# Package 'GenomicScores'

November 12, 2026

**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** 2.24.0

**License** Artistic-2.0

**Depends** R (>= 3.5), S4Vectors (>= 0.7.21), GenomicRanges, methods, BiocGenerics (>= 0.13.8)

**Imports** stats, utils, XML, httr, Biobase, BiocManager, BiocFileCache, IRanges (>= 2.3.23), Biostrings, Seqinfo, GenomeInfoDb (>= 1.45.5), AnnotationHub, rhdf5, DelayedArray, HDF5Array

**Suggests** RUnit, BiocStyle, knitr, rmarkdown, VariantAnnotation, gwascat, RColorBrewer, shiny, shinyjs, shinycustomloader, data.table, DT, magrittr, shinydashboard, BSgenome.Hsapiens.UCSC.hg38, phastCons100way.UCSC.hg38, MafDb.1Kgenomes.phase1.hs37d5, MafH5.gnomAD.v4.0.GRCh38, SNPlocs.Hsapiens.dbSNP144.GRCh37, TxDb.Hsapiens.UCSC.hg38.knownGene

**VignetteBuilder** knitr

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

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

**Encoding** UTF-8

**biocViews** Infrastructure, Genetics, Annotation, Sequencing, Coverage, AnnotationHubSoftware

**git_url** https://git.bioconductor.org/packages/GenomicScores

**git_branch** RELEASE_3_23

**git_last_commit** dce2b3e

**git_last_commit_date** 2026-04-28

**Repository** Bioconductor 3.23

**Date/Publication** 2026-11-11

**Author** Robert Castelo [aut, cre],
Pau Puigdevall [ctb],
Pablo Rodríguez [ctb]

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

# Contents

availableGScores . . . . . 2
GenomicScores-defunct . . . . . 4
GenomicScores-deprecated . . . . . 4
gscores . . . . . 5
GScores-class . . . . . 7
igscores . . . . . 11
makeGScoresPackage . . . . . 12
rgscores . . . . . 13
wgscores . . . . . 15

**Index** **17**

---

`availableGScores` *Exploring genomic scores resources*

---

## Description

Functions to explore genomic scores resources.

## Usage

```
availableGScores(use.internet=FALSE)
getGScores(x, accept.license=FALSE)
```

## Arguments

| | |
|---|---|
| `x` | A `character` vector of length 1 specifiying the genomic scores resource to fetch. |
| `use.internet` | A logical value specifying whether we want to check through the internet whether the annotation packages and `AnnotationHub` resources are ready to be downloaded. By default, i.e. `use.internet=FALSE`, this is not checked to speed up the execution of this function. |
| `accept.license` | A logical value specifying whether we accept the license under which we can use the requested data, when the data is distributed under such a license. |

## Details

The function `availableGScores()` shows genomic score sets available as `AnnotationHub` online resources.

## Value

The function `availableGScores()` returns a `data.frame` object with a row for each available resource of genomic scores and the following four columns:

- `Name`: Name of the Bioconductor package or `AnnotationHub` resource.
- `Organism`: Organism on which the genomic scores are defined.
- `Category`: Category to which the genomic scores belong to.
- `Installed`: Whether the resource is installed as a package (`TRUE`) or not (`FALSE`).
- `Cached`: Whether the resource is available within the local cache of the `AnnotationHub` (`TRUE`) or not (`FALSE`).
- `BiocManagerInstall`: Whether the resource can be installed as an annotation package through `BiocManager::install()` (`TRUE`) or not (`FALSE`).
- `AnnotationHub`: Whether the resource can be downloaded as a `GScores` object through the `AnnotationHub`, using the function `getGScores()` (`TRUE`), or not (`FALSE`).

The function `getGScores()` returns a `GScores` object.

## Author(s)

R. Castelo

## References

Puigdevall, P. and Castelo, R. GenomicScores: seamless access to genomewide position-specific scores from R and Bioconductor. *Bioinformatics*, 18:3208-3210, 2018.

