Package ‘HodgesTools’

December 8, 2022

Title Common Use Tools for Genomic Analysis

Version 1.0.0

Description Built by Hodges lab members for current and future Hodges lab members. Other individuals are welcome to use as well. Provides useful functions that the lab uses everyday to analyze various genomic datasets. Critically, only general use functions are provided; functions specific to a given technique are reserved for a separate package. As the lab grows, we expect to continue adding functions to the package to build on previous lab members code.

Depends R (>= 3.6)

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.1.2

Imports dplyr, ggplot2, magrittr, RColorBrewer, readr, ini, qqman, data.table, RecordLinkage

NeedsCompilation no

Author Tyler Hansen [aut, cre, cph],
Tim Scott [aut, ctb, cph],
Lindsey Guerin [aut, ctb, cph],
Verda Agan [aut, ctb, cph],
Emily Hodges [aut, fnd, cph]

Maintainer Tyler Hansen <tyler.j.hansen@vanderbilt.edu>

Repository CRAN

Date/Publication 2022-12-08 13:52:36 UTC

R topics documented:

append_section_to_ini *Append section to ini file*

Description

Takes a new section in ini format and adds to existing ini.

Usage

```
append_section_to_ini(ini_file, new_section)
```

Arguments

ini_file	file location of config.ini file
new_section	named list of the section list

Details

The new_section must be a named list of the section list. See examples.

Value

No return value. Edits and overwrites input config.ini file.

Author(s)

Tyler Hansen

Examples

```
#list of key-value pairs
CHRACC <- list(dir='/chrAcc_peaks/',
               peaks='/chrAcc_peaks/GM12878_genrich.narrowPeak')

#list of section, resulting in list of list.
new_section <- list(CHRACC=CHRACC)

#write ini
ini_file <- system.file("extdata", "config.ini")
append_section_to_ini(ini_file, new_section)
```

`cpg_analysis` *CpG Analysis*

Description

Compute observed/expected CpG ratio and GC% for regions of interest

Usage

```
cpg_analysis(
  list = FALSE,
  count,
  cpg_file,
  nuc_file,
  palette = "Set3",
  plot = "none"
)
```

Arguments

`list`	"boolean of whether input is a list of groups. Default = FALSE."
`count`	"numeric value for the number of files included in your list
`cpg_file`	"file names or list of files names for your CpGcount.txt files. This is defined in cpg_analysis.sh"
`nuc_file`	"file names or list of files names for your nucOutput_gc.txt files. This is defined in cpg_analysis.sh"
`palette`	"if choosing to plot, the RColorBrewer palette you would like to be applied to your plot"
`plot`	"one of three choices depending on what output you would like: 'none' for no plot, 'ratio' for observed/expected ratios, 'gc_percent' for GC%"

Details

The function reads in a nucOutput_gc and CpGcount text file The function uses the nucOutput_gc and CpGcount file to calculates observed/expected ratio and GC%. The function allows the option to plot the distribution of these values in ggplot2

Value

ggplot object or tibble if plot="none"

Author(s)

Lindsey Guerin

Examples

```
#load external data

gain_6hr_CpG <- system.file(package = "HodgesTools", "extdata",
"cov5root_6hr_gain.CpGcount.txt")
gain_12hr_CpG <- system.file(package = "HodgesTools", "extdata",
 "cov5root_12hr_gain.CpGcount.txt")
gain_6hr_nuc <- system.file(package = "HodgesTools", "extdata",
 "cov5root_6hr_gain.nucOutput_gc.txt")
gain_12hr_nuc <- system.file(package = "HodgesTools", "extdata",
"cov5root_12hr_gain.nucOutput_gc.txt")

#Make a density plot of GC% values for a list of two region of interest files
cpg_analysis(list = TRUE, count = 2, cpg_file = list(gain_6hr_CpG, gain_12hr_CpG),
nuc_file= list(gain_6hr_nuc, gain_12hr_nuc), palette = "Set3", plot ="gc_percent")

#Make a density plot of observed/expected values for a single set of regions of interest
cpg_analysis(list = FALSE, cpg_file = gain_6hr_CpG,
nuc_file = gain_6hr_nuc, palette = "Set3", plot ="ratio")
```

---

createManhattandQQ *Creating a Manhattan Plot and QQ plot*

Description

Creates a Manhattan plot and QQ plot using GWAS results output from PLINK

Usage

```
createManhattandQQ(
  gwas_results,
  highlights_file = NULL,
  suggestive_line = -log10(0.05),
  set_color_vector = c("gray10", "gray60"),
  genomewide_line = -log10(5e-08),
  annotate_Pval = 0.05,
  y_lim = c(0, 8)
)
```

Arguments

`gwas_results` output file listing SNP-trait association values for GWAS run using PLINK

`highlights_file`
a text file with a 'snp' column listing the SNPs to annotate/color on the Manhattan plot

`suggestive_line`
where to draw a "suggestive" line; default -log10(1e-5).

