Package ‘EpiCompare’

February 22, 2024

**Type** Package

**Title** Comparison, Benchmarking & QC of Epigenomic Datasets

**Version** 1.6.5

**Description** EpiCompare is used to compare and analyse epigenetic datasets for quality control and benchmarking purposes. The package outputs an HTML report consisting of three sections:
1. General metrics) Metrics on peaks (percentage of blacklisted and non-standard peaks, and peak widths) and fragments (duplication rate) of samples,
2. Peak overlap) Percentage and statistical significance of overlapping and non-overlapping peaks. Also includes upset plot and
3. Functional annotation) functional annotation (ChromHMM, ChIPseeker and enrichment analysis) of peaks. Also includes peak enrichment around TSS.

**License** GPL-3

**URL** https://github.com/neurogenomics/EpiCompare

**BugReports** https://github.com/neurogenomics/EpiCompare/issues

**Depends** R (>= 4.2.0)

**Imports** AnnotationHub, BRGenomics, ChIPseeker, data.table, genomation, GenomicRanges, IRanges, GenomeInfoDb, ggplot2, htmltools, methods, plotly, reshape2, rmarkdown, rtracklayer, stats, stringr, utils, BiocGenerics, downloadthis, parallel

VignetteBuilder knitr
biocViews Epigenetics, Genetics, QualityControl, ChIPSeq,
        MultipleComparison, FunctionalGenomics, ATACSeq, DNaseSeq

Config/testthat/edition 3
Encoding UTF-8
LazyData FALSE
RoxygenNote 7.2.3
git_url https://git.bioconductor.org/packages/EpiCompare
git_branch RELEASE_3_18
git_last_commit a863246
git_last_commit_date 2024-02-13
Repository Bioconductor 3.18
Date/Publication 2024-02-22
Author Sera Choi [aut] (<https://orcid.org/0000-0002-5077-1984>),
        Brian Schilder [aut] (<https://orcid.org/0000-0001-5949-2191>),
        Leyla Abbasova [aut],
        Alan Murphy [aut] (<https://orcid.org/0000-0002-2487-8753>),
        Nathan Skene [aut] (<https://orcid.org/0000-0002-6807-3180>),
        Thomas Roberts [cre]

Maintainer Thomas Roberts <tomroberts.work15@gmail.com>

R topics documented:

as_interactive ......................................................... 4
bplapply ............................................................... 4
checkCache ............................................................. 6
check_cell_lines ...................................................... 6
check_genome_build .................................................. 7
check_grlist_cols .................................................... 7
check_list_names ..................................................... 8
check_workers ........................................................ 8
clean_granges ......................................................... 9
CnR_H3K27ac ........................................................... 9
CnR_H3K27ac_picard .................................................. 10
CnT_H3K27ac .......................................................... 11
CnT_H3K27ac_picard .................................................. 11
compute_consensus_peaks .......................................... 12
compute_corr ........................................................ 14
download_button ...................................................... 16
encode_H3K27ac ...................................................... 17
EpiCompare ........................................................... 18
fig_length ............................................................ 22
fragment_info ......................................................... 22
gather_files ........................................................ 23
gather_files_names .......................................................... 24
get_bpparam ................................................................. 25
get_chromHMM_annotation ................................................. 26
group_files ................................................................. 27
hg19_blacklist .............................................................. 27
hg38_blacklist .............................................................. 28
is_granges ................................................................. 29
liftover_grlist .............................................................. 29
messenger ................................................................. 30
message_parallel .......................................................... 31
mm10_blacklist ............................................................. 31
mm9_blacklist ............................................................. 32
overlap_heatmap ............................................................ 32
overlap_percent ............................................................. 33
overlap_stat_plot .......................................................... 34
overlap_upset_plot .......................................................... 35
peak_info ................................................................. 36
plot_ChIPseeker_annotation .............................................. 37
plot_chromHMM ............................................................ 38
plot_corr ................................................................. 39
plot_enrichment ............................................................ 41
plot_precision_recall ...................................................... 42
precision_recall ............................................................ 44
precision_recall_matrix .................................................. 45
predict_precision_recall ................................................... 46
predict_values ............................................................. 47
prepare_blacklist ........................................................... 47
prepare_genome_builds .................................................... 48
prepare_peaklist ........................................................... 49
prepare_reference .......................................................... 50
read_bowtie ............................................................... 50
read_peaks ................................................................. 51
rebin_peaks ............................................................... 51
remove_nonstandard_chrom ............................................... 53
report_command ............................................................ 54
report_header ............................................................. 54
save_output ............................................................... 55
set_min_max ............................................................... 55
stopper ................................................................. 56
tidy_peakfile ............................................................. 56
translate_genome ........................................................... 57
tss_plot ................................................................. 58
width_boxplot ............................................................. 59
write_example_peaks ......................................................... 60

Index 61
Description

Convert a `ggplot` object to `plotly`, and enable it to be plotted within an Rmarkdown HTML file.

Usage

```r
as_interactive(
  plt,
  to_widget = isTRUE(getOption("knitr.in.progress")),
  add_boxmode = FALSE
)
```

Arguments

- `plt` ggplot object.
- `to_widget` Convert to a widget so it works within Rmarkdown HTML files. By default, this will be only be set to `TRUE` when being run within the context of `knitr` rendering.
- `add_boxmode` Add extra layout to enable dodged boxplots.

Value

A `plotly` object or a `tagList` wrapping the `plotly` object.

Source

GitHub Issue to check whether knitting

bpplapply Wrapper for `bplapply`

Description

Wrapper function for `bplapply` that automatically handles issues with `BiocParallel` related to different OS platforms.
bpplapply

Usage

bpplapply(
  X,
  FUN,
  apply_fun = parallel::mclapply,
  workers = check_workers(),
  progressbar = workers > 1,
  verbose = workers == 1,
  use_snowparam = TRUE,
  register_now = FALSE,
  ...
)

Arguments

X Any object for which methods length, [, and [[ are implemented.
FUN The function to be applied to each element of X.
apply_fun Iterator function to use.
workers Number of threads to parallelize across.
progressbar logical(1) Enable progress bar (based on plyr:::progress_text).
verbose Print messages.
use_snowparam Whether to use SnowParam (default: TRUE) or MulticoreParam (FALSE) when parallelising across multiple workers.
register_now Register the cores now with register (TRUE), or simply return the BPPARAM object (default: FALSE).
... Arguments passed on to BiocParallel::bplapply

BPPARAM An optional BiocParallelParam instance determining the parallel back-end to be used during evaluation, or a list of BiocParallelParam instances, to be applied in sequence for nested calls to BiocParallel functions.

BPREDO A list of output from bplapply with one or more failed elements. When a list is given in BPREDO, bpok is used to identify errors, tasks are rerun and inserted into the original results.

BPOPTIONS Additional options to control the behavior of the parallel evaluation, see bpoptions.

Value

(Named) list.

Examples

X <- stats::setNames(seq_len(length(letters)), letters)
out <- bpplapply(X, print)
checkCache

Description

Quick function to check if object is already saved.

Usage

checkCache(cache = BiocFileCache::BiocFileCache(ask = FALSE), url)

Arguments

- cache: BiocFileCache.
- url: Path to cached file.

Value

path

check_cell_lines

Description

Check whether a list of cell lines matches any of those that are made available through EpiCompare.

Usage

check_cell_lines(cell_lines = NULL, verbose = TRUE)

Arguments

- cell_lines: A character vector of cell line names. If NULL (default), will return names of all cell lines.
- verbose: Print messages.

Value

Character vector, or NULL.
check_genome_build

Description
Check that the genome build is valid and require specific reference datasets to be installed.

Usage
check_genome_build(genome_build, type = "txdb")

Arguments
- genome_build: Genome build name.
- type: whether to fetch the txdb or bsgen reference data

Value
txdb or bsgen

check_grlist_cols Check GRanges list columns

Description
Check that at least one of the required columns is in a list of GRanges objects. Elements that do not meet this criterion will be dropped from the list.

Usage
check_grlist_cols(grlist, target_cols)

Arguments
- grlist: Named list of GRanges objects.
- target_cols: A character vector of column names to search for.

Value
Named list of GRanges objects.
check_list_names

**Description**

This function checks whether the peaklist is named. If not, default file names are assigned.

**Usage**

```
check_list_names(peaklist, default_prefix = "sample")
```

**Arguments**

- `peaklist`: A list of peak files as GRanges object.
- `default_prefix`: Default prefix to use when creating names for peaklist.

**Value**

- named peaklist

check_workers

**Description**

Assign parallel worker cores.

**Usage**

```
check_workers(workers = NULL)
```

**Arguments**

- `workers`: Number of cores to parallelise across (in applicable functions). If NULL, will set to the total number of available cores minus 1.

**Value**

- Integer

**Examples**

```
workers <- check_workers()
```
**clean_granges**

---

**Clean GRanges**

**Description**

Remove columns from the metadata (GenomicRanges::mcols) that conflicts with GRanges conventions.

**Usage**

```r
clean_granges(
  gr,
  nono_cols = c("seqnames", "ranges", "strand", "seqlevels", "seqlengths", "isCircular",
                 "start", "end", "width", "element")
)
```

**Arguments**

- `gr`: A GRanges object.
- `nono_cols`: Problematic columns to search for and remove (if present).

**Value**

Cleaned GRanges object.

---

**CnR_H3K27ac**

---

**Example CUT&Run peak file**

**Description**

Human H3K27ac peak file generated with CUT&Run using K562 cell-line from Meers et al., (2019). Human genome build hg19 was used. Raw peak file (.BED) was obtained from GEO (https://trace.ncbi.nlm.nih.gov/Traces/sra/?run=SRR8581604). Peak calling was performed by Leyla Abbasova using MACS2. The peak file was then processed into GRanges object. Peaks located on chromosome 1 were subsetted to reduce the dataset size.

**Usage**

```r
data("CnR_H3K27ac")
```

**Format**

An object of class GRanges of length 2707.
Source

The code to prepare the .Rda file from the raw peak file is:

```r
# sequences were directly downloaded from https://trace.ncbi.nlm.nih.gov/Traces/sra/?run=SRR8581604
# and peaks (BED file) were generated by Leyla Abbasova (Neurogenomics Lab, Imperial College London)
CnR_H3K27ac <- ChIPseeker::readPeakFile("path", as = "GRanges")
CnR_H3K27ac <- CnR_H3K27ac[seqnames(CnR_H3K27ac) == "chr1"]
my_label <- c("name","score","strand","signalValue","pValue","qValue","peak")
colnames(GenomicRanges::mcols(CnR_H3K27ac)) <- my_label
usethis::use_data(CnR_H3K27ac, overwrite = TRUE)
```

---

**CnR_H3K27ac_picard**  
*Example Picard duplication metrics file 2*

---

Description

Duplication metrics output on CUT&Run H3K27ac file (sample accession: SRR8581604). Raw sequences were aligned to hg19 genome and after, Picard was performed by Leyla Abbasova. The duplication summary output generated by Picard was processed to reduce the size of data.