## See Also

`getGScores()` `phastCons100way.UCSC.hg38` `MafDb.1Kgenomes.phase1.hs37d5`

## Examples

```
availableGScores()

## Not run:
gsco <- getGScores("cadd.v1.6.hg38")

## End(Not run)
```

## GenomicScores-defunct *Defunct methods in package 'GenomicScores'*

### Description

Defunct classes and methods in 'GenomicScores'.

### Details

The following classes are defunct; use the replacement indicated below:

- `MafDb`: The `MafDb` class has been replaced by the `GScores-class` class.

The following methods are defunct; use the replacement indicated below:

- `scores()`: The `scores()` method has been replaced by the `gscores()` and `score()` methods.
- `mafByOverlaps()`: The `mafByOverlaps()` method has been replaced by the `gscores()` and `score()` methods.
- `mafById()`: The `mafById()` method has been replaced by the `gscores()` and `score()` methods.

### Author(s)

Robert Castelo <robert.castelo@upf.edu>

## GenomicScores-deprecated *Deprecated methods in package 'GenomicScores'*

### Description

There are currently no deprecated classes or methods in 'GenomicScores'.

### Author(s)

Robert Castelo <robert.castelo@upf.edu>

gscores *Accessing genomic gscores*

## Description

Functions to access genomic gscores through GScores objects.

## Usage

```
## S4 method for signature 'GScores,GenomicRanges'
gscores(x, ranges, ...)
## S4 method for signature 'GScores,character'
gscores(x, ranges, ...)
## S4 method for signature 'GScores'
score(x, ..., simplify=TRUE)
```

## Arguments

| | |
|---|---|
| x | A GScores object. |
| ranges | A GenomicRanges object with positions from where to retrieve genomic scores, or a character string vector with identifiers associated by the data producer with the genomic scores, e.g., dbSNP 'rs' identifiers in the case of some MafDb.* packages. |
| ... | In the call to the gscores() method one can additionally specify the following optional parameters: |

- pop: Character string vector specifying the scores populations to query, when there is more than one. By default, its value is 'defaultPopulation(x)'. Use populations() to find out the available scores populations.
- type: Character string specifying the type of genomic position being sought, which can be a single nucleotide range (snr), by default, or a nonsnr spanning multiple nucleotides. The latter is the case of indel variants in minor allele frequency data.
- scores.only: Flag setting whether only the scores should be returned as a numeric vector (TRUE), instead of returning them as a metadata column in a GRanges object (FALSE, default).
- summaryFun: Function 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.
- quantized: Flag setting whether the genomic scores should be returned quantized (TRUE) or dequantized (FALSE, default).
- ref: Vector of reference alleles in the form of either a character vector, a DNAStringSet object or a DNAStringSetList object. This argument is used only when either there are multiple scores per position or x is a MafDb.* package.

- `alt`: Vector of alternative alleles in the form of either a character vector, a `DNAStringSet` object or a `DNAStringSetList` object. This argument is used only when either there are multiple scores per position or `x` is a `MafDb.*` package.
- `minoverlap`: Integer value passed internally to the function `findOverlaps()` from the `IRanges` package, when querying genomic positions associated with multiple-nucleotide ranges (nonSNRs). By default, `minoverlap=1L`, which assumes that the sought nonSNRs are stored as in `VCF` files, using the nucleotide composition of the reference sequence. This argument is only relevant for genomic scores associated with nonSNRs.
- `caching`: Flag 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 `gscores()` method is called multiple times.

`simplify` Flag setting whether the result should be simplified to a vector (`TRUE`, default) if possible. This happens when scores from a single population are queried.

## Details

The method `gscores()` takes as first argument a `GScores` object, previouly loaded from either an annotation package or an `AnnotationHub` resource; see `getGScores()`.

The arguments `ref` and `alt` serve two purposes. One, when there are multiple scores per position, such as with CADD or M-CAP, and we want to select a score matching a specific combination of reference and alternate alleles. The other purpose is when the `GScores` object `x` is a `MafDb.*` package, then by providing `ref` and `alt` alelles we will get separate frequencies for reference and alternate alleles. The current lossy compression of these values yields a correct assignment for biallelic variants in the corresponding `MafDb.*` package and an approximation for multiallelic ones.

