Package 'goldmine'

May 2, 2015

Maintainer Jeffrey Bhasin <jeffb@case.edu>

Author Jeffrey Bhasin <jeffb@case.edu>

Version 1.0

License MIT

Title goldmine: Genomic context annotation for any set of genomic ranges using UCSC Genome Browser tables

Description Goldmine obtains data by direct downloading and updating of a local mirror of select UCSC Genome Browser annotation tables. The R package contains functions to assess genomic context of any given set of genomic ranges by performing overlaps with regions of annotated genomic features and produce long, short, and plot outputs. There are also functions for performing enrichment testing by drawing background sets matched for multiple variables (i.e., length, CpG density, etc).

Depends GenomicRanges,
data.table,
stringr,
ggplot2,
parallel,
IRanges

Imports httr,
RCurl,
R.utils,
gtools,
Matching,
rms,
grid,
gridExtra,
RColorBrewer,
reshape

R topics documented:

`goldmine-package` *goldmine: Genomic context annotation for any set of genomic ranges using UCSC Genome Browser tables*

Description

goldmine: Genomic context annotation for any set of genomic ranges using UCSC Genome Browser tables

---

`addGenes` *Add columns with distance to nearest gene and gene symbol(s)*

Description

Add columns with distance to nearest gene and gene symbol(s)

Usage

```
addGenes(query, geneset, genome, cachedir, sync = TRUE)
```

Arguments

`query`	Genomic regions to find nearest genes for as a GRanges, data.frame, or data.table with the columns "chr", "start", and "end"
`geneset`	Select one of "ucsc" for the UCSC Genes (from the knownGene table), "refseq" for RefSeq genes (from the refFlat table), or "ensembl" for the Ensembl genes (from the ensGene table)
`genome`	UCSC genome name to use (e.g. hg19, mm10)
`cachedir`	Path where cached UCSC tables are stores
`sync`	If TRUE, then check if newer versions of UCSC tables are available and download them if so. If FALSE, skip this check. Can be used to freeze data versions in an analysis-specific cachedir for reproducibility.

addNearest — *Add columns to query with distance to nearest subject and subject id(s)*

Description

Add columns to query with distance to nearest subject and subject id(s)

Usage

```
addNearest(query, subject, id = "name", prefix = "subject")
```

Arguments

query	Genomic regions to find nearest genes for as a GRanges, data.frame, or data.table with the columns "chr", "start", and "end"
id	Column name of the id field in subject to report as the nearest id(s). In case of ties, a comma separated list will be returned.
prefix	Append this string to names of the added columns
query	Genomic regions to find nearest genes for as a GRanges, data.frame, or data.table with the columns "chr", "start", and "end"

doPropMatch — *Perform propensity score matching to draw a multi-variate matched set of sequences from a background pool*

Description

Given a query set and a background pool, draw a set of sequences from the background pool that most closely match the query set with respect to multiple co-variates.

Usage

```
doPropMatch(query, pool, outdir = ".", formula, n = 1, bsg, genome,
  cachedir)
```

Arguments

query	A data.frame or data.table with columns "chr", "start", and "end" and any other columns. If a data.frame or data.table, must contain the columns "chr", "start", "end", where the "start" coordinates are 1-based.
pool	A data.frame or data.table with columns "chr", "start", and "end" and any other columns. If a data.frame or data.table, must contain the columns "chr", "start", "end", where the "start" coordinates are 1-based.
outdir	The function will write a PDF report of matching performance and FASTA files for both the query set and matched null set. Provide a directory in which to save these files.

formula	Formula used for matching. For example: "treat ~ sizeLog + freqCpG". The variable "treat" must always be given as the predicted variable. Combinations of predictors can be selected from the set of: size (length of sequence in bp), sizeLog (log of size - recommended for best matching performance), gc (GC percent), freqCpG (dinucleotide frequency of CpG sites), freqA (frequency of the base A), freqT, freqC, freqG, repeatPer (percent of sequence covered by repeat masked regions), distTSSCenterLogX1 (distance to transcription start sites, log transformed), and distTSECenterLogX1 (distance to transcription end sites, log transformed).
n	Number of times greater than the query the matched null set will be.
bsg	BString genome from which sequence data can be derived. For example, see the "BSgenome.Hsapiens.UCSC.hg19" for the "hg19" genome BSGenome package from Bioconductor. Similar packages exist for other genomes.
genome	The UCSC name specific to the genome of the query coordinates (e.g. "hg19", "hg18", "mm10", etc)
cachedir	A path to a directory where a local cache of UCSC tables are stored. If equal to NULL (default), the data will be downloaded to temporary files and loaded on the fly. Caching is highly recommended to save time and bandwidth.

