Package 'GenomicScores'

November 19, 2017

Type Package

Title Infrastructure to work with genomewide position-specific scores

Description Provide infrastructure to store and access genomewide position-specific scores within R and Bioconductor.

Version 1.2.0

License Artistic-2.0

Depends R (>= 3.4), S4Vectors (>= 0.7.21), GenomicRanges, methods, BiocGenerics (>= 0.13.8)

Imports utils, XML, Biobase, IRanges (>= 2.3.23), BSgenome, GenomeInfoDb, AnnotationHub

Suggests BiocStyle, knitr, rmarkdown, BSgenome.Hsapiens.UCSC.hg19, phastCons100way.UCSC.hg19, MafDb.1Kgenomes.phase1.hs37d5, SNPlocs.Hsapiens.dbSNP144.GRCh37, VariantAnnotation, TxDb.Hsapiens.UCSC.hg19.knownGene, gwascat

VignetteBuilder knitr

URL https://github.com/rcastelo/GenomicScores

BugReports https://github.com/rcastelo/GenomicScores/issues

Encoding UTF-8

biocViews Infrastructure, Genetics, Annotation, Sequencing, Coverage

NeedsCompilation no

Author Robert Castelo [aut, cre], Pau Puigdevall [ctb]

Maintainer Robert Castelo <robert.castelo@upf.edu>

R topics documented:

GScores-class *GScores objects*

Description

The goal of the GenomicScores package is to provide support to store and retrieve genomic scores associated to physical nucleotide positions along a genome. This is achieved through the GScores class of objects, which is a container for genomic score values.

Details

The GScores class attempts to provide a compact storage and efficient retrieval of genomic score values that have been typically processed and stored using some form of lossy compression. This class is currently based on a former version of the SNPlocs class defined in the BSgenome package, with the following slots:

provider (character), the data provider such as UCSC.

provider_version (character), the version of the data as given by the data provider, typically a date in some compact format.

download_url (character), the URL of the data provider from where the original data were downloaded.

download_date (character), the date on which the data were downloaded.

reference_genome (GenomeDescription), object with information about the reference genome whose physical positions have the genomic scores.

data_pkgname (character), name given to the set of genomic scores associated to a particular genome. When the genomic scores are stored within an annotation package, then this corresponds to the name of that package.

data_dirpath (character), absolute path to the local directory where the genomic scores are stored in one file per genome sequence.

data_serialized_objnames (character), named vector of filenames pointing to files containing the genomic scores in one file per genome sequence. The names of this vector correspond to the genome sequence names.

.data_cache (environment), data structure where objects storing genomic scores are cached into main memory.

The goal of the design behind the GScores class is to load into main memory only the objects associated with the queried sequences to minimize the memory footprint, which may be advantageous in workflows that parallelize the access to genomic scores by genome sequence.

GScores objects are created either from AnnotationHub resources or when loading specific annotation packages that store genomic score values. Two such annotation packages are:

phastCons100way.UCSC.hg19 Nucleotide-level phastCons conservation scores from the UCSC Genome Browser calculated from multiple genome alignments from the human genome version hg19 to 99 vertebrate species.

phastCons100way.UCSC.hg38 Nucleotide-level phastCons conservation scores from the UCSC Genome Browser calculated from multiple genome alignments from the human genome version hg38 to 99 vertebrate species.

Constructor

`GScores(provider, provider_version, download_url, download_date, reference_genome, data_pk`
Creates a `GScores` object. In principle, the end-user needs not to call this function.

`provider` character, containing the data provider.

`provider_version` character, containing the version of the data as given by the data provider.

`download_url` character, containing the URL of the data provider from where the original data were downloaded.

`reference_genome` GenomeDescription, storing the information about the associated reference genome.

`data_pkgname` character, name given to the set of genomic scores stored through this object.

`data_dirpath` character, absolute path to the local directory where the genomic scores are stored.

`data_serialized_objname` character vector, containing filenames where the genomic scores are stored.

Accessors

`name(x)`: get the name of the set of genomic scores.

`type(x)`: get the substring of the name of the set of genomic scores comprised between the first character until the first period. This should typically match the type of genomic scores such as, `phastCons`, `phyloP`, etc.

`provider(x)`: get the data provider.

`providerVersion(x)`: get the provider version.

`organism(x)`: get the organism associated with the genomic scores.

`referenceGenome(x)`: get the `GenomeDescription` object associated with the genome on which the genomic scores are defined.

`seqlevelsStyle(x)`: get the genome sequence style.

`seqinfo(x)`: get the genome sequence information.

`seqnames(x)`: get the genome sequence names.

`seqlengths(x)`: get the genome sequence lengths.

`qfun(x)`: get the quantizer function.

`dqfun(x)`: get the dequantizer function.

`citation(x)`: get citation information for the genomic scores data in the form of a `bibentry` object.

Author(s)

R. Castelo

See Also

`scores()` `phastCons100way.UCSC.hg19` `phastCons100way.UCSC.hg38`

Examples

```
## supporting annotation packages with genomic scores
if (require(phastCons100way.UCSC.hg19)) {
  library(GenomicRanges)

  gsco <- phastCons100way.UCSC.hg19
  gsco
  scores(gsco, GRanges(seqnames="chr7", IRanges(start=117232380, width=5)))
}

## supporting AnnotationHub resources
## Not run:
availableGScores()
gsco <- getGScores("phastCons100way.UCSC.hg19")
gsco
scores(gsco, GRanges(seqnames="chr7", IRanges(start=117232380, width=5)))

## End(Not run)

## meta data from a GScores object
name(gsco)
type(gsco)
provider(gsco)
providerVersion(gsco)
organism(gsco)
referenceGenome(gsco)
seqlevelsStyle(gsco)
seqinfo(gsco)
head(seqnames(gsco))
head(seqlengths(gsco))
qfun(gsco)
dqfun(gsco)
citation(gsco)
```

`MafDb-class` *MafDb class*

Description

Class for annotation packages storing minor allele frequency data.