## Value

The method `gscores()` returns a `GRanges` object with the genomic scores in a metadata column called `score`. The method `score()` returns a numeric vector with the genomic scores.

## Author(s)

R. Castelo

## References

Puigdevall, P. and Castelo, R. GenomicScores: seamless access to genomewide position-specific scores from R and Bioconductor. *Bioinformatics*, 18:3208-3210, 2018.

## See Also

`phastCons100way.UCSC.hg38` `MafDb.1Kgenomes.phase1.hs37d5`

## Examples

```
## one genomic range of width 5
gr1 <- GRanges(seqnames="chr22", IRanges(start=50528591, width=5))
gr1

## five genomic ranges of width 1
gr2 <- GRanges(seqnames="chr22", IRanges(start=50528591:50528596, width=1))
gr2

## accessing genomic gscores from an annotation package
if (require(phastCons100way.UCSC.hg38)) {
  library(GenomicRanges)

  gsco <- phastCons100way.UCSC.hg38
  gsco
  gscores(gsco, gr1)
  score(gsco, gr1)
  gscores(gsco, gr2)
  populations(gsco)
  gscores(gsco, gr2, pop="DP2")
}

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

  ## lookup allele frequencies for SNP rs1129038, located at 15:28356859, 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]
  gscores(mafdb, GRanges("15:28356859"), pop=populations(mafdb))
  gscores(mafdb, "rs1129038", pop=populations(mafdb))
}
```

---

# 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_group (character), name denoting a category of genomic scores to which the scores stored in the object belong to. Typical values are "Conservation", "MAF", "Pathogenicity", etc.

data_tag (character), name identifying the genomic scores stored in the object and which can be used, for instance, to assign a column name storing these scores.

data_pops (character), vector of character strings storing score population names. The term "default" is reserved to denote a score set that is not associated to a particular population name and is used by default.

data_nonsnrs (logical), flag indicating whether the object stores genomic scores associated with non-single nucleotide ranges.

data_nsites (integer), number of sites in the genome associated with the genomic scores stored in the object.

.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_pkgname, data_`
Creates a `GScores` object. In principle, the end-user needs not to call this function.

`provider` character string, containing the data provider.

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

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

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

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

`default_pop` character string, containing the name of the default scores population.

`data_group` character string, containing a name that indicates a category of genomic scores to which the scores in the object belong to. Typical names could be "Conservation", "MAF", etc.

`data_tag` character string, containing a tag that succintly labels genomic scores from a particular source. This can be used to automatically give, for instance, a name to a column storing genomic scores in data frame object. Its default value takes the prefix of the package name.

## 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.

`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.

`populations(x)`: get the identifiers of the available scores populations. If only one scores population is available, then it shows only the term `default`.

`defaultPopulation(x)`: get or set the default population of scores.

`gscoresCategory(x)`: get or set the genomic scores category label.

`gscoresTag(x)`: get or set the genomic scores tag label.

`gscoresNonSNRs(x)`: get whether there are genomic scores associated with non-single nucleotide ranges.

`nsites(x)`: get the number of sites in the genome with genomic scores.

`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

## References

Puigdevall, P. and Castelo, R. GenomicScores: seamless access to genomewide position-specific scores from R and Bioconductor. *Bioinformatics*, 18:3208-3210, 2018.

## See Also

`gscores()` `score()` `phastCons100way.UCSC.hg38`

## Examples

```
## one genomic range of width 5
gr1 <- GRanges(seqnames="chr22", IRanges(start=50528591, width=5))
gr1

## five genomic ranges of width 1
gr2 <- GRanges(seqnames="chr22", IRanges(start=50528591:50528596, width=1))
gr2

## supporting annotation packages with genomic scores
if (require(phastCons100way.UCSC.hg38)) {
  library(GenomicRanges)

  phast <- phastCons100way.UCSC.hg38
  phast
  gscores(phast, gr1)
  score(phast, gr1)
  gscores(phast, gr2)
  populations(phast)
  gscores(phast, gr2, pop="DP2")
}

## supporting AnnotationHub resources
## Not run:
availableGScores()
phast <- getGScores("phastCons100way.UCSC.hg38")
phast
gscores(phast, gr1)
```

```
## End(Not run)

## metadata from a GScores object
name(phast)
type(phast)
provider(phast)
providerVersion(phast)
organism(phast)
seqlevelsStyle(phast)
seqinfo(phast)
head(seqnames(phast))
head(seqlengths(phast))
gscoresTag(phast)
populations(phast)
defaultPopulation(phast)
qfun(phast)
dqfun(phast)
citation(phast)
```

---

`igscores` *GenomicScores shiny app*

---

## Description

Starts an interactive GenomicScores shiny web app.

