Package ‘motifbreakR’

January 21, 2017

Title A Package For Predicting The Disruptiveness Of Single Nucleotide Polymorphisms On Transcription Factor Binding Sites

Version 1.4.0

Author Simon Gert Coetzee [aut, cre] Dennis J. Hazelett [aut]

Maintainer Simon Gert Coetzee <Simon.Coetzee@cshs.org>

Description We introduce motifbreakR, which allows the biologist to judge in the first place whether the sequence surrounding the polymorphism is a good match, and in the second place how much information is gained or lost in one allele of the polymorphism relative to another. MotifbreakR is both flexible and extensible over previous offerings; giving a choice of algorithms for interrogation of genomes with motifs from public sources that users can choose from; these are 1) a weighted-sum probability matrix, 2) log-probabilities, and 3) weighted by relative entropy. MotifbreakR can predict effects for novel or previously described variants in public databases, making it suitable for tasks beyond the scope of its original design. Lastly, it can be used to interrogate any genome curated within Bioconductor (currently there are 22).

Depends R (>= 3.2), grid, MotifDb

Imports methods, compiler, grDevices, grImport, stringr, BiocGenerics, S4Vectors (>= 0.9.25), IRanges, GenomeInfoDb, GenomicRanges, Biostrings, BSgenome, rtracklayer, VariantAnnotation, BiocParallel, motifStack, Gviz, matrixStats, TFMPvalue

Suggests BSgenome.Hsapiens.UCSC.hg19, SNPloc.Hsapiens.dbSNP.20120608, SNPloc.Hsapiens.dbSNP142.GRCh37, knitr, rmarkdown, BSgenome.Drerio.UCSC.danRer7, BiocStyle

VignetteBuilder knitr

License GPL-2

LazyData true

BugReports https://github.com/Simon-Coetzee/motifbreakR/issues

biocViews ChIPSeq, Visualization, MotifAnnotation

NeedsCompilation no
calculatePvalue

**Description**

Calculate the significance of the matches for the reference and alternate alleles for the PWM.

**Usage**

```r
calculatePvalue(results, background = c(A = 0.25, C = 0.25, G = 0.25, T = 0.25))
```

**Arguments**

- `results`: The output of `motifbreakR` that was run with `filterp=TRUE`
- `background`: Numeric Vector; the background probabilities of the nucleotides

**Details**

This function is intended to be used on a selection of results produced by `motifbreakR`, and this can be (although not always) a very memory and time intensive process if the algorithm doesn’t converge rapidly.

**Value**

A GRanges object. The same Granges object that was input as results, but with `Refpvalue` and `Altpvalue` columns in the output modified from NA to the p-value calculated by `TFMsc2pv`.

**Source**

See Also
See `TFMsc2pv` from the `TFMPvalue` package for information about how the p-values are calculated.

Examples

```r
data(example.results)
rs2661839 <- example.results[names(example.results) %in% "rs2661839"]
rs2661839 <- calculatePvalue(rs2661839)
```

Description
From the abstract: "Recent advances in technology have led to a dramatic increase in the number of available transcription factor ChIP-seq and ChIP-chip data sets. Understanding the motif content of these data sets is an important step in understanding the underlying mechanisms of regulation. Here we provide a systematic motif analysis for 427 human ChIP-seq data sets using motifs curated from the literature and also discovered de novo using five established motif discovery tools. We use a systematic pipeline for calculating motif enrichment in each data set, providing a principled way for choosing between motif variants found in the literature and for flagging potentially problematic data sets. Our analysis confirms the known specificity of 41 of the 56 analyzed factor groups and reveals motifs of potential cofactors. We also use cell type-specific binding to find factors active in specific conditions. The resource we provide is accessible both for browsing a small number of factors and for performing large-scale systematic analyses. We provide motif matrices, instances and enrichments in each of the ENCODE data sets. The motifs discovered here have been used in parallel studies to validate the specificity of antibodies, understand cooperativity between data sets and measure the variation of motif binding across individuals and species."