Usage

```r
data("CnR_H3K27ac_picard")
```

Format

An object of class `data.frame` with 1 rows and 10 columns.

Source

The code to prepare the .Rda file is:

```r
picard <- read.table("path/to/picard/duplication/output",header = TRUE, fill = TRUE)
CnR_H3K27ac_picard <- picard[1,]
usethis::use_data(CnR_H3K27ac_picard, overwrite = TRUE)
```
CnT_H3K27ac

Example CUT&Tag peak file

Description

Human H3K27ac peak file generated with CUT&Tag using K562 cell-line from Kaya-Okur et al., (2019). Human genome build hg19 was used. Raw peak file (.BED) was obtained from GEO (https://trace.ncbi.nlm.nih.gov/Traces/sra/?run=SRR8383507). Peak calling was performed by Leyla Abbasova using MACS2. The peak file was then imported as an GRanges object. Peaks located on chromosome 1 were subsetted to reduce the dataset size.

Usage

data("CnT_H3K27ac")

Format

An object of class GRanges of length 1670.

Source

The code to prepare the .Rda file from the raw peak file is:

# sequences were directly downloaded from https://trace.ncbi.nlm.nih.gov/Traces/sra/?run=SRR8383507
# and peaks (BED file) were generated by Leyla Abbasova (Neurogenomics Lab, Imperial College London)
CnT_H3K27ac <- ChIPseeker::readPeakFile("path", as = "GRanges")
CnT_H3K27ac <- CnT_H3K27ac[seqnames(CnT_H3K27ac) == "chr1"]
my_label <- c("name","score","strand","signalValue","pValue","qValue","peak")
colnames(GenomicRanges::mcols(CnT_H3K27ac)) <- my_label
usethis::use_data(CnT_H3K27ac)

CnT_H3K27ac_picard

Example Picard duplication metrics file

Description

Duplication metrics output of CUT&Tag H3K27ac file (sample accession: SRR8581604). Raw sequences were aligned to hg19 genome and Picard was performed by Leyla Abbasova. The duplication summary output generated by Picard was processed to reduce the size of data.

Usage

data("CnT_H3K27ac_picard")
compute_consensus_peaks

Description

Compute consensus peaks from a list of GRanges.

Usage

```r
compute_consensus_peaks(
  grlist,
  groups = NULL,
  genome_build,
  lower = 2,
  upper = Inf,
  min.gapwidth = 1L,
  method = c("granges", "consensusseeker"),
  ...
)
```

Arguments

- **grlist**: Named list of GRanges objects.
- **groups**: A character vector of the same length as grlist defining how to group GRanges objects when computing consensus peaks.
- **genome_build**: Genome build name.
- **lower**, **upper**: The lower and upper bounds for the slice.
- **min.gapwidth**: Ranges separated by a gap of at least min.gapwidth positions are not merged.
- **method**: Method to call peaks with:
  - "granges": Simple overlap procedure using GRanges functions. Faster but less accurate.
  - "consensusseeker": Uses findConsensusPeakRegions to compute consensus peaks. Slower but more accurate.

Source

The code to prepare the .Rda file is:

```r
picard <- read.table("path/to/picard/duplication/output",header = TRUE, fill = TRUE)
CnT_H3K27ac_picard <- picard[1,]
usethis::use_data(CnT_H3K27ac_picard, overwrite = TRUE)
```
Arguments passed on to `consensusSeeker::findConsensusPeakRegions`

- **narrowPeaks**: a GRanges containing called peak regions of signal enrichment based on pooled, normalized data for all analyzed experiments. All GRanges entries must have a metadata field called "name" which identifies the region to the called peak. All GRanges entries must also have a row name which identifies the experiment of origin. Each peaks entry must have an associated narrowPeaks entry. A GRanges entry is associated to a narrowPeaks entry by having an identical metadata "name" field and a identical row name.

- **peaks**: a GRanges containing called peaks of signal enrichment based on pooled, normalized data for all analyzed experiments. All GRanges entries must have a metadata field called "name" which identifies the called peak. All GRanges entries must have a row name which identifies the experiment of origin. Each peaks entry must have an associated narrowPeaks entry. A GRanges entry is associated to a narrowPeaks entry by having a identical metadata "name" field and a identical row name.

- **chrInfo**: a SeqInfo containing the name and the length of the chromosomes to analyze. Only the chromosomes contained in this SeqInfo will be analyzed.

- **extendingSize**: a numeric value indicating the size of padding on both sides of the position of the peaks median to create the consensus region. The minimum size of the consensus region is equal to twice the value of the extendingSize parameter. The size of the extendingSize must be a positive integer. Default = 250.

- **expandToFitPeakRegion**: a logical indicating if the region size, which is set by the extendingSize parameter is extended to include the entire narrow peak regions of all peaks included in the unextended consensus region. The narrow peak regions of the peaks added because of the extension are not considered for the extension. Default: FALSE.

- **shrinkToFitPeakRegion**: a logical indicating if the region size, which is set by the extendingSize parameter is shrinked to fit the narrow peak regions of the peaks when all those regions are smaller than the consensus region. Default: FALSE.

- **minNbrExp**: a positive numeric or a positive integer indicating the minimum number of experiments in which at least one peak must be present for a potential consensus region. The numeric must be a positive integer inferior or equal to the number of experiments present in the narrowPeaks and peaks parameters. Default = 1.

- **nbrThreads**: a numeric or a integer indicating the number of threads to use in parallel. The nbrThreads must be a positive integer. Default = 1.

### Details

_Note:_ If you get the error "Error in serialize(data, node$con) : error writing to connection", try running `closeAllConnections` and rerun `compute_consensus_peaks`. This error can sometimes occur when `compute_consensus_peaks` has been disrupted partway through.

### Value

Named list of consensus peak GRanges.
**Source**

*GenomicRanges tutorial*

*consensusSeeker*

**Examples**

```r
data("encode_H3K27ac") # example dataset as GRanges object
data("Cnt_H3K27ac") # example dataset as GRanges object
data("CnR_H3K27ac") # example dataset as GRanges object
grlist <- list(CnR=CnR_H3K27ac, CnT=CnT_H3K27ac, ENCODE=encode_H3K27ac)

consensus_peaks <- compute_consensus_peaks(grlist = grlist,
                                           groups = c("Imperial",
                                                      "Imperial",
                                                      "ENCODE"))
```

---

**compute_corr**

*Compute correlation matrix*

**Description**

Compute correlation matrix on all peak files.

**Usage**

```r
compute_corr(
  peakfiles,
  reference = NULL,
  genome_build,
  keep_chr = NULL,
  drop_empty_chr = FALSE,
  bin_size = 5000,
  method = "spearman",
  intensity_cols = c("total_signal", "qValue", "Peak Score", "score"),
  return_bins = FALSE,
  fill_diag = NA,
  workers = check_workers(),
  save_path = tempfile(fileext = ".corr.csv.gz")
)
```

**Arguments**

peakfiles A list of peak files as GRanges object and/or as paths to BED files. If paths are provided, EpiCompare imports the file as GRanges object. EpiCompare also accepts a list containing a mix of GRanges objects and paths.Files must be listed and named using list(). E.g. list("name1"=file1,"name2"=file2). If no names are specified, default file names will be assigned.
compute_corr

reference A named list containing reference peak file(s) as GRanges object. Please ensure that the reference file is listed and named i.e. list("reference_name" = reference_peak). If more than one reference is specified, individual reports for each reference will be generated. However, please note that specifying more than one reference can take awhile. If a reference is specified, it enables two analyses: (1) plot showing statistical significance of overlapping/non-overlapping peaks; and (2) ChromHMM of overlapping/non-overlapping peaks.

genome_build The build of **all** peak and reference files to calculate the correlation matrix on. If all peak and reference files are not of the same build use liftover_grlist to convert them all before running. Genome build should be one of hg19, hg38, mm9, mm10.

keep_chr Which chromosomes to keep.

drop_empty_chr Drop chromosomes that are not present in any of the peakfiles (default: FALSE).

bin_size Default of 100. Base-pair size of the bins created to measure correlation. Use smaller value for higher resolution but longer run time and larger memory usage.

method Default spearman (i.e. non-parametric). A character string indicating which correlation coefficient (or covariance) is to be computed. One of "pearson", "kendall", or "spearman": can be abbreviated.

intensity_cols Depending on which columns are present, this value will be used to get quantiles and ultimately calculate the correlations:

- "total_signal": Used by the peak calling software SEACR. **NOTE**: Another SEACR column (e.g. "max_signal") can be used together or instead of "total_signal".
- "qValue" Used by the peak calling software MACS2/3. Should contain the negative log of the p-values after multiple testing correction.
- "Peak Score" : Used by the peak calling software HOMER.

return_bins If TRUE, returns a named list with both the rebinned (standardised) peaks ("bin") and the correlation matrix ("cor"). If FALSE (default), returns only the correlation matrix (unlisted).

fill_diag Fill the diagonal of the overlap matrix.

workers Number of threads to parallelize across.

save_path Path to save a table of correlation results to.

Value
correlation matrix

Examples
data("CnR_H3K27ac")
data("CnT_H3K27ac")
data("encode_H3K27ac")
peakfiles <- list(CnR_H3K27ac=CnR_H3K27ac, CnT_H3K27ac=CnT_H3K27ac)
reference <- list("encode_H3K27ac"=encode_H3K27ac)

#increasing bin_size for speed but lower values will give more granular corr
corr_mat <- compute_corr(peakfiles = peakfiles, 
reference = reference, 
genome_build = "hg19", 
bin_size = 200000, 
workers = 1)

download_button

Description

Save an object as RDS and create a download button that can be rendered to Rmarkdown HTML pages. Uses the package downloadthis.

Usage

download_button(
  object, 
  save_output = FALSE, 
  outfile_dir = NULL, 
  filename = NULL, 
  button_label = paste0("Download: ", "<code>", filename, "</code>"), 
  output_extension = ".rds", 
  icon = "fa fa-save", 
  button_type = "success", 
  self_contained = TRUE, 
  add_download_button = TRUE, 
  verbose = TRUE
)

Arguments

object R object to serialize.

save_output Default FALSE. If TRUE, all outputs (tables and plots) of the analysis will be saved in a folder (EpiCompare_file).

outfile_dir Directory to save the file to.

filename Name of the file to save.

button_label Character (HTML), button label

output_extension Extension of the output file. Currently, .csv, .xlsx, and .rds are supported. If a (named) list is passed to the function, only .xlsx and .rds are supported.

icon Fontawesome tag e.g.: "fa fa-save"

button_type Character, one of the standard Bootstrap types

self_contained A boolean to specify whether your HTML output is self-contained. Default to FALSE.

add_download_button Add download buttons for each plot or dataset.

verbose Print messages.
**Value**

Download button as HTML text.