## Usage

```
igscores()
```

## Details

The goal of the `GenomicScores` package is to provide support to store and retrieve genomic scores associated to physical nucleotide positions along a genome.

The `igscores()` function starts an interactive shiny web app that allows the user to query annotation packages storing genomic scores. Internally, it calls to the function `gscores()`; see its manual page for a description of the arguments and their default and alternative values.

## Value

None.

## Author(s)

P. Rodríguez and R. Castelo

## References

Puigdevall, P. and Castelo, R. GenomicScores: seamless access to genomewide position-specific scores from R and Bioconductor. *Bioinformatics*, 18:3208-3210, 2018.

## See Also

gscores

## Examples

```
## Not run:
igscores() ## this will open your browser with the GenomicScores shiny web app

## End(Not run)
```

---

# makeGScoresPackage *Building genomic gscores packages*

---

## Description

Build a genomic gscores packages from a GScores object.

## Usage

```
makeGScoresPackage(gsco, version, maintainer, author,
                   destDir=".", license="Artistic-2.0")
```

## Arguments

| | |
|---|---|
| gsco | GScores. The GScores object from which the package will be created. |
| version | Character. Version of the package. |
| maintainer | Character. Maintainer of the package, including email. |
| author | Character. Author of the package. |
| destDir | Character. The path to the directory where the package will be created. |
| license | Character. The license of the package. |

## Details

This function allows one to create an R package from a GScores object. This may be useful if one wants to have a tar-ball package version of genomic scores available only through the AnnotationHub; see the vignette.

## Value

It returns invisibily the package directory.

### Author(s)

R. Castelo

### References

Puigdevall, P. and Castelo, R. GenomicScores: seamless access to genomewide position-specific scores from R and Bioconductor. *Bioinformatics*, 18:3208-3210, 2018.

### See Also

`availableGScores()` `getGScores()`

### Examples

```
## accessing genomic scores from AnnotationHub resources
## and building a package from them
## Not run:
availableGScores()
gsco <- getGScores("fitCons.UCSC.hg19")
makeGScoresPackage(gsco, version="1.0", maintainer="me <me@example.com>", author="me")

## End(Not run)
```

---

## rgscores *Sampling genomic gscores*

---

### Description

Function for randomly sampling genomic gscores from `GScores` objects.

### Usage

```
## S4 method for signature 'GScores,missing'
rgscores(n, object, ...)
## S4 method for signature 'missing,GScores'
rgscores(n, object, ...)
## S4 method for signature 'numeric,GScores'
rgscores(n, object, ...)
## S4 method for signature 'integer,GScores'
rgscores(n, object, ...)
```

### Arguments

| | |
|---|---|
| `n` | Number of scores to sample. |
| `object` | A `GScores` object. |
| `...` | In the call to the `rgscores()` method one can additionally set the following arguments: |

- pop Character string vector of length one, specifying the scores population from which we want to sample scores. By default, its value is `defaultPopulation(object)`. Use `populations()` to find out the available scores populations.
- `scores.only` Flag setting whether only the scores should be returned as a numeric vector (`TRUE`), instead of returning them as a metadata column in a `GRanges` object (`FALSE`, default).
- `ranges` Either a `GRanges` object or a character string vector of sequence names. Scores will be sampled from the given genomic regions, which by default correspond to the entire genomic space of `object`. Currently, only entire chromosomes are considered.

## Details

The method `rgscores()` samples scores randomly from a `GScores` object.

## Value

A `GRanges` object with the sampled genomic positions and scores. When `scores.only=TRUE` then a numeric vector is returned with the sampled scores.