Usage
encodemotif

Format

`MotifDb` object of length 2064; to access metadata use `mcols(encodemotif)`

- `providerName` Name provided by ENCODE
- `providerId` Same as `providerName`
- `dataSource` "ENCODE-motif"
- `geneSymbol` Gene symbol for the transcription factor
- `geneId` Entrez gene id for the transcription factor
- `geneIdType` "ENTREZ"
- `proteinId` UNIPROT id for the transcription factor
- `proteinIdType` "UNIPROT"
- `organism` "Hsapiens"
- `sequenceCount` NA not available
bindingSequence Consensus sequence for the motif
bindingDomain NA incomplete
tfFamily NA incomplete

experimentType occurs in two forms:
For motifs that were discovered in this study, the format is `cellType_source-LabMetadata:MotifFinder#Location`
for example `H1-hESC_encode-Myers_seq_hsa_v041610.2_r1:MEME#2#Intergenic`.
For motifs that were "known" the format tends to be `TF_source_sourceId` for example `AP1_jaspar_MA0099.2`.

pubmedID "24335146" see Source for more details

Details
Load with `data(encodemotif)`

Value `MotifList-class` object

Source

See Also
`http://compbio.mit.edu/encode-motifs/`

Examples
`data(encodemotif)`
`encodemotif`
**Examples**

```r
data(example.results)
example.results
```

---

**factorbook**  
*MotifDb object containing motif information from around the genomic regions bound by 119 human transcription factors in Factorbook.*

---

**Description**

From the abstract: "Chromatin immunoprecipitation coupled with high-throughput sequencing (ChIP-seq) has become the dominant technique for mapping transcription factor (TF) binding regions genome-wide. We performed an integrative analysis centered around 457 ChIP-seq data sets on 119 human TFs generated by the ENCODE Consortium. We identified highly enriched sequence motifs in most data sets, revealing new motifs and validating known ones. The motif sites (TF binding sites) are highly conserved evolutionarily and show distinct footprints upon DNase I digestion. We frequently detected secondary motifs in addition to the canonical motifs of the TFs, indicating tethered binding and cobinding between multiple TFs. We observed significant position and orientation preferences between many cobinding TFs. Genes specifically expressed in a cell line are often associated with a greater occurrence of nearby TF binding in that cell line. We observed cell-line-specific secondary motifs that mediate the binding of the histone deacetylase HDAC2 and the enhancer-binding protein EP300. TF binding sites are located in GC-rich, nucleosome-depleted, and DNase I sensitive regions, flanked by well-positioned nucleosomes, and many of these features show cell type specificity. The GC-richness may be beneficial for regulating TF binding because, when unoccupied by a TF, these regions are occupied by nucleosomes in vivo. We present the results of our analysis in a TF-centric web repository Factorbook (http://factorbook.org) and will continually update this repository as more ENCODE data are generated."

---

**Usage**

```r
factorbook
```

---

**Format**

*MotifDb object of length 79; to access metadata use mcols(factorbook)*

- **providerName**: Name listed in meme output of "Supp TableS2.pdf" for the citation indicated below
- **providerId**: Same as providerName
- **dataSource**: "FactorBook"
- **geneSymbol**: NA these motifs don’t have a direct 1 to 1 relationship with a transcription factor
- **geneId**: NA
- **geneIdType**: NA
- **proteinId**: NA
- **proteinIdType**: NA
- **organism**: "Hsapiens"
- **sequenceCount**: NA
- **bindingSequence**: Consensus sequence for the motif
- **bindingDomain**: NA
tfFamily  NA
experimentType  NA
pubmedID  "22955990" see Source for more details

Details

Load with data(factorbook)

Value

MotifList-class object

Source


See Also

http://factorbook.org

Examples

data(factorbook)
factorbook

hocomoco  MotifDb object containing motif information from Homo Sapiens Comprehensive Model Collection (HOCOMOCO) of transcription factor (TF) binding models

Description

From the abstract: "We present the Homo sapiens comprehensive model collection (HOCOMOCO, http://autosome.ru/HOCOMOCO/, http://cbrc.kaust.edu.sa/hocomoco/) containing carefully hand-curated TFBS models constructed by integration of binding sequences obtained by both low- and high-throughput methods. To construct position weight matrices to represent these TFBS models, we used ChIPMunk software in four computational modes, including newly developed periodic positional prior mode associated with DNA helix pitch. We selected only one TFBS model per TF, unless there was a clear experimental evidence for two rather distinct TFBS models. We assigned a quality rating to each model. HOCOMOCO contains 426 systematically curated TFBS models for 401 human TFs, where 172 models are based on more than one data source."