Usage

```
## S4 method for signature 'MafDb'
mafByOverlaps(x, ranges, pop="AF", type=c("snvs", "nonsnvs"), caching=TRUE)
## S4 method for signature 'MafDb'
mafById(x, ids, pop, caching)
## S4 method for signature 'MafDb'
populations(x)
```

Arguments

`x`	A `MafDb` object.

ranges	Either a GRanges object, a GPos object or a character string vector with the format "CHR:START[-END]".
ids	A character string vector with variant identifiers annotated by the MAF data source, typically dbSNP 'rs' identifiers. Note that the mapping of these identifiers to genomic positions and MAF values might be a subset of the most up to date dbSNP 'rs' identifier assignment to variants. To access the latter, please use the snpsById() method from the BSgenome package with the desired SNPlocs.* package.
pop	Character string vector with the populations for which we want to retrieve MAF values.
type	Character string setting the type of variant to seek, which can be either 'snvs' (default) when we seek single nucleotide variants or 'nonsnvs', otherwise.
caching	logical; TRUE (default) indicates that the function stores into main memeory the MAF data as it gets loaded from disk, improving performance; FALSE forces this function to load MAF data from disk each time, decreasing performance and memory requirements.

Details

The MafDb class is derived from the GScores class and it serves the purpose of providing support to store and access minor allele frequency (MAF) data from R and Bioconductor. Two annotation packages using the MafDb class are:

MafDb.1Kgenomes.phase1.hs37d5	MAF values from the 1000 Genomes Project Phase 1.
MafDb.1Kgenomes.phase3.hs37d5	MAF values from the 1000 Genomes Project Phase 3.

This object class tries to reduce the disk space required to store MAF values for millions of SNPs by coding their double-precision values, which range between 0 and 1, into a single-byte raw object type. To achieve this, the original MAF values are rounded to one significant digit for AF < 0.1 and two significant digits for AF >= 0.1. When a variant has multiple alternate alleles, only the largest MAF value is stored.

Author(s)

R. Castelo

Examples

```
## Not run:
  ## lookup allele frequencies for rs1129038, a SNP associated to blue and brown eye colors
  ## as reported by Eiberg et al. Blue eye color in humans may be caused by a perfectly associated
  ## founder mutation in a regulatory element located within the HERC2 gene inhibiting OCA2 expression.
   ## Human Genetics, 123(2):177-87, 2008 [http://www.ncbi.nlm.nih.gov/pubmed/18172690]

  if (require(MafDb.1Kgenomes.phase1.hs37d5)) {
    mafdb <- MafDb.1Kgenomes.phase1.hs37d5
    mafdb

    ## specialized interface
    populations(mafdb)
```

```
  rng <- GRanges("15", IRanges(28356859, 28356859))
  mafByOverlaps(mafdb, rng)
  mafByOverlaps(mafdb, "15:28356859-28356859")
  mafByOverlaps(mafdb, "15:28356859")
  mafById(mafdb, "rs1129038")
}

## End(Not run)
```

---

scores *Accessing genomic scores*

Description

Functions to access genomic scores through GScores objects.

Usage

```
availableGSscores()
getGScores(x)
## S4 method for signature 'GScores,GenomicRanges'
scores(object, ranges, ...)
```

Arguments

x	A character vector of length 1 specifiying the genomic scores resource to fetch. The function availableGScores() shows the available genomic scores resources.
object	A GScores object.
ranges	A GenomicRanges object with positions from where to retrieve genomic scores.
...	In the call to the scores() method one can additionally set the following arguments: • scores.onlyFlag set to FALSE (default) when scores are return in a metadata column called scores from the input GenomicRanges object. When set to TRUE, the only the numeric vector of scores is returned. • summaryFunFunction to summarize genomic scores when more than one position is retrieved. By default, this is set to the arithmetic mean, i.e., the mean() function. • quantizedFlag setting whether the genomic scores should be returned quantized (TRUE) or dequantized (FALSE, default). • cachingFlag setting whether genomic scores per chromosome should be kept cached in memory (TRUE, default) or not (FALSE). The latter option minimizes the memory footprint but slows down the performance when the scores() method is called multiple times.

Details

The method scores() takes as first argument a GScores-class object that can be loaded from an annotation package or from an AnnotationHub resource. These two possibilities are illustrated in the examples below.

Value

The function `availableGScores()` returns a character vector with the names of the AnnotationHub resources corresponding to different available sets of genomic scores. The function `getGScores()` return a `GScores` object. The method `scores()` returns a numeric vector.

Author(s)

R. Castelo

See Also

`phastCons100way.UCSC.hg19` `phastCons100way.UCSC.hg38`

Examples

```
## accessing genomic scores from an annotation package
if (require(phastCons100way.UCSC.hg19)) {
  library(GenomicRanges)

  gsco <- phastCons100way.UCSC.hg19
  gsco
  scores(gsco, GRanges(seqnames="chr7", IRanges(start=117232380, width=5)))
}

## accessing genomic scores from AnnotationHub resources
## Not run:
availableGScores()
gsco <- getGScores("phastCons100way.UCSC.hg19")
scores(gsco, GRanges(seqnames="chr7", IRanges(start=117232380, width=5)))

## End(Not run)
```

Index