**Source**

- csv2 Issue.
- Plotly Issue

**Examples**

```r
button <- download_button(object=mtcars)
```

---

<table>
<thead>
<tr>
<th>encode_H3K27ac</th>
<th>Example ChIP-seq peak file</th>
</tr>
</thead>
</table>

**Description**

Human H3K27ac peak file generated with ChIP-seq using K562 cell-line. Human genome build hg19 was used. The peak file (.BED) was obtained from ENCODE project ([https://www.encodeproject.org/files/ENCFF044JNJ/](https://www.encodeproject.org/files/ENCFF044JNJ/)). The BED file was then imported as a GRanges object. Peaks located on chromosome 1 were subsetted to reduce the dataset size.

**Usage**

```r
data("encode_H3K27ac")
```

**Format**

An object of class GRanges of length 5142.

**Source**

The code to prepare the .Rda file from the raw peak file is:

```r
# dataset was directly downloaded from
# https://www.encodeproject.org/files/ENCFF044JNJ/ encode_H3K27ac <- ChIPseeker::readPeakFile("path", as = "GRanges")
encode_H3K27ac <- encode_H3K27ac[seqnames(encode_H3K27ac) == "chr1"]
my_label <- c("name","score","strand","signalValue","pValue","qValue","peak")
colnames(GenomicRanges::mcols(encode_H3K27ac)) <- my_label
usethis::use_data(encode_H3K27ac, overwrite = TRUE)
```
EpiCompare

Compare epigenomic datasets

Description

This function compares and analyses multiple epigenomic datasets and outputs an HTML report containing all results of the analysis. The report is mainly divided into three sections: (1) General Metrics on Peakfiles, (2) Peak Overlaps and (3) Functional Annotation of Peaks.

Usage

EpiCompare(
  peakfiles,
  genome_build,
  genome_build_output = "hg19",
  blacklist = NULL,
  picard_files = NULL,
  reference = NULL,
  upset_plot = FALSE,
  stat_plot = FALSE,
  chromHMM_plot = FALSE,
  chromHMM_annotation = "K562",
  chipseeker_plot = FALSE,
  enrichment_plot = FALSE,
  tss_plot = FALSE,
  tss_distance = c(-3000, 3000),
  precision_recall_plot = FALSE,
  n_threshold = 20,
  corr_plot = FALSE,
  bin_size = 5000,
  interact = TRUE,
  add_download_button = FALSE,
  save_output = FALSE,
  output_filename = "EpiCompare",
  output_timestamp = FALSE,
  output_dir,
  display = NULL,
  run_all = FALSE,
  workers = 1,
  quiet = FALSE,
  error = FALSE,
  debug = FALSE
)

Arguments

peakfiles A list of peak files as GRanges object and/or as paths to BED files. If paths are provided, EpiCompare imports the file as GRanges object. EpiCompare also
accepts a list containing a mix of GRanges objects and paths. Files must be listed and named using `list()`. E.g. `list("name1"=file1, "name2"=file2)`. If no names are specified, default file names will be assigned.

**genome_build**

A named list indicating the human genome build used to generate each of the following inputs:

- "peakfiles": Genome build for the `peakfiles` input. Assumes genome build is the same for each element in the `peakfiles` list.
- "reference": Genome build for the `reference` input.
- "blacklist": Genome build for the `blacklist` input.

Example input list:

```r
genome_build = list(peakfiles="hg38", reference="hg19", blacklist="hg19")
```

Alternatively, you can supply a single character string instead of a list. This should only be done in situations where all three inputs (peakfiles, reference, blacklist) are of the same genome build. For example:

```r
genome_build = "hg19"
```

**genome_build_output**

Genome build to standardise all inputs to. Liftovers will be performed automatically as needed. Default: "hg19".

**blacklist**

A GRanges object containing blacklisted genomic regions. Blacklists included in EpiCompare are:

- NULL (default): Automatically selects the appropriate blacklist based on the `genome_build_output` argument.
- "hg19_blacklist": Regions of hg19 genome that have anomalous and/or unstructured signals. `hg19_blacklist`
- "hg38_blacklist": Regions of hg38 genome that have anomalous and/or unstructured signals. `hg38_blacklist`
- "mm10_blacklist": Regions of mm10 genome that have anomalous and/or unstructured signals. `mm10_blacklist`
- "mm9_blacklist": Blacklisted regions of mm10 genome that have been lifted over from `mm10_blacklist`. `mm9_blacklist`
- <user_input>: A custom user-provided blacklist in GRanges format.

**picard_files**

A list of summary metrics output from Picard. Files must be in data.frame format and listed using `list()` and named using `names()`. To import Picard duplication metrics (.txt file) into R as data frame, use:

```r
picard <- read.table("/path/to/picard/output", header = TRUE, fill = TRUE).
```

**reference**

A named list containing reference peak file(s) as GRanges object. Please ensure that the reference file is listed and named i.e. `list("reference_name" = reference_peak)`. If more than one reference is specified, individual reports for each reference will be generated. However, please note that specifying more than one reference can take awhile. If a reference is specified, it enables two analyses: (1) plot showing statistical significance of overlapping/non-overlapping peaks; and (2) ChromHMM of overlapping/non-overlapping peaks.

**upset_plot**

Default FALSE. If TRUE, the report includes upset plot of overlapping peaks.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_plot</td>
<td>FALSE</td>
<td>If TRUE, the function creates a plot showing the statistical significance of overlapping/non-overlapping peaks. Reference peak file must be provided.</td>
</tr>
<tr>
<td>chromHMM_plot</td>
<td>FALSE</td>
<td>If TRUE, the function outputs ChromHMM heatmap of individual peak files. If a reference peak file is provided, ChromHMM annotation of overlapping and non-overlapping peaks is also provided.</td>
</tr>
<tr>
<td>chromHMM_annotation</td>
<td></td>
<td>ChromHMM annotation for ChromHMM plots. Default K562 cell-line. Cell-line options are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &quot;K562&quot; = K-562 cells</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &quot;Gm12878&quot; = Cellosaurus cell-line GM12878</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &quot;H1hesc&quot; = H1 Human Embryonic Stem Cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &quot;Hepg2&quot; = Hep G2 cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &quot;Hmec&quot; = Human Mammary Epithelial Cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &quot;Hsmm&quot; = Human Skeletal Muscle Myoblasts</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &quot;Huvec&quot; = Human Umbilical Vein Endothelial Cells</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &quot;Nhek&quot; = Normal Human Epidermal Keratinocytes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &quot;Nhlf&quot; = Normal Human Lung Fibroblasts</td>
</tr>
<tr>
<td>chipseeker_plot</td>
<td>FALSE</td>
<td>If TRUE, the report includes a barplot of ChIPseeker annotation of peak files.</td>
</tr>
<tr>
<td>enrichment_plot</td>
<td>FALSE</td>
<td>If TRUE, the report includes dotplots of KEGG and GO enrichment analysis of peak files.</td>
</tr>
<tr>
<td>tss_plot</td>
<td>FALSE</td>
<td>If TRUE, the report includes peak count frequency around transcriptional start site. Note that this can take awhile.</td>
</tr>
<tr>
<td>tss_distance</td>
<td></td>
<td>A vector specifying the distance upstream and downstream around transcription start sites (TSS). The default value is c(-3000, 3000); meaning peak frequency 3000bp upstream and downstream of TSS will be displayed.</td>
</tr>
<tr>
<td>precision_recall_plot</td>
<td>FALSE</td>
<td>If TRUE, creates a precision-recall curve plot and an F1 plot using plot_precision_recall.</td>
</tr>
<tr>
<td>n_threshold</td>
<td></td>
<td>Number of thresholds to test.</td>
</tr>
<tr>
<td>corr_plot</td>
<td>FALSE</td>
<td>If TRUE, creates a correlation plot across all peak files using plot_corr.</td>
</tr>
<tr>
<td>bin_size</td>
<td>100</td>
<td>Default of 100. Base-pair size of the bins created to measure correlation. Use smaller value for higher resolution but longer run time and larger memory usage.</td>
</tr>
<tr>
<td>interact</td>
<td>TRUE</td>
<td>Default TRUE. By default, plots are interactive. If set FALSE, all plots in the report will be static.</td>
</tr>
<tr>
<td>add_download_button</td>
<td></td>
<td>Add download buttons for each plot or dataset.</td>
</tr>
<tr>
<td>save_output</td>
<td>FALSE</td>
<td>Default FALSE. If TRUE, all outputs (tables and plots) of the analysis will be saved in a folder (EpiCompare_file).</td>
</tr>
</tbody>
</table>
output_filename

Default EpiCompare.html. If otherwise, the html report will be saved in the specified name.

output_timestamp

Default FALSE. If TRUE, date will be included in the file name.

output_dir

Path to where output HTML file should be saved.

display

After completion, automatically display the HTML report file in one of the following ways:

- "browser": Display the report in your default web browser.
- "rstudio": Display the report in Rstudio.
- NULL (default): Do not display the report.

run_all

Convenience argument that enables all plots/features (without specifying each argument manually) by overriding the default values. Default: FALSE.

workers

Number of threads to parallelize across.

quiet

An option to suppress printing during rendering from knitr, pandoc command line and others. To only suppress printing of the last "Output created: " message, you can set rmarkdown.render.message to FALSE.

error

If TRUE, the Rmarkdown report will continue to render even when some chunks encounter errors (default: FALSE). Passed to opts_chunk.

debug

Run in debug mode, where are messages and warnings are printed within the HTML report (default: FALSE).

Value

Path to one or more HTML report files.

Examples

### Load Data ###
data("encode_H3K27ac") # example dataset as GRanges object
data("CnT_H3K27ac") # example dataset as GRanges object
data("CnR_H3K27ac") # example dataset as GRanges object
data("CnT_H3K27ac_picard") # example Picard summary output
data("CnR_H3K27ac_picard") # example Picard summary output

#### Prepare Input ####

# create named list of peakfiles
peakfiles <- list(CnR=CnR_H3K27ac, CnT=CnT_H3K27ac)
# create named list of picard outputs
picard_files <- list(CnR=CnR_H3K27ac_picard, CnT=CnT_H3K27ac_picard)
# reference peak file
reference <- list("ENCODE" = encode_H3K27ac)

### Run EpiCompare ###

output_html <- EpiCompare(peakfiles = peakfiles,
genome_build = list(peakfiles="hg19",
reference="hg19"),
icard_files = picard_files,
fragment_info

Description
This function outputs a summary on fragments using metrics generated by Picard. Provides the number of mapped fragments, duplication rate and number of unique fragments.