Usage

hocomoco
Format

MotifDb object of length 426; to access metadata use mcols(hocomoco)

providerName Name provided by HOCOMOCO
providerId ID provided by HOCOMOCO including experiment type
dataSource "HOCOMOCO"
geneSymbol Gene symbol for the transcription factor
geneId Entrez gene id for the transcription factor
geneIdType "ENTREZ"
proteinId UNIPROT id for the transcription factor
proteinIdType "UNIPROT"
organism "Hsapiens"
sequenceCount Number of sequences evaluated for producing the PWM
bindingSequence Consensus sequence for the motif
bindingDomain NA incomplete
tfFamily NA incomplete

experimentType from http://autosome.ru/HOCOMOCO/Details.php#200 quoted here:

"TFBS model identification modes
To construct TFBS models ChIPMunk was run four times: two times (f1) and (f2) with uniform model positional prior and two times (si) and (do) with informative model positional prior.
The min-to-max (f1) model length estimation mode was used with the min length of 7 bp and increasing it by 1 bp until the default max length of 25 bp was reached following the optimal length selection procedure as in Kulakovskiy and Makeev, Biophysics, 2009. For max-to-min (f2) model length estimation mode we started from 25 bp and searched for the best alignment decreasing the length by 1 bp until the minimal length of 7 bp. We also used the single (si) and double box (do) model positional priors in order to simulate DNA helix turn. For a single box, the positional weights are to be distributed as \( \cos(\pi n / T) \), where \( T=10.5 \) is the DNA helix pitch, \( n \) is the coordinate within the alignment, and the center of the alignment of the length \( L \) is at \( n=0 \). During the internal cycle of PWM optimization the PWM column scores are multiplied by prior values so the columns closer to the center of the alignment (\( n=0 \)) receive no score penalty while the columns around (\( n = 5,6,-5,-6 \) ) contribute much less to the score of the PWM under optimization. The single box model prior was used along with the min-to-max length estimation mode (si). We also used the double box model prior with a shape prior equal to \( \sin(\pi n / T) \), which was used to search for possibly longer double box models in the max-to-min length estimation mode (do).

Model quality assignment
The resulting models were rated (from A to F) according to their quality. Model quality rates from A-to-D were assigned to proteins known to be TFs, including those listed in Schaefer et al., Nucleic Acids Research, 2011 with addition of a number of proteins having relevant models and sufficient evidence to be TFs. The ratings were assigned by human curation according to the following criteria:

Relevant distribution of position-specific information content over alignment columns, which means a model LOGO representation displaying well formed core positions with a high information content surrounded by flanking letters with lower information content; the information content at flanking positions decreasing with the distance from the model core.

"Stability", which means that in more than one of the ChIPMunk modes we obtained models with a similar length, consensus, and comparable number of aligned binding sites, along with
a similar shape of model LOGO representation. "Similarity" of the model to the binding sequence consensus for this TF given in the UniProt or other databases, which means similarity of the shape of the model LOGO and TFBS lengths to those of other TFs from the same TF family. "A total number of binding sites" was also considered as a quality measure, as a large set of binding regions (mostly but not limited to ChIP-Seq and parallel SELEX) implies that there are many observations of each letter in any position of the alignment, particularly many observations of non-consensus letters in core positions. In positions with low information content, where there is no strong consensus, all variants have many observations, and thus the observed letter frequencies are less dependent on statistical fluctuations.