Value

A list containing the sequences of both the target and pool along with a GRanges of the matched results which can be used as a null set in testEnrichment().

---

drawGenomePool *Draw a length-matched pool of sequences from the genome*

Description

Given a query set of ranges, draw a length-matched pool of sequences. Returned ranges are required to (1) not overlap with each other or the query, (2) not extend off chromosome ends, (3) not extend over assembly gaps as defined in the UCSC "gap" table for the given genome assembly.

Usage

```
drawGenomePool(query, n, genome, cachedir, chrs = NULL)
```

Arguments

query	A data.frame or data.table with columns "chr", "start", and "end" and any other columns. If a data.frame or data.table, must contain the columns "chr", "start", "end", where the "start" coordinates are 1-based.
n	Number of times greater than the query set that the size of the returned background pool will be
genome	The UCSC name specific to the genome of the query coordinates (e.g. "hg19", "hg18", "mm10", etc)
cachedir	A path to a directory where a local cache of UCSC tables are stored. If equal to NULL (default), the data will be downloaded to temporary files and loaded on the fly. Caching is highly recommended to save time and bandwidth.

Value

A GRanges of the background sequences.

`getCpgFeatures` *Generate feature sets based on CpG island, shore, and shelf regions*

Description

Uses the "cpgIslandExt" table to generate shore (+/- 2kb from islands) and shelf (+/- 2kb from shores) regions. Will only function for genomes with this table available. Can be concatenated using c() with other tables, such as from getFeatures(), and provided as input for the "features" argument of goldmine().

Usage

```
getCpgFeatures(genome, cachedir)
```

Arguments

`genome`	The UCSC name specific to the genome of the query coordinates (e.g. "hg19", "hg18", "mm10", etc)
`cachedir`	A path to a directory where a local cache of UCSC tables are stored. If equal to `NULL` (default), the data will be downloaded to temporary files and loaded on the fly. Caching is highly recommended to save time and bandwidth.

`getFeatures` *Obtain feature sets from UCSC genome browser tables*

Description

Given a vector of table names from the UCSC genome browser that all contain "chrom", "chromStart", and "chromEnd" fields, converts them to input suitable for the goldmine() "features" argument (changes column names and adjusts 0-based start coordinates to 1-based).

Usage

```
getFeatures(tables = c("wgEncodeRegDnaseClusteredV3",
  "wgEncodeRegTfbsClusteredV3"), genome, cachedir, sync = TRUE)
```

Arguments

`tables`	A vector of table names from UCSC (default: DnaseI and TFBS from encode for hg19).
`genome`	The UCSC name specific to the genome of the query coordinates (e.g. "hg19", "hg18", "mm10", etc)
`cachedir`	A path to a directory where a local cache of UCSC tables are stored. If equal to `NULL` (default), the data will be downloaded to temporary files and loaded on the fly. Caching is highly recommended to save time and bandwidth.
`sync`	If TRUE, then check if newer versions of UCSC tables are available and download them if so. If FALSE, skip this check. Can be used to freeze data versions in an analysis-specific cachedir for reproducibility.

`getGeneModels` *Return sets of ranges for individual gene model components based on the contents of the output from getGenes()*

Description

For custom analysis that requires the genomic ranges of gene model components.

Usage

```
getGeneModels(genes = getGenes(geneset = "ucsc", genome = genome, cachedir =
  cachedir), promoter = c(1000, 500), end3 = c(1000, 1000), genome,
  cachedir, sync = TRUE)
```

Arguments

`genes`	Genes of interest from the output table of getGenes(). If not given, will default to the UCSC knownGene table.
`promoter`	A numeric vector of length 2 specifying the number of bp upstream and downstream of transcription start sites for which to create promoter ranges. Given as c(upstream,downstream). Note that "upstream" in the context of the 5' end of the gene means out from the gene body.
`end3`	A numeric vector of length 2 specifying the number of bp upstream and downstream of transcription end sites for which to create gene 3' end ranges. Given as c(upstream,downstream). Note that "upstream" in the context of the 3' end of the gene means into the gene body.
`genome`	The UCSC name specific to the genome of the query coordinates (e.g. "hg19", "hg18", "mm10", etc)
`cachedir`	A path to a directory where a local cache of UCSC tables are stored. If equal to `NULL` (default), the data will be downloaded to temporary files and loaded on the fly. Caching is highly recommended to save time and bandwidth.
`sync`	If TRUE, then check if newer versions of UCSC tables are available and download them if so. If FALSE, skip this check. Can be used to freeze data versions in an analysis-specific cachedir for reproducibility.