Usage
fragment_info(picard_list)

Arguments
picard_list Named list of duplication metrics generated by Picard as data frame. Data frames must be named and listed using list(). e.g. list("name1"=file1, "name2"=file2). To import Picard duplication metrics (.txt file) into R as data frame, use picard <- read.table("/path/to/picard/output", header = TRUE, fill = TRUE).
Value

A table summarizing metrics on fragments.

Examples

```r
### Load Data ###
data(CnT_H3K27ac_picard) # example picard output
data(CnR_H3K27ac_picard) # example picard output

### Import Picard Metrics ###
# To import Picard duplication metrics (.txt file) into R as data frame
# CnT_H3K27ac_picard <- read.table("/path/to/picard/output.txt",
#   header = TRUE,fill = TRUE)

### Create Named List ###
picard_list <- list("CnT_H3K27ac"=CnT_H3K27ac_picard,
                    "CnR_H3K27ac"=CnR_H3K27ac_picard)
df <- fragment_info(picard_list = picard_list)
```

---

gather_files  
Gather files

Description

Recursively find peak/picard files stored within subdirectories and import them as a list of GRanges objects.

Usage

```r
gather_files(
  dir,
  type = "peaks.stringent",
  nfcore_cutandrun = FALSE,
  return_paths = FALSE,
  rbind_list = FALSE,
  workers = check_workers(),
  verbose = TRUE
)
```

Arguments

dir  Directory to search within.
type  File type to search for. Options include:

- "<pattern>" Finds files matching an arbitrary regex pattern specified by user.
- "peaks.stringent" Finds files ending in ".stringent.bed$"
- "peaks.consensus" Finds files ending in ".consensus.peaks.bed$"
- "peaks.consensus.filtered" Finds files ending in ".consensus.peaks.filtered.awk.bed$"
- "picard" Finds files ending in ".target.markdup.MarkDuplicates.metrics.txt$"
gather_files_names

nfcore_cutandrun

Whether the files were generated by the nf-core/cutandrun Nextflow pipeline. If TRUE, can use the standardised folder structure to automatically generate more descriptive file names with sample IDs.

return_paths

Return only the file paths without actually reading them in as GRanges.

rbind_list

Bind all objects into one.

workers

Number of cores to parallelise across (in applicable functions). If NULL, will set to the total number of available cores minus 1.

verbose

Print messages.

Details

For "peaks.stringent" files called with SEACR, column names will be automatically added:

- total_signal : Total signal contained within denoted coordinates.
- max_signal : Maximum bedgraph signal attained at any base pair within denoted coordinates.
- max_signal_region : Region representing the farthest upstream and farthest downstream bases within the denoted coordinates that are represented by the maximum bedgraph signal.

Value

A named list of GRanges objects.

Examples

```r
### Make example files ###
save_paths <- EpiCompare::write_example_peaks()
dir <- unique(dirname(save_paths))
### Gather/import files ###
peaks <- EpiCompare::gather_files(dir=dir,
                                   type="peaks.narrow",
                                   workers = 1)
```

Description

Support function for gather_files.

Usage

gather_files_names(paths, type, nfcore_cutandrun, verbose = TRUE)
get_bpparam

Arguments

paths  Character vector of file paths.
type   File type to search for. Options include:
  • "<pattern>" Finds files matching an arbitrary regex pattern specified by user.
  • "peaks.stringent" Finds files ending in "*.stringent.bed$"
  • "peaks.consensus" Finds files ending in "*.consensus.peaks.bed$"
  • "peaks.consensus.filtered" Finds files ending in "*.consensus.peaks.filtered.awk.bed$"
  • "picard" Finds files ending in "*.target.markdup.MarkDuplicates.metrics.txt$"

nfcore_cutandrun  Whether the files were generated by the nf-core/cutandrun Nextflow pipeline. If TRUE, can use the standardised folder structure to automatically generate more descriptive file names with sample IDs.

verbose  Print messages.

Value

Named character vector.

get_bpparam  Get BiocParallel parameters

Description

Get (and optionally register) BiocParallel parameter (BPPARAM). SnowParam is the default function as it tends to be more robust. However, because it doesn’t work on Windows, this function automatically detected the Operating System and switches to SerialParam as needed.

Usage

get_bpparam(
  workers,
  progressbar = workers > 1,
  use_snowparam = TRUE,
  register_now = FALSE
)

Arguments

workers  Number of threads to parallelize across.
progressbar logical(1) Enable progress bar (based on plyr:::progress_text).
use_snowparam  Whether to use SnowParam (default: TRUE) or MulticoreParam (FALSE) when parallelising across multiple workers.
register_now  Register the cores now with register (TRUE), or simply return the BPPARAM object (default: FALSE).
get_chromHMM_annotation

Description

Download ChromHMM annotation file(s) for a given cell-line (returned as a GRanges object) or a list of cell-lines (returned as a named list of GRanges objects). All annotations are aligned to the hg19 genome build. All data can be found on the UCSC Genome Browser here.

Usage

```r
get_chromHMM_annotation(
  cell_line,
  cache = BiocFileCache::BiocFileCache(ask = FALSE)
)
```

Arguments

- `cell_line` 
  ChromHMM annotation for user-specified cell-line. Cell-line options are:
  - "K562" = K-562 cells
  - "Gm12878" = Cellosaurus cell-line GM12878
  - "H1hesc" = H1 Human Embryonic Stem Cell
  - "Hepg2" = Hep G2 cell
  - "Hmec" = Human Mammary Epithelial Cell
  - "Hsmm" = Human Skeletal Muscle Myoblasts
  - "Huvec" = Human Umbilical Vein Endothelial Cells
  - "Nhek" = Normal Human Epidermal Keratinocytes
  - "Nhlf" = Normal Human Lung Fibroblasts

Value

group_files  

**Description**

Assign group names to each file in a named list based on a series of string searches based on combinations of relevant metadata factors.

**Usage**

`group_files(peakfiles, searches)`

**Arguments**

- **peakfiles** A list of peak files as GRanges object and/or as paths to BED files. If paths are provided, EpiCompare imports the file as GRanges object. EpiCompare also accepts a list containing a mix of GRanges objects and paths. Files must be listed and named using `list()`. E.g. `list("name1"=file1,"name2"=file2)`. If no names are specified, default file names will be assigned.

- **searches** A named list of substrings to group `peakfiles` by.

**Value**

Named peak files

**Examples**

data("encode_H3K27ac") # example dataset as GRanges object
data("CnT_H3K27ac") # example dataset as GRanges object
data("CnR_H3K27ac") # example dataset as GRanges object
peakfiles <- list(CnR_H3K27ac=CnR_H3K27ac, 
                  CnT_H3K27ac=CnT_H3K27ac, 
                  encode_H3K27ac=encode_H3K27ac)

peaks_grouped <- group_files(peakfiles = peakfiles, 
                             searches=list(assay=c("H3K27ac"), 
                                           source=c("Cn","ENCORE")))

hg19_blacklist  

**Description**

Obtained from https://www.encodeproject.org/files/ENCFF001TDO/. The ENCODE blacklist includes regions of the hg19 genome that have anomalous and/or unstructured signals independent of the cell-line or experiment. Removal of ENCODE blacklist is recommended for quality measure.
Usage

data("hg19_blacklist")

Format

An object of class GRanges of length 411.

Source

The code to prepare the .Rda file is:

```r
# blacklisted regions were directly downloaded
# from https://www.encodeproject.org/files/ENCFF001TDO/
hg19_blacklist <- ChIPseeker::readPeakFile(file.path(path), as = "GRanges")
usethis::use_data(hg19_blacklist, overwrite = TRUE)
```

---

hg38_blacklist  Human genome hg38 blacklisted regions

Description

Obtained from https://www.encodeproject.org/files/ENCFF356LFX/. The ENCODE blacklist includes regions of the hg38 genome that have anomalous and/or unstructured signals independent of the cell-line or experiment. Removal of ENCODE blacklist is recommended for quality measure.

Usage

data("hg38_blacklist")

Format

An object of class GRanges of length 910.

Source

The code to prepare the .Rda file is:

```r
## blacklisted regions were directly downloaded
## from https://www.encodeproject.org/files/ENCFF356LFX/
hg38_blacklist <- ChIPseeker::readPeakFile(file.path(path), as = "GRanges")
usethis::use_data(hg38_blacklist, overwrite = TRUE)
```
is_granges

Is an object of class GRanges

Description

Check whether an object is of the class GRanges.

Usage

is_granges(obj)

Arguments

obj Any R object.

Value

Boolean.

liftover_grlist

Liftover peak list

Description

Perform genome build liftover to one or more GRanges objects at once.

Usage

liftover_grlist(
  grlist,
  input_build,
  output_build = "hg19",
  style = "UCSC",
  keep_chr = paste0("chr", c(seq_len(22), "X", "Y")),
  as_grangeslist = FALSE,
  merge_all = FALSE,
  verbose = TRUE
)
**Arguments**

- **grlist**: A named list of `GRanges` objects, or simply a single unlisted `GRanges` object. Can perform liftover within species or across species.
- **input_build**: The genome build of `grlist`.
- **output_build**: Desired genome build for `grlist` to be lifted over to.
- **style**: Chromosome style, set by `seqlevelsStyle`.
  - "UCSC": Uses the chromosome style "chr1".
  - "NCBI": Uses the chromosome style "1"
- **keep_chr**: Which chromosomes to keep.
- **as_grangeslist**: Return as a `GRangesList`.
- **merge_all**: Merge all `GRanges` into a single `GRanges` object.
- **verbose**: Print messages.

**Value**

Named list of lifted `GRanges` objects.

**Examples**

```r
grlist <- list("gr1"=GenomicRanges::GRanges("4:1-100000"),
               "gr2"=GenomicRanges::GRanges("6:1-100000"),
               "gr3"=GenomicRanges::GRanges("8:1-100000"))

grlist_lifted <- liftover_grlist(grlist = grlist,
                                 input_build = "hg19",
                                 output_build="hg38")
```

**messager** *Print messages*

**Description**

Conditionally print messages. Allows developers to easily control verbosity of functions, and meet Bioconductor requirements that dictate the message must first be stored to a variable before passing to `message`.

**Usage**

```r
messager(..., v = TRUE, parallel = FALSE)
```

**Arguments**

- **v**: Whether to print messages or not.
- **parallel**: Whether to enable message print when wrapped in parallelised functions.
**message_parallel**

**Value**
Null

**Description**
Send messages to console even from within parallel processes

**Usage**
message_parallel(...)

**Value**
A message

---

**mm10_blacklist**

**Mouse genome mm10 blacklisted regions**

**Description**
Obtained from [https://www.encodeproject.org/files/ENCFF547MET/](https://www.encodeproject.org/files/ENCFF547MET/). The ENCODE blacklist includes regions of the mm10 genome that have anomalous and/or unstructured signals independent of the cell-line or experiment. Removal of ENCODE blacklist is recommended for quality measure.