`set_color_vector`
: a character vector listing colors in palette of interest (you must create this chr object before calling the createManhattanandQQ function and assign it to set_color_vector)

`genomewide_line`
: where to draw a "genome-wide significant" line; default -log10(5e-8)

`annotate_Pval`
: if set, SNPs below this p-value will be annotated on the plot; default is 0.05

`y_lim`
: set the y-axis limits; default is c(0,8)

Details

This function reads in a GWAS result file output from plink2 listing the coordinates, ids, and associated p-values for SNPs under study This function also has the option of reading in a "highlights" file listing the IDs of SNPs to annotate/color on the Manhattan plot

Value

a Manhattan plot of SNP-trait associations and QQ plot

Author(s)

Verda Agan

Examples

```
#' #load external data.
gwas_results <- system.file(package = "HodgesTools", "extdata",
"createManhattandQQ_example_sum_stats.txt")
snps_to_annotate <- system.file(package = "HodgesTools", "extdata",
"createManhattandQQ_example_highlights_file.txt")

#Make a Manhattan plot that highlights a select list of SNPs subset from GWAS results
createManhattandQQ(gwas_results, highlights_file=snps_to_annotate,
suggestive_line = -log10(0.001), set_color_vector = c("gray10", "gray60"),
genomewide_line = -log10(5e-8), annotate_Pval = 0.001, y_lim =c(0,8))

#Make a Manhattan plot that doesn't highlight a select list of SNPs subset from GWAS results
createManhattandQQ(gwas_results, suggestive_line = -log10(0.001),
set_color_vector = c("gray10", "gray60"), genomewide_line = -log10(5e-8),
annotate_Pval = 0.001, y_lim =c(0,8))
```

`helper_collapseTableByLevenSim` *helper_collapseTableByLevenSim*

Description

Reads in a table and value for Levenshtein threshold and returns a table collapsed by threshold (highest p-value for each group)

Usage

```
helper_collapseTableByLevenSim(inputTable, levenSimThresholdVal)
```

Arguments

`inputTable`	dataframe. HOMER output table modified in the parent script–ready for filtering by Levenshtein similarity.
`levenSimThresholdVal`	float. Value for thresholding TFs. For groups of TFs with similar consensus sequences, the TF with the lowest p-value by HOMER will be retained.

Value

tibble

Author(s)

Tim Scott

`helper_getMaxLevenSimCol` *helper_getMaxLevenSimCol*

Description

Reads in a vector of motifs and returns a

Usage

```
helper_getMaxLevenSimCol(vectorOfMotifs)
```

Arguments

`vectorOfMotifs`	vector of char. Vector of motifs to filter through.

Value

data.frame

Author(s)

Tim Scott

`helper_makeBigTableFromListOfStandardTables` *makeBigTableFromListofTables*

Description

Reads in a list of tables and return list of tables with percent Fold Change (enrichment)

Usage

```
helper_makeBigTableFromListOfStandardTables(inputListOfTables)
```

Arguments

`inputListOfTables`
list of dataframe. List of HOMER TF knownResults Tables.

Value

list of tibble

Author(s)

Tim Scott

`plot_HOMERTFs` *Plot HOMER TF enrichment results*

Description

Plot HOMER TF enrichment results

Usage

```
plot_HOMERTFs(
  dir = "/directory/of/results/",
  show = 3,
  qThreshold = 0.05,
  levenSimThreshold = 1
)
```

Arguments

`dir` string. Input directory containing HOMER findMotifsGenome.pl output files in format: *knownResults.txt

`show` int. Number of rows to show per input file, ranked by p-value.

`qThreshold` int. Value for thresholding HOMER enrichment results by q-value.

`levenSimThreshold` float. Value for thresholding TFs. For groups of TFs with similar consensus sequences, the TF with the lowest p-value by HOMER will be retained.

Details

Make bigTable of all TFs to pull from so a single TF can have data from different input Files (e.g. across CTs):

(5) Create bigTable of all q-value TF results concatenated together (6) Filter by consensus list (4)and make a gg-plot appropriate table and Plot (7-8) Order factors and plot

Value

ggplot object

Strategy:

Find motifs to extract:

(1) Filter each element table by q-value. Should basically chop off the bottom portion of list, making the rest of the script less computationally cumbersome (2) Collapse each element table by by Levenshtein Similarity (3) Filter each element table (in my case: cell type-specific results file) to top X rows (4) Extract consensus columns from each element table and store as a variable]

Author(s)

Tim Scott

`read_bed` *Read bed file*

Description

Reads in a tab-delimited BED formatted file into R.

Usage

```
read_bed(file, extra_col_names = c(), length = FALSE, verbose = TRUE)
```

Arguments

`file` bed file

`extra_col_names` list of strings specifying extra column names

`length` boolean of whether to add length column

`verbose` boolean set to see function behavior

Details

First three columns of file must be the genomic coordinates of the regions (i.e. chr start end).

read_bed will auto-detect BED3 and BED6 formats. It will also detect BED3+ and BED6+ formats assigning generic or user-defined col_names to the additional column(s).

Value

tibble

Author(s)

Tyler Hansen & Tim Scott

Examples

```
#load external data.
BED3 <- system.file(package = "HodgesTools", "extdata", "test_BED3.bed")
BED6 <- system.file(package = "HodgesTools", "extdata", "test_BED6.bed")
BED4 <- system.file(package = "HodgesTools", "extdata", "test_BED4.bed")
BED8 <- system.file(package = "HodgesTools", "extdata", "test_BED8.bed")

# Read 3-column BED file.
read_bed(BED3)

# Read 6-column BED file.
read_bed(BED6)

# Read 3-column BED file and add length column.
read_bed(BED3, length = TRUE)

# Read 3 column format BED file with additional fourth column. Add generic column names.
read_bed(BED4)

# Read 3 column format BED file with additional fourth column. Specify additional column names.
read_bed(BED4, extra_col_names = c("fourthColumn"))

# Read 6 column format BED file with additional columns. Specify additional column names.
read_bed(BED8, extra_col_names = c("seventhColumn", "eigthColumn"))
```

Index