## Author(s)

R. Castelo

## References

Puigdevall, P. and Castelo, R. GenomicScores: seamless access to genomewide position-specific scores from R and Bioconductor. *Bioinformatics*, 18:3208-3210, 2018.

## See Also

`phastCons100way.UCSC.hg38` `MafDb.1Kgenomes.phase1.hs37d5`

## Examples

```
## accessing genomic gscores from an annotation package
if (require(phastCons100way.UCSC.hg38)) {
  library(GenomicRanges)

  phast <- phastCons100way.UCSC.hg38
  set.seed(123)
  rgscores(10L, phast, ranges=c("chr22", "chrY"))
}

if (require(MafDb.1Kgenomes.phase1.hs37d5)) {
  mafdb <- MafDb.1Kgenomes.phase1.hs37d5
  set.seed(123)
  rgscores(10L, mafdb, ranges=c("21", "22"))
}
```

wgscores *Discovering genomic gscores*

## Description

Functions to discover which genomic scores are present in given genomic ranges through GScores objects.

## Usage

```
## S4 method for signature 'GScores,GenomicRanges'
wgscores(x, ranges, ...)
```

## Arguments

| | |
|---|---|
| x | A GScores object. |
| ranges | A GenomicRanges object with genomic ranges where to search for genomic scores. |
| ... | In the call to the wgscores() method one can additionally set the following arguments:<br>• pop: Character string vector specifying the scores populations to query, when there is more than one. By default, its value is 'defaultPopulation(x)'. Use populations() to find out the available scores populations.<br>• type: Character string specifying the type of genomic position being sought, which can be a single nucleotide range (snr), by default, or a nonsnr spanning multiple nucleotides. The latter is the case of indel variants in minor allele frequency data.<br>• caching: Flag 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 wgscores() method is called multiple times. |

## Details

The method wgscores() takes as first argument a GScores object, previouly loaded from either an annotation package or an AnnotationHub resource; see getGScores() and a GenomicRanges object as a second argument. It will search for which genomic scores fall within the provided genomic ranges and return them in an ordered GenomicRanges object with the scores in the metadata columns.

## Value

The method wgscores() returns a GRanges object with the genomic scores in metadata columns named after the corresponding score population name.

## Author(s)

R. Castelo

## References

Puigdevall, P. and Castelo, R. GenomicScores: seamless access to genomewide position-specific scores from R and Bioconductor. *Bioinformatics*, 18:3208-3210, 2018.

## See Also

MafDb.1Kgenomes.phase1.hs37d5

## Examples

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

  ## lookup allele frequencies for SNP rs1129038, located at 15:28356859 in
  ## GRCh37, a SNP associated to blue and brown eye colors as reported by
  ## Eiberg et al. (Human Genetics, 2008; http://www.ncbi.nlm.nih.gov/pubmed/18172690).
  ## 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.
  ##
  ## for the sake of illustrating this functionality let's create a
  ## GenomicRanges object with the SNP rs1129038 and enlarge its range
  ## by 200nt at each flank.
  snp <- GRanges("15:28356859")
  rngsnp <- flank(snp, width=100, both=TRUE)
  width(rngsnp)

  ## now use this genomic range to search for the rs1129038 SNP
  wgscores(mafdb, rngsnp)

  ## let's illustrate this same functionality for INDELs, starting
  ## from the specific INDEL rs113993960 that leads to the loss of
  ## phenylalanine at amino acid position 508 of the CFTR protein,
  ## commonly referred to as F508del in the CFTR gene, which is
  ## concretely a deletion of four nucleotides at position
  ## 7:117199644 in GRCh37 and enlarge its range by 20nt on each flank.
  indel <- GRanges(seqnames="chr7",
                   ranges=IRanges(start=117199644, width=4))
  rngindel <- flank(indel, width=20, both=TRUE)
  width(rngindel)