**Usage**
data("mm10_blacklist")

**Format**
An object of class GRanges of length 164.

**Source**
The code to prepare the .Rda file is:
```r
## blacklisted regions were directly downloaded
## from https://www.encodeproject.org/files/ENCFF547MET/
mm10_blacklist <- ChIPseeker::readPeakFile(file.path(path), as = "GRanges")
usethis::use_data(mm10_blacklist, overwrite = TRUE)
```
**mm9_blacklist**

*Mouse genome mm9 blacklisted regions*

**Description**

Blacklisted regions of the mm9 genome build obtained by lifting over the mm10_blacklist.

**Usage**

```r
data("mm9_blacklist")
```

**Format**

An object of class GRanges of length 292.

**Source**

```r
tmp <- base::get("mm10_blacklist", asNamespace("EpiCompare")) mm9_blacklist <- liftover_grlist(grlist = tmp, input_build = "mm10", output_build = "mm9", keep_chr = NULL) usethis::use_data(mm9_blacklist, overwrite = TRUE)
```

---

**overlap_heatmap**

*Generate heatmap of percentage overlap*

**Description**

This function generates a heatmap showing percentage of overlapping peaks between peak files.

**Usage**

```r
overlap_heatmap(
  peaklist,
  interact = TRUE,
  draw_cellnote = TRUE,
  fill_diag = NA,
  verbose = TRUE
)
```

**Arguments**

- **peaklist**
  A list of peak files as GRanges object. Files must be listed and named using `list()`. e.g. `list("name1"=file1, "name2"=file2)`. If not named, default file names will be assigned.

- **interact**
  Default TRUE. By default heatmap is interactive. If FALSE, heatmap is static.

- **draw_cellnote**
  Draw the numeric values within each heatmap cell.

- **fill_diag**
  Fill the diagonal of the overlap matrix.

- **verbose**
  Print messages.
### Examples

```r
### Load Data ###
data("encode_H3K27ac") # example peak file GRanges object
data("CnT_H3K27ac") # example peak file GRanges object
### Create Named List ###
peaklist <- list("encode"=encode_H3K27ac, "CnT"=CnT_H3K27ac)
### Run ###
my_heatmap <- overlap_heatmap(peaklist = peaklist)
```

### Description

This function calculates the percentage of overlapping peaks and outputs a table or matrix of results.

### Usage

```r
overlap_percent(
  peaklist1,  # A list of peak files as GRanges object. Files must be listed and named using list(). e.g. list("name1"=file1, "name2"=file2). If not named, default file names will be assigned.
  peaklist2,  # peaklist1 A list of peak files as GRanges object. Files must be listed and named using list(). e.g. list("name1"=file1, "name2"=file2).
  invert = FALSE,  # If TRUE, keep only the ranges in x that do not overlap ranges.
  precision_recall = TRUE,  # Return precision-recall results for all combinations of peaklist1 (the "query") and peaklist2 (the "subject"). See subsetByOverlaps for more details on this terminology.
  suppress_messages = TRUE)  # Suppress messages.
```

### Arguments

- **peaklist1**: A list of peak files as GRanges object. Files must be listed and named using `list()`. e.g. `list("name1"=file1, "name2"=file2)`. If not named, default file names will be assigned.
- **peaklist2**: A list of peak files as GRanges object. Files must be listed and named using `list()`. e.g. `list("name1"=file1, "name2"=file2)`. 
- **invert**: If TRUE, keep only the ranges in x that do not overlap ranges.
- **precision_recall**: Return precision-recall results for all combinations of peaklist1 (the "query") and peaklist2 (the "subject"). See `subsetByOverlaps` for more details on this terminology.
- **suppress_messages**: Suppress messages.

### Value

data frame
Examples

```r
### Load Data ###
data("encode_H3K27ac") # example peakfile GRanges object
data("CnT_H3K27ac") # example peakfile GRanges object
data("CnR_H3K27ac") # example peakfile GRanges object

### Create Named Peaklist ###
peaks <- list("CnT"=CnT_H3K27ac, "CnR"=CnR_H3K27ac)
reference_peak <- list("ENCODE"=encode_H3K27ac)

### Run ###
overlap <- overlap_percent(peaklist1=peaks,
                           peaklist2=reference_peak)
```

**Description**

This function calculates the statistical significance of overlapping/ non-overlapping peaks against a reference peak file. If the reference peak file has the BED6+4 format (peak called by MACS2), the function generates a series of box plots showing the distribution of q-values for sample peaks that are overlapping and non-overlapping with the reference. If the reference peak file does not have the BED6+4 format, the function uses enrichPeakOverlap from ChIPseeker package to calculate the statistical significance of overlapping peaks only. In this case, please provide an annotation file as a TxDb object.

**Usage**

```r
overlap_stat_plot(
  reference,
  peaklist,
  txdb = NULL,
  interact = FALSE,
  nShuffle = 50,
  digits = 4,
  workers = check_workers()
)
```

**Arguments**

- `reference` A reference peak file as GRanges object.
- `peaklist` A list of peak files as GRanges object. Files must be listed and named using `list()`. E.g. `list("name1"=file1, "name2"=file2)`. If not named, default file names will be assigned.
- `txdb` A TxDb annotation object from Bioconductor. This is required only if the reference file does not have BED6+4 format.
**overlap_upset_plot**

Interpret

Default TRUE. By default, plots are interactive. If set FALSE, all plots in the report will be static.

**nShuffle**

shuffle numbers

**digits**

integer indicating the number of decimal places (round) or significant digits (signif) to be used. For round, negative values are allowed (see ‘Details’).

**workers**

Number of threads to parallelize across.

**Value**

A named list.

- "plot" boxplot/barplot showing the statistical significance of overlapping/non-overlapping peaks.
- "data" Plot data.

**Examples**

```r
### Load Data ###
data("encode_H3K27ac") # example peakfile GRanges object
data("CnT_H3K27ac") # example peakfile GRanges object
data("CnR_H3K27ac") # example peakfile GRanges object
### Create Named Peaklist & Reference ###
peaklist <- list("CnT"=CnT_H3K27ac, "CnR"=CnR_H3K27ac)
reference <- list("ENCODE"=encode_H3K27ac)
out <- overlap_stat_plot(reference = reference,
                         peaklist = peaklist,
                         workers = 1)
```

**Description**

This function generates upset plot of overlapping peaks files using the `ComplexUpset` package.

**Usage**

```r
overlap_upset_plot(peaklist, verbose = TRUE)
```

**Arguments**

- `peaklist` A named list of peak files as GRanges object. Objects must be listed and named using `list()`. e.g. `list("name1"=file1, "name2"=file2)`. If not named, default file names are assigned.
- `verbose` Print messages

**Value**

Upset plot of overlapping peaks.
Examples

### Load Data ###
data("encode_H3K27ac") # load example data
data("CnT_H3K27ac") # load example data
peaklist <- list("encode"=encode_H3K27ac, "CnT"=CnT_H3K27ac)
my_plot <- overlap_upset_plot(peaklist = peaklist)

---

peak_info

**Summary of Peak Information**

Description

This function outputs a table summarizing information on the peak files. Provides the total number of peaks and the percentage of peaks in blacklisted regions.

Usage

peak_info(peaklist, blacklist)

Arguments

- peaklist: A named list of peak files as GRanges object. Objects listed using `list("name1" = peak, "name2" = peak2).
- blacklist: A GRanges object containing blacklisted regions.

Value

A summary table of peak information

Examples

### Load Data ###
data("encode_H3K27ac") # example peakfile GRanges object
data("CnT_H3K27ac") # example peakfile GRanges object
data("hg19_blacklist") # example blacklist GRanges object

### Named Peaklist ###
peaklist <- list("encode"=encode_H3K27ac, "CnT"=CnT_H3K27ac)

### Run ###
df <- peak_info(peaklist = peaklist,
               blacklist = hg19_blacklist)
plot_ChIPseeker_annotation

Create ChIPseeker annotation plot

Description

This function annotates peaks using ChIPseeker::annotatePeak. It outputs functional annotation of each peak file in a barplot.

Usage

plot_ChIPseeker_annotation(
  peaklist,
  txdb = NULL,
  tss_distance = c(-3000, 3000),
  interact = FALSE
)

Arguments

peaklist A list of peak files as GRanges object. Files must be listed and named using list(). e.g. list("name1"=file1, "name2"=file2). If not named, default file names will be assigned.

txdb A TxDb annotation object from Bioconductor.

tss_distance A vector specifying the distance upstream and downstream around transcription start sites (TSS). The default value is c(-3000, 3000); meaning peak frequency 3000bp upstream and downstream of TSS will be displayed.

interact Default TRUE. By default, plots are interactive. If set FALSE, all plots in the report will be static.

Value

ggplot barplot

Examples

### Load Data ###
data("CnT_H3K27ac") # example peakfile GRanges object
data("CnR_H3K27ac") # example peakfile GRanges object
peaklist <- list("CnT"=CnT_H3K27ac, "CnR"=CnR_H3K27ac)
my_plot <- plot_ChIPseeker_annotation(peaklist = peaklist,
tss_distance = c(-50, 50))
plot_chromHMM

Plot ChromHMM heatmap

Description

Creates a heatmap using outputs from ChromHMM using ggplot2. The function takes a list of peak-files, performs ChromHMM and outputs a heatmap. ChromHMM annotation file must be loaded prior to using this function. ChromHMM annotations are aligned to hg19, and will be automatically lifted over to the genome_build to match the build of the peaklist.

Usage

plot_chromHMM(
  peaklist,
  chromHMM_annotation,
  genome_build,
  cell_line = NULL,
  interact = FALSE,
  return_data = FALSE
)

Arguments

peaklist A named list of peak files as GRanges object. If list is not named, default names will be assigned.

chromHMM_annotation ChromHMM annotation list.

genome_build The human genome reference build used to generate peakfiles. "hg19" or "hg38".

cell_line If not cell_line, will replace chromHMM_annotation by importing chromHMM data for a given cell line using get_chromHMM_annotation.

interact Default TRUE. By default, the heatmaps are interactive. If FALSE, the function generates a static ChromHMM heatmap.

return_data Return the plot data as in addition to the plot itself.

Value

ChromHMM heatmap, or a named list.

Examples

### Load Data ###
data("CnT_H3K27ac") # example dataset as GRanges object
data("CnR_H3K27ac") # example dataset as GRanges object
### Create Named Peaklist ###
peaklist <- list(CnT=CnT_H3K27ac, CnR=CnR_H3K27ac)
### Run ###
my_plot <- plot_chromHMM(peaklist = peaklist, 
  cell_line = "K562", 
  genome_build = "hg19")

### plot_corr

**Plot correlation of peak files**

#### Description

Plot correlation by binning genome and measuring correlation of peak quantile ranking. This ranking is based on p-value or other peak intensity measure dependent on the peak calling approach.