Value

A list containing one GenomicRanges object for each of the gene model portions: Promoters, 3' Ends, Exons, Introns, Intergenic, 5' UTRs, 3' UTRs. The "srow" column can be used to match individual ranges to individual genes in the table given to the "genes" argument by row number.

getGenes — *Load table of gene ranges via UCSC Genome Browser tables*

Description

Load table of gene ranges via UCSC Genome Browser tables

Usage

```
getGenes(geneset = "ucsc", genome, cachedir = NULL, sync = TRUE)
```

Arguments

geneset	Select one of "ucsc" for the UCSC Genes (from the knownGene table), "refseq" for RefSeq genes (from the refFlat table), or "ensembl" for the Ensembl genes (from the ensGene table)
genome	UCSC genome name to use (e.g. hg19, mm10)
cachedir	Path where cached UCSC tables are stores
sync	If TRUE, then check if newer versions of UCSC tables are available and download them if so. If FALSE, skip this check. Can be used to freeze data versions in an analysis-specific cachedir for reproducibility.

getUCSCTable — *Load an annotation table from the UCSC Genome Browser as an R data.table*

Description

If only table and genome are given, the function will load the data directly into the R workspace. If cachedir is a path to a directory, this directory will be used to maintain a cache of UCSC tables so they do not need to be re-downloaded on each call. If the data already exists and sync=TRUE, the function will only re-download and re-extract if the modified dates are different between the cachedir and remote copies. Note that start coordinates in these raw data tables are 0-based, whereas the Goldmine annotation functions will convert these to be 1-based.

Usage

```
getUCSCTable(table, genome, cachedir = NULL, version = "latest",
  sync = TRUE, url = "http://hgdownload.cse.ucsc.edu/goldenPath/",
  fread = TRUE)
```

Arguments

table	The UCSC string specific for the table to sync (e.g. "knownGene", "kgXref", etc)
genome	The UCSC string specific to the genome to be downloaded (e.g. "hg19", "hg19", "mm10", etc)

cachedir	A path to a directory where a cachedir cache of UCSC tables are stored. If equal to NULL (default), the data will be downloaded to temporary files and loaded on the fly.
version	If "latest" (default) then use the newest version of the table available. If set to a timestamp string of an archived table (format: YYYY-MM-DD-HH-MM-SS), then load this specific version. Obtain these strings by examining the file names under your cache directory. An archive file with a date stamp is saved automatically with each download of a new version. This feature only works if you have a cachedir cache that contains the desired versions.
sync	If TRUE, then check if a newer version is available and download if it is. If FALSE, skip this check. Only has an effect if a cachedir cache directory (cachedir) is given.
url	The root of the remote http URL to download UCSC data from (set by default to http://hgdownload.cse.ucsc.edu/goldenPath/)

Value

A data.frame or data.table of the desired UCSC table.

`gmWrite` *Write individual CSV files to disk from the output of goldmine()*

Description

Write a CSV file for each output table in a goldmine() output list object. Useful for importing to spreadsheet applications. Will save the "context", "genes", and any tables in "features" to individual CSV files.

Usage

```
gmWrite(gm, path = ".")
```

Arguments

gm	The output list object from goldmine().
path	The directory to write the files into (default: current working directory).

`goldmine` *Explore relationships between a set of genomic ranges and known genes/features*

Description

Computes the overlap between a query set of genomic ranges given as a GenomicRanges, data.frame, or data.table with gene and feature sets of interest. Reports both summarized overlaps (same number of rows as the query - a "wide format") and in separate tables, individual overlap events (one row for each pair of overlapping query and gene/feature item - a "long format" similar to an inner join).

Usage

```
goldmine(query, genes = getGenes(geneset = "ucsc", genome = genome, cachedir =
  cachedir), features = list(), promoter = c(1000, 500), end3 = c(1000,
  1000), genome, cachedir, sync = TRUE)
```