Quality A was assigned to high confidence models complying with all four criteria listed in the section above. Quality B was assigned to models built from large sequence sets that failed no more than one out of the three remaining criteria. Quality C was assigned to models built from small sequence sets but (with a number of specifically marked exceptions) complying with the three remaining criteria. Quality D models missed part of the known consensus sequence or had no clearly significant core positions in the TFBS model. Quality E (error) was assigned to models for proteins not convincingly shown to be TFs or to models exhibiting an irrelevant LOGO shape or a wrong consensus sequence. Quality F (failure) was assigned to TFs for which there was no reliable model identified.

**pubmedID** "23175603" see **Source** for more details

**Details**

Load with `data(hocomoco)`

**Value**

`MotifList-class` object

**Source**


**See Also**


**Examples**

```r
data(hocomoco)
hocomoco```

---

**homer** `MotifDb object containing motif information from motif databases included in HOMER.`
**Description**

From the website: "Homer includes several motif databases that are used to help annotate results and conduct searches for known motifs. HOMER contains a custom motif database based on independent analysis of mostly ChIP-Seq data sets which is heavily utilized in the software." See [http://homer.salk.edu/homer/motif/motifDatabase.html](http://homer.salk.edu/homer/motif/motifDatabase.html) for more information on how these files were generated, and Homer’s sources.

**Usage**

`homer`

**Format**

`MotifDb` object of length 247; to access metadata use `mcols(homer)`

- `providerName` Name provided HOMER
- `providerId` Factor Name provided by HOMER
- `dataSource` "HOMER"
- `geneSymbol` Symbol provided by HOMER
- `geneId` Entrez gene id for the transcription factor
- `geneIdType` "ENTREZ"
- `proteinId` UNIPROT id for the transcription factor
- `proteinIdType` "UNIPROT"
- `organism` "Hsapiens"
- `sequenceCount` NA
- `bindingSequence` Consensus sequence for the motif
- `bindingDomain` DBD provided by HOMER
- `tfFamily` NA
- `experimentType` The Celltype, IP, Assay, and GEO id if applicable for the motif
- `pubmedID` "20513432" see Source for more details

**Details**

Load with `data(homer)`

**Value**

`MotifList-class` object

**Source**


**See Also**

[http://homer.salk.edu/homer/index.html](http://homer.salk.edu/homer/index.html) [http://homer.salk.edu/homer/motif/motifDatabase.html](http://homer.salk.edu/homer/motif/motifDatabase.html) [http://homer.salk.edu/homer/motif/HomerMotifDB/homerResults.html](http://homer.salk.edu/homer/motif/HomerMotifDB/homerResults.html)
Examples

data(homer)
homer

motifbreakR

Description

Predict The Disruptiveness Of Single Nucleotide Polymorphisms On Transcription Factor Binding Sites.

Usage

motifbreakR(snpList, pwmList, threshold = 0.85, filterp = FALSE, method = "default", show.neutral = FALSE, verbose = FALSE, bkg = c(A = 0.25, C = 0.25, G = 0.25, T = 0.25), BPPARAM = bpparam())

Arguments

snpList
The output of snps.from.rsid or snps.from.file

pwmList
An object of class MotifList containing the motifs that you wish to interrogate

threshold
Numeric; the maximum p-value for a match to be called or a minimum score threshold

filterp
Logical; filter by p-value instead of by pct score.

method
Character; one of default, log, ic, or notrans; see details.

show.neutral
Logical; include neutral changes in the output

verbose
Logical; if running serially, show verbose messages

bkg
Numeric Vector; the background probabilities of the nucleotides used with method=log

BPPARAM
a BiocParallel object see register and see getClass("BiocParallelParam") for additional parameter classes. Try BiocParallel::registered() to see what's available and for example BiocParallel::bpparam("SerialParam") would allow serial evaluation.

Details

motifbreakR works with position probability matrices (PPM). PPM are derived as the fractional occurrence of nucleotides A,C,G, and T at each position of a position frequency matrix (PFM). PFM are simply the tally of each nucleotide at each position across a set of aligned sequences. With a PPM, one can generate probabilities based on the genome, or more practically, create any number of position specific scoring matrices (PSSM) based on the principle that the PPM contains information about the likelihood of observing a particular nucleotide at a particular position of a true transcription factor binding site. What follows is a discussion of the three different algorithms that may be employed in calls to the motifbreakR function via the method argument.

Suppose we have a frequency matrix $M$ of width $n$ (i.e. a PPM as described above). Furthermore, we have a sequence $s$ also of length $n$, such that $s_i \in \{A, T, C, G\}, i = 1, \ldots n$. Each column of $M$ contains the frequencies of each letter in each position.
Commonly in the literature, sequences are scored as the sum of log probabilities:

**Equation 1**

\[ F(s, M) = \sum_{i=1}^{n} \log\left( \frac{M_{s,i}}{b_{s_i}} \right) \]

where \(b_{s_i}\) is the background frequency of letter \(s_i\) in the genome of interest. This method can be specified by the user as `method='log'`.

As an alternative to this method, we introduced a scoring method to directly weight the score by the importance of the position within the match sequence. This method of weighting is accessed by specifying `method='ic'` (information content). A general representation of this scoring method is given by:

**Equation 2**

\[ F(s, M) = p_s \cdot \omega_M \]

where \(p_s\) is the scoring vector derived from sequence \(s\) and matrix \(M\), and \(\omega_M\) is a weight vector derived from \(M\). First, we compute the scoring vector of position scores \(p\)

**Equation 3**

\[ p_s = (M_{s,i}) \quad \text{where} \quad \frac{i = 1, \ldots, n}{s_i \in \{A, C, G, T\}} \]

and second, for each \(M\) a constant vector of weights \(\omega_M = (\omega_1, \omega_2, \ldots, \omega_n)\).

There are two methods for producing \(\omega_M\). The first, which we call weighted sum, is the difference in the probabilities for the two letters of the polymorphism (or variant), i.e. \(\Delta p_{s,i}\), or the difference of the maximum and minimum values for each column of \(M\):

**Equation 4.1**

\[ \omega_i = \max\{M_i\} - \min\{M_i\} \quad \text{where} \quad i = 1, \ldots, n \]

The second variation of this theme is to weight by relative entropy. Thus the relative entropy weight for each column \(i\) of the matrix is given by:

**Equation 4.2**

\[ \omega_i = \sum_{j \in \{A, C, G, T\}} M_{j,i} \log_2\left( \frac{M_{j,i}}{b_i} \right) \quad \text{where} \quad i = 1, \ldots, n \]

where \(b_i\) is again the background frequency of the letter \(i\).

Thus, there are 3 possible algorithms to apply via the `method` argument. The first is the standard summation of log probabilities (`method='log'`). The second and third are the weighted sum and information content methods (`method='default'` and `method='ic'`) specified by equations 4.1 and 4.2, respectively. **motifbreakR** assumes a uniform background nucleotide distribution \((b)\) in equations 1 and 4.2 unless otherwise specified by the user. Since we are primarily interested in the difference between alleles, background frequency is not a major factor, although it can change the results. Additionally, inclusion of background frequency introduces potential bias when collections of motifs are employed, since motifs are themselves unbalanced with respect to nucleotide composition. With these cautions in mind, users may override the uniform distribution if so desired. For all
three methods, *motifbreakR* scores and reports the reference and alternate alleles of the sequence \( F(s_{\text{REF}}, M) \) and \( F(s_{\text{ALT}}, M) \), and provides the matrix scores \( p_{s_{\text{REF}}} \) and \( p_{s_{\text{ALT}}} \) of the SNP (or variant). The scores are scaled as a fraction of scoring range 0-1 of the motif matrix, \( M \). If either of \( F(s_{\text{REF}}, M) \) and \( F(s_{\text{ALT}}, M) \) is greater than a user-specified threshold (default value of 0.85) the SNP is reported. By default *motifbreakR* does not display neutral effects, \( \Delta p_i < 0.4 \) but this behaviour can be overridden.

Additionally, now, with the use of TFMPvalue-package, we may filter by p-value of the match. This is unfortunately a two step process. First, by invoking `filterp=TRUE` and setting a threshold at a desired p-value e.g 1e-4, we perform a rough filter on the results by rounding all values in the PWM to two decimal place, and calculating a scoring threshold based upon that. The second step is to use the function `calculatePvalue()` on a selection of results which will change the `Refpvalue` and `Altpvalue` columns in the output from NA to the p-value calculated by TFMsc2pv. This can be (although not always) a very memory and time intensive process if the algorithm doesn’t converge rapidly.

**Value**

a GRanges object containing:

- **REF** the reference allele for the SNP
- **ALT** the alternate allele for the SNP
- **snpPos** the coordinates of the SNP
- **motifPos** the coordinates of the SNP within the TF binding motif
- **geneSymbol** the geneSymbol corresponding to the TF of the TF binding motif
- **dataSource** the source of the TF binding motif
- **providerName, providerId** the name and id provided by the source
- **seqMatch** the sequence on the 5’ -> 3’ direction of the "+" strand that corresponds to DNA at the position that the TF binding motif was found.
- **pctRef** The score as determined by the scoring method, when the sequence contains the reference SNP allele, normalized to a scale from 0 - 1. If `filterp = FALSE`, this is the value that is thresholded.
- **pctAlt** The score as determined by the scoring method, when the sequence contains the alternate SNP allele, normalized to a scale from 0 - 1. If `filterp = FALSE`, this is the value that is thresholded.
- **scoreRef** The score as determined by the scoring method, when the sequence contains the reference SNP allele
- **scoreAlt** The score as determined by the scoring method, when the sequence contains the alternate SNP allele
- **Refpvalue** p-value for the match for the `pctRef` score, initially set to NA. see `calculatePvalue` for more information
- **Altpvalue** p-value for the match for the `pctAlt` score, initially set to NA. see `calculatePvalue` for more information
- **alleleRef** The proportional frequency of the reference allele at position `motifPos` in the motif
- **alleleAlt** The proportional frequency of the alternate allele at position `motifPos` in the motif
- **effect** one of weak, strong, or neutral indicating the strength of the effect.

each SNP in this object may be plotted with `plotMB`
motifbreakR_motif

See Also

See `snps.from.rsid` and `snps.from.file` for information about how to generate the input to this function and `plotMB` for information on how to visualize its output.

Examples