#### Usage

```r
plot_corr(
  peakfiles, 
  reference = NULL, 
  genome_build, 
  bin_size = 5000, 
  keep_chr = NULL, 
  drop_empty_chr = FALSE, 
  method = "spearman", 
  intensity_cols = c("total_signal", "qValue", "Peak Score", "score"), 
  interact = FALSE, 
  draw_cellnote = TRUE, 
  fill_diag = NA, 
  workers = check_workers(), 
  show_plot = TRUE, 
  save_path = tempfile(fileext = ".corr.csv.gz")
)
```

#### Arguments

- **peakfiles**: A list of peak files as GRanges object and/or as paths to BED files. If paths are provided, EpiCompare imports the file as GRanges object. EpiCompare also accepts a list containing a mix of GRanges objects and paths. Files must be listed and named using `list()`. E.g. `list("name1"=file1, "name2"=file2)`. If no names are specified, default file names will be assigned.

- **reference**: A named list containing reference peak file(s) as GRanges object. Please ensure that the reference file is listed and named i.e. `list("reference_name" = reference_peak)`. If more than one reference is specified, individual reports for each reference will be generated. However, please note that specifying more than one reference can take awhile. If a reference is specified, it enables two analyses: (1) plot showing statistical significance of overlapping/non-overlapping peaks; and (2) ChromHMM of overlapping/non-overlapping peaks.
genome_build  The build of **all** peak and reference files to calculate the correlation matrix on. If all peak and reference files are not of the same build use liftover_grlist to convert them all before running. Genome build should be one of hg19, hg38, mm9, mm10.

bin_size  Default of 100. Base-pair size of the bins created to measure correlation. Use smaller value for higher resolution but longer run time and larger memory usage.

keep_chr  Which chromosomes to keep.

drop_empty_chr  Drop chromosomes that are not present in any of the peakfiles (default: FALSE).

method  Default spearman (i.e. non-parametric). A character string indicating which correlation coefficient (or covariance) is to be computed. One of "pearson", "kendall", or "spearman": can be abbreviated.

intensity_cols  Depending on which columns are present, this value will be used to get quantiles and ultimately calculate the correlations:

- "total_signal": Used by the peak calling software SEACR. **NOTE**: Another SEACR column (e.g. "max_signal") can be used together or instead of "total_signal".
- "qValue": Used by the peak calling software MACS2/3. Should contain the negative log of the p-values after multiple testing correction.
- "Peak Score": Used by the peak calling software HOMER.

interact  Default TRUE. By default heatmap is interactive. If FALSE, heatmap is static.

draw_cellnote  Draw the numeric values within each heatmap cell.

fill_diag  Fill the diagonal of the overlap matrix.

workers  Number of threads to parallelize across.

show_plot  Show the plot.

save_path  Path to save a table of correlation results to.

Value

list with correlation plot (corr_plot) and correlation matrix (data)

Examples

data("CnR_H3K27ac")
data("CnT_H3K27ac")
data("encode_H3K27ac")
peakfiles <- list(CnR_H3K27ac=CnR_H3K27ac, CnT_H3K27ac=CnT_H3K27ac)
reference <- list("encode_H3K27ac"=encode_H3K27ac)
## Increasing bin_size for speed here,
## but lower values will give more precise results (and lower correlations)

cp <- plot_corr(peakfiles = peakfiles,
                reference = reference,
                genome_build = "hg19",
                bin_size = 5000,
                workers = 1)
Description

This function runs KEGG and GO enrichment analysis of peak files and generates dot plots.

Usage

```
plot_enrichment(
  peaklist,
  txdb = NULL,
  tss_distance = c(-3000, 3000),
  pvalueCutoff = 0.05,
  interact = FALSE,
  verbose = TRUE
)
```

Arguments

- `peaklist`: A list of peak files as GRanges object. Files must be listed and named using `list()`. e.g. `list("name1"=file1, "name2"=file2)`. If not named, default file names will be assigned.
- `txdb`: A TxDb annotation object from Bioconductor.
- `tss_distance`: A vector specifying the distance upstream and downstream around transcription start sites (TSS). The default value is `c(-3000,3000)`; meaning peak frequency 3000bp upstream and downstream of TSS will be displayed.
- `pvalueCutoff`: P-value cutoff, passed to `compareCluster`.
- `interact`: Default TRUE. By default, plots are interactive. If set FALSE, all plots in the report will be static.
- `verbose`: Print messages.

Value

KEGG and GO dot plots

Examples

```R
### Load Data ###
data("CnT_H3K27ac") # example peakfile GRanges object
data("CnR_H3K27ac") # example peakfile GRanges object
### Create Named Peaklist ###
peaklist <- list("CnT"=CnT_H3K27ac, "CnR"=CnR_H3K27ac)
enrich_res <- plot_enrichment(peaklist = peaklist, pvalueCutoff=1,
  tss_distance = c(-50,50))
```
plot_precision_recall  Plot precision-recall curves

Description

Plot precision-recall curves (and optionally F1 plots) by iteratively testing for peak overlap across a series of thresholds used to filter peakfiles. Each GRanges object in peakfiles will be used as the "query" against each GRanges object in reference as the subject. Will automatically use any columns that are specified with thresholding_cols and present within each GRanges object to create percentiles for thresholding. NOTE: Assumes that all GRanges in peakfiles and reference are already aligned to the same genome build.

Usage

plot_precision_recall(
  peakfiles,
  reference,
  thresholding_cols = c("total_signal", "qValue", "Peak Score"),
  initial_threshold = 0,
  n_threshold = 20,
  max_threshold = 1,
  workers = check_workers(),
  plot_f1 = TRUE,
  subtitle = NULL,
  color = "peaklist1",
  shape = color,
  facets = "peaklist2 ~ .",
  interact = FALSE,
  show_plot = TRUE,
  save_path = tempfile(fileext = "precision_recall.csv"),
  verbose = TRUE
)

Arguments

peakfiles A list of peak files as GRanges object and/or as paths to BED files. If paths are provided, EpiCompare imports the file as GRanges object. EpiCompare also accepts a list containing a mix of GRanges objects and paths. Files must be listed and named using list(). E.g. list("name1"=file1, "name2"=file2). If no names are specified, default file names will be assigned.

reference A named list containing reference peak file(s) as GRanges object. Please ensure that the reference file is listed and named i.e. list("reference_name" = reference_peak). If more than one reference is specified, individual reports for each reference will be generated. However, please note that specifying more than one reference can take awhile. If a reference is specified, it enables two analyses: (1) plot showing statistical significance of overlapping/non-overlapping peaks; and (2) ChromHMM of overlapping/non-overlapping peaks.
thesholding_cols

Depending on which columns are present, GRanges will be filtered at each threshold according to one or more of the following:

- "total_signal": Used by the peak calling software SEACR. **NOTE**: Another SEACR column (e.g. "max_signal") can be used together or instead of "total_signal".
- "qValue": Used by the peak calling software MACS2/3. Should contain the negative log of the p-values after multiple testing correction.
- "Peak Score": Used by the peak calling software HOMER.

initial_threshold

Numeric threshold that was provided to SEACR (via the parameter `--ctrl`) when calling peaks without an IgG control.

n_threshold

Number of thresholds to test.

max_threshold

Maximum threshold to test.

workers

Number of threads to parallelize across.

plot_f1

Generate a plot with the F1 score vs. threshold as well.

subtitle

Plot subtitle.

color

Variable to color data points by.

shape

Variable to set data point shapes by.

facets

**[Deprecated]** Please use rows and cols instead.

interact

Default TRUE. By default, plots are interactive. If set FALSE, all plots in the report will be static.

show_plot

Show the plot.

save_path

File path to save precision-recall results to.

verbose

Print messages.

**Value**

list with data and precision recall and F1 plots

**Examples**

data("CnR_H3K27ac")
data("CnT_H3K27ac")
data("encode_H3K27ac")

peakfiles <- list(CnR_H3K27ac=CnR_H3K27ac, CnT_H3K27ac=CnT_H3K27ac)
reference <- list("encode_H3K27ac" = encode_H3K27ac)

pr_out <- plot_precision_recall(peakfiles = peakfiles,
                               reference = reference,
                               workers = 1)
**precision_recall**  
Compute precision-recall

**Description**
Compute precision and recall using each GRanges object in peakfiles as the "query" against each GRanges object in reference as the subject.

**Usage**
```
precision_recall(
  peakfiles,  
  reference,  
  thresholding_cols = c("total_signal", "qValue", "Peak Score"),  
  initial_threshold = 0,  
  n_threshold = 20,  
  max_threshold = 1,  
  cast = TRUE,  
  workers = 1,  
  verbose = TRUE,  
  save_path = tempfile(fileext = "precision_recall.csv"),  
  ...  
)
```

**Arguments**
- **peakfiles**: A list of peak files as GRanges object and/or as paths to BED files. If paths are provided, EpiCompare imports the file as GRanges object. EpiCompare also accepts a list containing a mix of GRanges objects and paths. Files must be listed and named using `list()`.
  E.g. `list("name1"=file1, "name2"=file2)`. If no names are specified, default file names will be assigned.
- **reference**: A named list containing reference peak file(s) as GRanges object. Please ensure that the reference file is listed and named i.e. `list("reference_name" = reference_peak)`. If more than one reference is specified, individual reports for each reference will be generated. However, please note that specifying more than one reference can take awhile. If a reference is specified, it enables two analyses: (1) plot showing statistical significance of overlapping/non-overlapping peaks; and (2) ChromHMM of overlapping/non-overlapping peaks.
- **thresholding_cols**: Depending on which columns are present, GRanges will be filtered at each threshold according to one or more of the following:
  - "total_signal": Used by the peak calling software SEACR. **NOTE**: Another SEACR column (e.g. "max_signal") can be used together or instead of "total_signal".
  - "qValue": Used by the peak calling software MACS2/3. Should contain the negative log of the p-values after multiple testing correction.
• "Peak Score": Used by the peak calling software HOMER.

- `initial_threshold` Numeric threshold that was provided to SEACR (via the parameter --ctrl) when calling peaks without an IgG control.
- `n_threshold` Number of thresholds to test.
- `max_threshold` Maximum threshold to test.
- `cast` Cast the data into a format that’s more compatible with ggplot2.
- `workers` Number of threads to parallelize across.
- `verbose` Print messages.
- `save_path` File path to save precision-recall results to.
- `...` Arguments passed on to bpplapply
  - `apply_fun` Iterator function to use.
  - `register_now` Register the cores now with register (TRUE), or simply return the BPPARAM object (default: FALSE).
  - `use_snowparam` Whether to use SnowParam (default: TRUE) or MulticoreParam (FALSE) when parallelising across multiple workers.
  - `progressbar` logical(1) Enable progress bar (based on plyr:::progress_text).
- `X` Any object for which methods length, [ and [ are implemented.
- `FUN` The function to be applied to each element of X.

