguments

contigs	The raw loaded filtered_contig_annotation.csv
column	The column in which the barcodes are listed
connector	The type of character in which is attaching the defualt barcode with any other characters
num_connects	The number of strings combined with the connectors

Value

list with the suffixes of the barcodes removed.

Examples

```
stripBarcode(contig_list[[1]], column = 1, connector = "_", num_connects = 1)
```

---

subsetContig *Subset the product of combineTCR() combineBCR() or expression2List()*

Description

This function allows for the subsetting of the product of combineTCR() combineBCR() or expression2List() by the name of the individual list element. In general the names of are samples + _ + ID, allowing for users to subset the product of combineTCR(), combineBCR(), or expression2List() across a string or individual name.

Usage

```
subsetContig(df, name, variables = NULL)
```

Arguments

df	The product of combineTCR(), combineBCR(), or expression2List().
name	The column header you'd like to use to subset.
variables	The values to subset by, must be in the names(df).

Value

list of contigs that have been filtered for the name parameter

Examples

```
x <- contig_list
combined <- combineTCR(x, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")
subset <- subsetContig(combined, name = "sample", variables = c("PX"))
```

---

vizGenes *Visualizing the distribution of gene usage*

Description

This function will allow for the visualizing the distribution of the any VDJ and C gene of the TCR or BCR using heatmap or bar chart. This function requires assumes two chains were used in defining clonotype, if not, it will default to the only chain present regardless of the chain parameter.

Usage

```
vizGenes(
  df,
  gene = "V",
  chain = "TRA",
  plot = "heatmap",
  y.axis = "sample",
  order = "gene",
  scale = TRUE,
  group.by = NULL,
  split.by = NULL,
  exportTable = FALSE
)
```

Arguments

df	The product of combineTCR(), combineBCR(), expression2List(), or combineExpression().
gene	Which part of the immune receptor to visualize - V, D, J, C
chain	indicate the specific chain should be used - e.g. "TRA", "TRG", "IGH", "IGL" (no both option here)
plot	The type of plot to return - heatmap or bar
y.axis	Variable to separate the y-axis, can be both categorical or other gene gene segments such as V, D, J, or C.
order	Categorical variable to organize the x-axis, either "gene" or "variance"
scale	Converts the proportion of total genes
group.by	The column header used for grouping.
split.by	If using a single-cell object, the column header to group the new list. NULL will return clusters.
exportTable	Returns the data frame used for forming the graph.

Value

ggplot bar diagram or heatmap of gene usage

Examples

```
#Making combined contig data
x <- contig_list
combined <- combineTCR(x, rep(c("PX", "PY", "PZ"), each=2),
rep(c("P", "T"), 3), cells ="T-AB")

vizGenes(combined, gene = "V", chain = "TRB", plot = "bar", scale = TRUE)
```

Index