  ## now use this genomic range to search for the rs113993960 INDEL
  wgscores(mafdb, rngindel, type="nonsnrs")
}
```

# Index


∗ **GenomicScores**
  igscores, 11
∗ **internal**
  GenomicScores-defunct, 4
  GenomicScores-deprecated, 4
∗ **methods**
  GScores-class, 7
∗ **misc**
  GenomicScores-defunct, 4
  GenomicScores-deprecated, 4
∗ **shiny**
  igscores, 11
∗ **utilities**
  availableGScores, 2
  gscores, 5
  makeGScoresPackage, 12
  rgscores, 13
  wgscores, 15

availableGScores, 2, *13*

citation *(*GScores-class*)*, 7
citation,character-method *(*GScores-class*)*, 7
citation,GScores-method *(*GScores-class*)*, 7
citation,missing-method *(*GScores-class*)*, 7
class:GScores *(*GScores-class*)*, 7

defaultPopulation *(*GScores-class*)*, 7
defaultPopulation,GScores-method *(*GScores-class*)*, 7
defaultPopulation<- *(*GScores-class*)*, 7
defaultPopulation<-,GScores,character-method *(*GScores-class*)*, 7
dqfun *(*GScores-class*)*, 7
dqfun,GScores-method *(*GScores-class*)*, 7

GenomicScores *(*GScores-class*)*, 7
GenomicScores-defunct, 4
GenomicScores-deprecated, 4
getGScores, *3*, *6*, *13*, *15*
getGScores *(*availableGScores*)*, 2
GScores *(*GScores-class*)*, 7
gscores, *4*, 5, *10–12*
gscores,GScores,character-method *(*gscores*)*, 5
gscores,GScores,GenomicRanges-method *(*gscores*)*, 5
GScores-class, 7
gscoresCategory *(*GScores-class*)*, 7
gscoresCategory,GScores-method *(*GScores-class*)*, 7
gscoresCategory<- *(*GScores-class*)*, 7
gscoresCategory<-,GScores,character-method *(*GScores-class*)*, 7
gscoresNonSNRs *(*GScores-class*)*, 7
gscoresNonSNRs,GScores-method *(*GScores-class*)*, 7
gscoresTag *(*GScores-class*)*, 7
gscoresTag,GScores-method *(*GScores-class*)*, 7
gscoresTag<- *(*GScores-class*)*, 7
gscoresTag<-,GScores,character-method *(*GScores-class*)*, 7

igscores, 11

MafDb.1Kgenomes.phase1.hs37d5, *3*, *6*, *14*, *16*
makeGScoresPackage, 12

name *(*GScores-class*)*, 7
name,GScores-method *(*GScores-class*)*, 7
nsites *(*GScores-class*)*, 7
nsites,GScores-method *(*GScores-class*)*, 7

organism,GScores-method *(*GScores-class*)*, 7

phastCons100way.UCSC.hg38, *3*, *6*, *10*, *14*
populations, *5*, *14, 15*
populations *(*GScores-class*)*, 7
populations,GScores-method
*(*GScores-class*)*, 7
provider,GScores-method
*(*GScores-class*)*, 7
providerVersion,GScores-method
*(*GScores-class*)*, 7

qfun *(*GScores-class*)*, 7
qfun,GScores-method *(*GScores-class*)*, 7

rgscores, 13
rgscores,GScores,missing-method
*(*rgscores*)*, 13
rgscores,integer,GScores-method
*(*rgscores*)*, 13
rgscores,missing,GScores-method
*(*rgscores*)*, 13
rgscores,numeric,GScores-method
*(*rgscores*)*, 13

score, *4*, *10*
score,GScores-method *(*gscores*)*, 5
scores *(*GenomicScores-defunct*)*, 4
scores,GScores,GenomicRanges-method
*(*GenomicScores-defunct*)*, 4
seqinfo,GScores-method *(*GScores-class*)*,
7
seqlengths,GScores-method
*(*GScores-class*)*, 7
seqlevelsStyle,GScores-method
*(*GScores-class*)*, 7
seqnames,GScores-method
*(*GScores-class*)*, 7
show,GScores-method *(*GScores-class*)*, 7

type *(*GScores-class*)*, 7
type,GScores-method *(*GScores-class*)*, 7

wgscores, 15
wgscores,GScores,GenomicRanges-method
*(*wgscores*)*, 15