**Value**

Overlap

**Examples**

```r
data("CnR_H3K27ac")
data("CnT_H3K27ac")
data("encode_H3K27ac")
peakfiles <- list(CnR_H3K27ac=CnR_H3K27ac, CnT_H3K27ac=CnT_H3K27ac)
reference <- list("encode_H3K27ac" = encode_H3K27ac)

pr_df <- precision_recall(peakfiles = peakfiles,
                          reference = reference,
                          workers = 1)
```

**Description**

Converts a list of peak files to a symmetric matrix where the y-axis indicates precision and the x-axis indicates recall.
Usage

```r
precision_recall_matrix(peaklist, fill_diag = NA, verbose = TRUE)
```

Arguments

- `fill_diag` Fill the diagonal of the overlap matrix.
- `verbose` Print messages.

Value

```r
matrix
```

---

**predict_precision_recall**

*Predict precision-recall*

Description

Predict specific values of precision or recall by fitting a model to a precision-recall curve. Predictions that are <0 will automatically be set to 0. Predictions that are >100 will automatically be set to 100.

Usage

```r
predict_precision_recall(
  pr_df,
  fun = stats::loess,
  precision = seq(10, 100, 10),
  recall = seq(10, 100, 10)
)
```

Arguments

- `pr_df` Precision-recall data.frame generated by `precision_recall`.
- `fun` Function to fit the data with.
- `precision` Precision values to predict recall from.
- `recall` Recall values to predict precision from.

Value

A named list of fitted models and predictions.

Source

Fix for producing NAs from loess fun.
**Examples**

data("CnR_H3K27ac")
data("CnT_H3K27ac")
data("encode_H3K27ac")
peakfiles <- list(CnR_H3K27ac=CnR_H3K27ac, CnT_H3K27ac=CnT_H3K27ac)
reference <- list("encode_H3K27ac" = encode_H3K27ac)
pr_df <- precision_recall(peakfiles = peakfiles,
                          reference = reference)
predictions <- predict_precision_recall(pr_df = pr_df)

**predict_values**

*Predict values*

**Description**

Fit a model and make predictions from it.

**Usage**

predict_values(df, fun, values, input_var, predicted_var)

**Arguments**

- df: data.frame
- fun: Function to fit the data with.
- values: Values to make predictions from.
- input_var: Input variable column name.
- predicted_var: Predicted variable name.

**Value**

data.frame

**prepare_blacklist**

*Prepare blacklist as GRanges*

**Description**

Selects the appropriate blacklist in a variety of conditions.
prepare_genome_builds

Usage

prepare_blacklist(
  blacklist,
  output_build,
  blacklist_build = NULL,
  verbose = TRUE
)

Arguments

  output_build  Desired genome build for grlist to be lifted over to.
  blacklist_build Genome build of the blacklist. Only used when blacklist is a user-supplied
                  GRanges object.
  verbose       Print messages.

Value

  A GRanges objects of blacklisted genomic regions from the relevant genome build.

prepare_genome_builds  Prepare genome builds

Description

  Parse the genome_build argument into peaklist_build and reference_build.

Usage

prepare_genome_builds(genome_build, blacklist = NULL)

Arguments

  genome_build  A named list indicating the human genome build used to generate each of the
                following inputs:
                • "peakfiles": Genome build for the peakfiles input. Assumes genome
                  build is the same for each element in the peakfiles list.
                • "reference": Genome build for the reference input.
                • "blacklist": Genome build for the blacklist input.
                Example input list:
                genome_build = list(peakfiles="hg38", reference="hg19", blacklist="hg19")

                Alternatively, you can supply a single character string instead of a list. This
                should only be done in situations where all three inputs (peakfiles, reference,
                blacklist) are of the same genome build. For example:
                genome_build = "hg19"
prepare_peaklist

**blacklist**

A GRanges object containing blacklisted genomic regions. Blacklists included in EpiCompare are:

- NULL (default): Automatically selects the appropriate blacklist based on the genome_build_output argument.
- "hg19_blacklist": Regions of hg19 genome that have anomalous and/or unstructured signals. hg19_blacklist
- "hg38_blacklist": Regions of hg38 genome that have anomalous and/or unstructured signals. hg38_blacklist
- "mm10_blacklist": Regions of mm10 genome that have anomalous and/or unstructured signals. mm10_blacklist
- "mm9_blacklist": Blacklisted regions of mm10 genome that have been lifted over from mm10_blacklist. mm9_blacklist
- <user_input>: A custom user-provided blacklist in GRanges format.

**Value**

Named list.

---

**prepare_peaklist**

Prepare peaklist as GRanges

**Description**

Prepare peaklist as GRanges

**Usage**

prepare_peaklist(peaklist, remove_empty = TRUE, as_grangeslist = FALSE)

**Arguments**

- **peaklist**
  A named list of peaks as GRanges or paths to BED files.
- **remove_empty**
  Remove any empty elements in the list.
- **as_grangeslist**
  Convert output to class GRangesList before returning.

**Value**

A list of GRanges objects
prepare_reference  
Prepare reference as GRanges

Description
Prepare reference as GRanges

Usage
```r
prepare_reference(
  reference,  # A named list of GRanges objects, or a single GRanges object to be converted into a named list.
  max_elements = NULL,  # Max number of elements to use within the list. Set to NULL (default) to use all elements.
  remove_empty = TRUE,  # Remove any empty elements in the list.
  as_list = TRUE,  # Return as a list.
  as_grangeslist = FALSE  # Return as a GRangesList (overrides as_list).
)
```

Arguments
- **reference**: A named list of GRanges objects, or a single GRanges object to be converted into a named list.
- **max_elements**: Max number of elements to use within the list. Set to `NULL` (default) to use all elements.
- **remove_empty**: Remove any empty elements in the list.
- **as_list**: Return as a list.
- **as_grangeslist**: Return as a GRangesList (overrides as_list).

Value
A list of GRanges objects

read_bowtie  
Read bowtie

Description
Read a bowtie file.

Usage
```r
read_bowtie(path, verbose = TRUE)
```

Arguments
- **path**: Path to bowtie file.
- **verbose**: Print messages.
**read_peaks**

**Value**

```
data.table
```

**Description**

Read peak files.

**Usage**

```
read_peaks(path, type, verbose = TRUE)
```

**Arguments**

- `path`: Path to peak file.
- `type`: File type to search for. Options include:
  - ":<pattern>" finds files matching an arbitrary regex pattern specified by user.
  - "peaks.stringent" finds files ending in "*.stringent.bed$"
  - "peaks.consensus" finds files ending in "*.consensus.peaks.bed$"
  - "peaks.consensus.filtered" finds files ending in "*.consensus.peaks.filtered.awk.bed$"
  - "picard" finds files ending in "*.target.markdup.MarkDuplicates.metrics.txt$"
- `verbose`: Print messages.

**Value**

```
GRanges
```

**rebin_peaks**

**Description**

Standardise a list of peak files by rebinning them into fixed-width tiles across the genome.
rebin_peaks

Usage

rebin_peaks(
  peakfiles,
  genome_build,
  intensity_cols = c("total_signal", "qValue", "Peak Score", "score"),
  bin_size = 5000,
  keep_chr = NULL,
  sep = c(":", "-"),
  drop_empty_chr = FALSE,
  as_sparse = TRUE,
  workers = check_workers(),
  verbose = TRUE,
  ...
)

Arguments

peakfiles A list of peak files as GRanges object and/or as paths to BED files. If paths are provided, EpiCompare imports the file as GRanges object. EpiCompare also accepts a list containing a mix of GRanges objects and paths. Files must be listed and named using list(). E.g. list("name1"=file1, "name2"=file2). If no names are specified, default file names will be assigned.

genome_build The build of all peak and reference files to calculate the correlation matrix on. If all peak and reference files are not of the same build use liftover_grlist to convert them all before running. Genome build should be one of hg19, hg38, mm9, mm10.

intensity_cols Depending on which columns are present, this value will be used to get quantiles and ultimately calculate the correlations:
  • "total_signal": Used by the peak calling software SEACR. NOTE: Another SEACR column (e.g. "max_signal") can be used together or instead of "total_signal".
  • "qValue": Used by the peak calling software MACS2/3. Should contain the negative log of the p-values after multiple testing correction.
  • "Peak Score": Used by the peak calling software HOMER.

bin_size Default of 100. Base-pair size of the bins created to measure correlation. Use smaller value for higher resolution but longer run time and larger memory usage.

keep_chr Which chromosomes to keep.

sep Separator to be used after chromosome name (first item) and between start/end genomic coordinates (second item).

drop_empty_chr Drop chromosomes that are not present in any of the peakfiles (default: FALSE).

as_sparse Return the rebinned peaks as a sparse matrix (default: TRUE), which is more efficiently stored than a dense matrix (FALSE).

workers Number of threads to parallelize across.

verbose Print messages.

... Arguments passed on to bpplapply
**apply_fun**  Iterator function to use.

**register_now**  Register the cores now with `register` (TRUE), or simply return the BPPARAM object (default: FALSE).

**use_snowparam**  Whether to use SnowParam (default: TRUE) or MulticoreParam (FALSE) when parallelising across multiple workers.

**progressbar**  logical(1) Enable progress bar (based on plyr:::progress_text).

**X**  Any object for which methods `length`, `[`, and `[[` are implemented.

**FUN**  The function to be applied to each element of `X`.

---

### Value

Binned peaks matrix

### Examples

```r
data("CnR_H3K27ac")
data("CnT_H3K27ac")
peakfiles <- list(CnR_H3K27ac=CnR_H3K27ac, CnT_H3K27ac=CnT_H3K27ac)

#increasing bin_size for speed
peakfiles_rebinned <- rebin_peaks(peakfiles = peakfiles,
genome_build = "hg19",
bin_size = 5000,
workers = 1)
```

---

**remove_nonstandard_chrom**

Remove non-standard chromosomes

### Description

Remove non-standard chromosomes from a list of GRanges objects.

### Usage

```r
remove_nonstandard_chrom(
grlist,
keep_chr = paste0("chr", c(seq_len(22), "X", "Y")),
verbose = TRUE
)
```

### Arguments

- **grlist**  Named list of GRanges objects.
- **keep_chr**  Which chromosomes to keep.
- **verbose**  Print messages.
Value

Named list of \texttt{GRanges} objects.

\begin{Verbatim}
\texttt{report\_command}
\end{Verbatim}

\textit{Report command}

\section*{Description}

Reconstruct the \texttt{EpiCompare} command used to generate the current Rmarkdown report.

\section*{Usage}

\texttt{report\_command(params, peaklist\_tidy, reference\_tidy)}

\section*{Arguments}

\begin{itemize}
  \item \texttt{params} Parameters supplied to the Rmarkdown template.
  \item \texttt{peaklist\_tidy} Post-processed target peaks.
  \item \texttt{reference\_tidy} Post-processed reference peaks.
\end{itemize}

\section*{Value}

String reconstructing R function call.