```r
library(BSgenome.Hsapiens.UCSC.hg19)
# prepare variants
load(system.file("extdata", "pca.enhancer.snps.rda", package = "motifbreakR")) # loads snps.mb
pca.enhancer.snps <- sample(snps.mb, 20)
# Get motifs to interrogate
data(hocomoco)
motifs <- sample(hocomoco, 50)
# run motifbreakR
results <- motifbreakR(pca.enhancer.snps, motifs, threshold = 0.85, method = "ic", BPPARAM = BiocParallel::SerialParam())
```

motifbreakR_motif

MotifDb object containing motif information from the motif databases of HOCOMOCO, Homer, FactorBook and ENCODE.

Description

This object contains all the MotifList-class objects that were generated for this package. See the individual help sections for hocomoco, homer, factorbook, and ensemblmotif, for how the data is formatted.

Usage

motifbreakR_motif

Format

MotifDb object of length 2816; to access metadata use mcols(motifbreakR_motif)

Details

Load with data(motifbreakR_motif)

Value

MotifList-class object
Source


See Also

hocomoco, homer, factorbook, and encodemotif

Examples

data(motifbreakR_motif)
motifbreakR_motif

plotMB(results, rsid, reverseMotif = TRUE, effect = c("strong", "weak"))

Description

Plot a genomic region surrounding a genomic variant, and potentially disrupted motifs

Usage

plotMB(results, rsid, reverseMotif = TRUE, effect = c("strong", "weak"))

Arguments

results The output of motifbreakR
rsid Character; the identifier of the variant to be visualized
reverseMotif Logical; if the motif is on the "+" strand show the the motifs as reversed FALSE or reverse complement TRUE
effect Character; show motifs that are strongly effected c("strong"), weakly effected c("weak"), or both c("strong", "weak")

Details

plotMB produces output showing the location of the SNP on the chromosome, the surrounding sequence of the + strand, the footprint of any motif that is disrupted by the SNP or SNV, and the DNA sequence motif(s)
Value

plots a figure representing the results of motifbreakR at the location of a single SNP, returns invisible NULL.

See Also

See motifbreakR for the function that produces output to be visualized here, also snps.from.rsid and snps.from.file for information about how to generate the input to motifbreakR function.

Examples

data(example.results)
example.results
## Not run:
plotMB(example.results, "rs2661839", effect = "strong")
## End(Not run)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>file</td>
<td>Character; a character containing the path to a bed file or a vcf file see Details for a description of the required format</td>
</tr>
<tr>
<td>dbSNP</td>
<td>OPTIONAL; an object of class SNPlocs to lookup rsids; see available.SNPs in injectSNPs to check for available SNPlocs</td>
</tr>
<tr>
<td>search.genome</td>
<td>an object of class BSgenome for the species you are interrogating; see available.genomes for a list of species</td>
</tr>
<tr>
<td>format</td>
<td>Character; one of bed or vcf</td>
</tr>
</tbody>
</table>

Details

snps.from.file takes a character vector describing the file path to a bed file that contains the necessary information to generate the input for motifbreakR see http://www.genome.ucsc.edu/FAQ/FAQformat.html#format1 for a complete description of the BED format. Our convention deviates in that there is a required format for the name field. name is defined as chromosome:start:REF:ALT or the rsid from dbSNP (if you’ve included the optional SNPlocs argument). For example if you were to include rs123 in it’s alternate format it would be entered as chr7:24966446:C:A
Import SNPs from rsid for use in motifbreakR

Description
Import SNPs from rsid for use in motifbreakR

Usage