Arguments

query	A GenomicRanges, data.frame, or data.table of regions to annotate. If a data.frame or data.table, must contain the columns "chr", "start", "end", where the "start" coordinates are 1-based. All additional columns will be retained in the output object.
genes	Genes of interest from the output table of getGenes(). If not given, will default to the UCSC knownGene table.
features	A list() of GenomicRanges, data.table, or data.frame objects giving feature sets of interest. If a data.frame or data.table, must contain the columns "chr", "start", "end", where the "start" coordinates are 1-based. All additional columns will be retained in the output object. See also the getFeatures() function.
promoter	A numeric vector of length 2 specifying the number of bp upstream and downstream of transcription start sites for which to create promoter ranges. Given as c(upstream,downstream). Note that "upstream" in the context of the 5' end of the gene means out from the gene body.
end3	A numeric vector of length 2 specifying the number of bp upstream and downstream of transcription end sites for which to create gene 3' end ranges. Given as c(upstream,downstream). Note that "upstream" in the context of the 3' end of the gene means into the gene body.
genome	The UCSC name specific to the genome of the query coordinates (e.g. "hg19", "hg18", "mm10", etc)
cachedir	A path to a directory where a local cache of UCSC tables are stored. If equal to NULL (default), the data will be downloaded to temporary files and loaded on the fly. Caching is highly recommended to save time and bandwidth.
sync	If TRUE, then check if newer versions of UCSC tables are available and download them if so. If FALSE, skip this check. Can be used to freeze data versions in an analysis-specific cachedir for reproducibility.

Value

A list: "context" shows a percent overlap for each range in the query set with gene model regions and each feature set ("wide" format - same number of rows as the query and in the same order), "genes" contains a detailed view of each query region overlap with individual gene isoforms ("long" format - a row for each pair of query and isoform overlaps), "features" is a list of tables which for each table given in the "features" argument which contain a row for each instance of a query region overlapping with a feature region (also "long" format).

makeDT — *Make a data.table from a GRanges or a data.frame*

Description

Given a data.frame or GRanges, a data.table object will be created. If the input is already a data.table, it is simply returned.

Usage

```
makeDT(obj)
```

Arguments

obj	A data.frame or GRanges

Value

A data.table made from the data in obj.

makeGRanges — *Make a GRanges from a data.frame or data.table with the fields "chr", "start", and "end"*

Description

Given a data.frame or data.table with the columns "chr", "start", and "end", a GenomicRanges (GRanges) object will be created. All other columns will be passed on as metadata. If the input is already a GRanges, it is simply returned. If the column "strand" exists, it will be set as the strand.

Usage

```
makeGRanges(obj, strand = F)
```

Arguments

obj	A data.frame or data.table with columns "chr", "start", and "end" and any other columns
strand	Use the information in the "strand" column to set strand in the GRanges, if it is present.

Value

A GRanges made from the data in obj.

`readUCSCAnnotation` *Read UCSC table files from disk and join all related tables*

Description

Read UCSC table files from disk and join all related tables

Usage

```
readUCSCAnnotation(genome = "hg19", path = "")
```

`sortDT` *Sort a data.frame, data.table, or GRanges by chr (accounting for mixed string and numeric names), start, end and return a data.table*

Description

Sort a data.frame, data.table, or GRanges by chr (accounting for mixed string and numeric names), start, end and return a data.table

Usage

```
sortDT(obj)
```

Arguments

`obj`	A data.frame, data.table, or GRanges

`sortGRanges` *Sort a data.frame, data.table, or GRanges by chr (accounting for mixed string and numeric names), start, end and return a GRanges*

Description

Sort a data.frame, data.table, or GRanges by chr (accounting for mixed string and numeric names), start, end and return a GRanges

Usage

```
sortGRanges(obj)
```

Arguments

`obj`	A data.frame, data.table, or GRanges

testEnrichment *Test enrichment between a query set and a null set*

Description

Provide both a query set of ranges (as a GenomicRanges, data.frame, or data.table) and a null set of ranges (same format options). Counts for each feature will be computed in both the query and null sets, and tested for significance of difference using binom.test().

Usage

```
testEnrichment(query, null, features)
```

Arguments

query	A data.frame or data.table with columns "chr", "start", and "end" and any other columns. If a data.frame or data.table, must contain the columns "chr", "start", "end", where the "start" coordinates are 1-based.
null	A data.frame or data.table with columns "chr", "start", and "end" and any other columns. If a data.frame or data.table, must contain the columns "chr", "start", "end", where the "start" coordinates are 1-based.
features	A data.frame or data.table with columns "chr", "start", and "end" and any other columns. If a data.frame or data.table, must contain the columns "chr", "start", "end", where the "start" coordinates are 1-based. Additionally, there must be a column named "name" which will be used as a factor to divide the ranges into subsets. Each subset will be tested for enrichment individually.

Value

A table reporting enrichment results for each factor given in the "name" column in features.

writeBEDFromGRanges *Write a BED format file from a GenomicRanges object*

Description

Creates BED file suitable for upload as a custom track to the UCSC genome browser. Note that start coordinates are 0-based in the BED format.

Usage

```
writeBEDFromGRanges(gr, file, name = NULL)
```

Arguments

gr	A GenomicRanges object.
file	Filename of the BED file to write.
name	Column name to use for the name field in the BED file (optional)

Index