\section*{Examples}

\begin{Verbatim}
# report\_command()
\end{Verbatim}

\section*{report\_header}

\textit{Report header}

\section*{Description}

Generate a header for \texttt{EpiCompare} reports generated using the \texttt{EpiCompare.Rmd} template.

\section*{Usage}

\texttt{report\_header()}

\section*{Value}

Header string to be rendering within Rmarkdown file.

\section*{Examples}

\texttt{report\_header()}

\section*{Value}

Named list of \texttt{GRanges} objects.
save_output

Description
This function saves data frames and plots generated by EpiCompare.

Usage
save_output(
  save_output = FALSE,
  file,
  file_type,
  filename,
  outpath,
  interactive = FALSE,
  verbose = TRUE
)

Arguments

save_output Default FALSE. If TRUE, outputs are saved.
file Tables and plots to be saved.
file_type Type of file to be saved. "data.frame", "ggplot", "image"
filename Name of file.
outpath Outpath
interactive Default FALSE. If TRUE, interactive plots are saved as html.
verbose Print messages.

Value
Saved data frames and plots.

set_min_max

Description
Set the min/max values in a data.frame.

Usage
set_min_max(df, colname, min_val = 0, max_val = 100)
Arguments

<table>
<thead>
<tr>
<th>df</th>
<th>data.frame</th>
</tr>
</thead>
<tbody>
<tr>
<td>colname</td>
<td>Column name to check.</td>
</tr>
<tr>
<td>min_val</td>
<td>Minimum value.</td>
</tr>
<tr>
<td>max_val</td>
<td>Maximum value.</td>
</tr>
</tbody>
</table>

Value

data.frame

stopper

| Stop messages |

Description

Conditionally print stop messages. Allows developers to easily control verbosity of functions, and meet Bioconductor requirements that dictate the stop message must first be stored to a variable before passing to stop.

Usage

stopper(..., v = TRUE)

Arguments

| v | Whether to print messages or not. |

Value

Null

tidy_peakfile

| Tidy peakfiles in GRanges |

Description

This function filters peak files by removing peaks in blacklisted regions and in non-standard chromosomes. It also checks that the input list of peakfiles is named. If no names are provided, default file names will be used.

Usage

tidy_peakfile(peaklist, blacklist)
translate_genome

Arguments

peaklist A named list of peak files as GRanges object. Objects must be named and listed using list(). e.g. list("name1"=file1, "name2"=file2) If not named, default names are assigned.

blacklist Peakfile specifying blacklisted regions as GRanges object.

Value

list of GRanges object

Examples

### Load Data ###
data("encode_H3K27ac") # example peakfile GRanges object
data("CnT_H3K27ac") # example peakfile GRanges object
data("hg19_blacklist") # blacklist region for hg19 genome

### Create Named Peaklist ###
peaklist <- list("encode"=encode_H3K27ac, "CnT"=CnT_H3K27ac)

### Run ###
peaklist_tidy <- tidy_peakfile(peaklist = peaklist,
  blacklist = hg19_blacklist)

translate_genome  Translate genome

Description

Translate the name of a genome build from one format to another.

Usage

translate_genome(
  genome,
  style = c("UCSC", "Ensembl", "NCBI"),
  default_genome = NULL,
  omit_subversion = TRUE
)

Arguments

gene A character vector of genomes equivalent to UCSC version or Ensembl Assemblies

style A single value equivalent to "UCSC" or "Ensembl" specifying the output genome

default_genome Default genome build when genome is NULL.

omit_subversion Omit any subversion suffixes after the ".".
Value

Standardized genome build name as a character string.

Examples

```r
gene <- translate_genome(genome="hg38", style="Ensembl")
gene2 <- translate_genome(genome="mm10", style="UCSC")
```

---

**tss_plot**

Read count frequency around TSS

Description

This function generates a plot of read count frequency around TSS.

Usage

```r
tss_plot(  
  peaklist,  
  txdb = NULL,  
  tss_distance = c(-3000, 3000),  
  conf = 0.95,  
  resample = 500,  
  interact = FALSE,  
  workers = check_workers()  
)
```

Arguments

- **peaklist**: A list of peak files as GRanges object. Files must be listed and named using `list()`. e.g. `list("name1"=file1,"name2"=file2)` If not named, default file names will be assigned.
- **txdb**: A TxDb annotation object from Bioconductor.
- **tss_distance**: A vector specifying the distance upstream and downstream around transcription start sites (TSS). The default value is `c(-3000, 3000)`; meaning peak frequency 3000bp upstream and downstream of TSS will be displayed.
- **conf**: Confidence interval threshold estimated by bootstrapping (0.95 means 95 Argument passed to `plotAvgProf`.
- **resample**: Number of bootstrapped iterations to run. Argument passed to `plotAvgProf`.
- **interact**: Default TRUE. By default, plots are interactive. If set FALSE, all plots in the report will be static.
- **workers**: Number of cores to parallelise bootstrapping across. Argument passed to `plotAvgProf`. 
### width_boxplot

**Value**

A named list of profile plots.

**Examples**

```r
### Load Data ###
data("CnT_H3K27ac") # example peaklist GRanges object
data("CnR_H3K27ac") # example peaklist GRanges object

### Create Named Peaklist ###
peaklist <- list("CnT"=CnT_H3K27ac, "CnR"=CnR_H3K27ac)
my_plot <- tss_plot(peaklist = peaklist,
                    tss_distance=c(-50,50),
                    workers = 1)

width_boxplot(peaklist, interact = FALSE)
```

**Description**

This function creates boxplots showing the distribution of widths in each peak file.

**Usage**

```r
width_boxplot(peaklist, interact = FALSE)
```

**Arguments**

- `peaklist`: A list of peak files as GRanges object. Files must be named and listed using `list()`. E.g. `list("name1"=file1, "name2"=file2)`
- `interact`: Default TRUE. By default, plots are interactive. If set FALSE, all plots in the report will be static.

**Value**

A boxplot of peak widths.

**Examples**

```r
### Load Data ###
data("encode_H3K27ac") # example peaklist GRanges object
data("CnT_H3K27ac") # example peaklist GRanges object
peaklist <- list("encode"=encode_H3K27ac, "CnT"=CnT_H3K27ac)
my_plot <- width_boxplot(peaklist = peaklist)
```
write_example_peaks  Write example peaks

Description
Write example peaks datasets to disk.

Usage

```r
write_example_peaks(
  dir = file.path(tempdir(), "processed_results"),
  datasets = c("encode_H3K27ac", "CnT_H3K27ac", "CnR_H3K27ac")
)
```

Arguments

- `dir`  Directory to save peak files to.
- `datasets`  Example datasets from EpiCompare to write.

Value

Named vector of paths to saved peak files.

Examples

```r
save_paths <- EpiCompare::write_example_peaks()
```
Index

* datasets
  - CnR_H3K27ac, 9
  - CnR_H3K27ac_picard, 10
  - CnT_H3K27ac, 11
  - CnT_H3K27ac_picard, 11
  - encode_H3K27ac, 17
  - hg19_blacklist, 27
  - hg38_blacklist, 28
  - mm10_blacklist, 31
  - mm9_blacklist, 32

* internal
  - as_interactive, 4
  - check_cell_lines, 6
  - check_genome_build, 7
  - check_grlist_cols, 7
  - check_list_names, 8
  - checkCache, 6
  - clean_granges, 9
  - fig_length, 22
  - gather_files_names, 24
  - get_bpparam, 25
  - get_chromHMM_annotation, 26
  - is_granges, 29
  - message_parallel, 31
  - messager, 30
  - precision_recall_matrix, 45
  - predict_values, 47
  - prepare_blacklist, 47
  - prepare_genome_builds, 48
  - prepare_peaklist, 49
  - prepare_reference, 50
  - read_bowtie, 50
  - read_peaks, 51
  - remove_nonstandard_chrom, 53
  - save_output, 55
  - set_min_max, 55
  - stopper, 56

  as_interactive, 4

  BiocParallel, 25
  BiocParallel::bplapply, 5
  BiocParallelParam, 5
  bplapply, 4
  bpoptions, 5
  bplapply, 4, 45, 52
  check_cell_lines, 6
  check_genome_build, 7
  check_grlist_cols, 7
  check_list_names, 8
  check_workers, 8
  checkCache, 6
  clean_granges, 9
  closeAllConnections, 13
  CnR_H3K27ac, 9
  CnR_H3K27ac_picard, 10
  CnT_H3K27ac, 11
  CnT_H3K27ac_picard, 11
  compareCluster, 41
  compute_consensus_peaks, 12, 13
  compute_corr, 14
  consensusSeekerR::findConsensusPeakRegions, 13
  data.table, 51
  download_button, 16
  encode_H3K27ac, 17
  enrichPeakOverlap, 34
  EpiCompare, 18, 54
  fig_length, 22
  findConsensusPeakRegions, 12
  fragment_info, 22
  gather_files, 23, 24
  gather_files_names, 24
  get_bpparam, 25
  get_chromHMM_annotation, 26, 38
  ggplot, 4
GRanges, 7, 9, 12, 13, 19, 23, 24, 26, 29, 30, 42–44, 48–51, 53, 54
GRangesList, 30, 49, 50
group_files, 27
hg19_blacklist, 19, 27, 49
hg38_blacklist, 19, 28, 49
is_granges, 29
layout, 4
liftover_grlist, 15, 29, 40, 52
list, 38
message, 30
message_parallel, 31
messager, 30
mm10_blacklist, 19, 31, 49
mm9_blacklist, 19, 32, 49
MulticoreParam, 5, 25, 45, 53

opts_chunk, 21
overlap_heatmap, 32
overlap_percent, 33
overlap_stat_plot, 34
overlap_upset_plot, 35
peak_info, 36
plot_ChIPseeker_annotation, 37
plot_chromHMM, 38
plot_corr, 20, 39
plot_enrichment, 41
plot_precision_recall, 20, 42
plotAvgProf, 58
plotly, 4
precision_recall, 44, 46
precision_recall_matrix, 45
predict_precision_recall, 46
predict_values, 47
prepare_blacklist, 47
prepare_genome_builds, 48
prepare_peaklist, 49
prepare_reference, 50
read_bowtie, 50
read_peaks, 51
rebin_peaks, 51
register, 5, 25, 45, 53
remove_nonstandard_chrom, 53
report_command, 54
report_header, 54
save_output, 55
seqlevelsStyle, 30
SerialParam, 25
set_min_max, 55
SnowParam, 5, 25, 45, 53
stop, 56
stopper, 56
subsetByOverlaps, 33
tagList, 4
tidy_peakfile, 56
translate_genome, 57
tss_plot, 58
width_boxplot, 59
write_example_plot, 60