```
snps.from.rsid(rsid = NULL, dbSNP = NULL, search.genome = NULL)
```

Arguments

- **rsid**: Character; a character vector of rsid values from dbSNP
- **dbSNP**: an object of class SNPlocs to lookup rsids; see availableSNPs in injectSNPs to check for available SNPlocs
- **search.genome**: an object of class BSgenome for the species you are interrogating; see available.genomes for a list of species

Details

`snps.from.rsid` take an rsid, or character vector of rsids and generates the required object to input into motifbreakR
Value

a GRanges object containing:

SNP_id           The rsid of the SNP with the "rs" portion stripped
alleles_as_ambig THE IUPAC ambiguity code between the reference and alternate allele for this
                  SNP
REF              The reference allele for the SNP
ALT              The alternate allele for the SNP

See Also

See motifbreakR for analysis; See snps.from.file for an alternate method for generating a list
of variants.

Examples

library(BSgenome.Hsapiens.UCSC.hg19)
library(SNPlocs.Hsapiens.dbSNP.20120608)
snps.file <- system.file("extdata", "pca.enhancer.snps", package = "motifbreakR")
snps <- as.character(read.table(snps.file)[,1])
snps.mb <- snps.from.rsid(snps,
                          dbSNP = SNPlocs.Hsapiens.dbSNP.20120608,
                          search.genome = BSgenome.Hsapiens.UCSC.hg19)
Index

*Topic datasets
  encodemotif, 3
  example.results, 4
  factorbook, 5
  hocomoco, 6
  homer, 8
  motifbreakR_motif, 13
available.genomes, 15, 16
calculatePvalue, 2, 12
encodemotif, 3, 13, 14
example.results, 4
factorbook, 5, 13, 14
GRanges, 4
hocomoco, 6, 13, 14
homer, 8, 13, 14
injectSNPs, 15, 16
motifbreakR, 2, 4, 10, 15–17
motifbreakR_motif, 13
MotifDb, 3, 5, 7, 9, 13
plotMB, 12, 13, 14
register, 10
snps.from.file, 13, 15, 15, 17
snps.from.rsid, 13, 15, 16, 16
TFMsc2pv, 2, 3, 12