# Package ‘GenomicDistributionsData’

May 7, 2026

**Title** Reference data for GenomicDistributions package

**Version** 1.21.0

**Description** This package provides ready to use reference data for GenomicDistributions package. Raw data was obtained from ensembldb and processed with helper functions. Data files are available for the following genome assemblies: hg19, hg38, mm9 and mm10.

**License** BSD_2_clause + file LICENSE

**biocViews** ExperimentHub, ExperimentData, Genome

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.1.1

**Depends** R (>= 4.1)

**Imports** ExperimentHub (>= 1.14.0), AnnotationHub, AnnotationFilter, data.table, utils, BSgenome, GenomeInfoDb, GenomicFeatures, GenomicRanges, ensembldb

**Suggests** knitr, BiocStyle, rmarkdown

**VignetteBuilder** knitr

**BugReports** `http://github.com/databio/GenomicDistributionsData`

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

**git_branch** devel

**git_last_commit** e024c0a

**git_last_commit_date** 2026-04-28

**Repository** Bioconductor 3.24

**Date/Publication** 2026-05-07

**Author** Michal Stolarczyk [aut],
Kristyna Kupkova [aut, cre],
Jose Verdezoto [aut],
John Lawson [ctb],
Jason Smith [ctb],
Nathan C. Sheffield [ctb]

**Maintainer** Kristyna Kupkova <`kristynakupkova@gmail.com`>

# Contents

---

`.requireAndReturn` *Checks to make sure a package object is installed, and if so, returns it. If the library is not installed, it issues a warning and returns NULL.*

---

### Description

Checks to make sure a package object is installed, and if so, returns it. If the library is not installed, it issues a warning and returns NULL.

### Usage

```
.requireAndReturn(BSgenomeString)
```

### Arguments

| | |
|---|---|
| `BSgenomeString` | A BSgenome compatible genome string. |

### Value

A BSgenome object if installed.

---

`buildChromSizes` *Build chromosome sizes object*

---

## Description

Build chromosome sizes object

## Usage

```
buildChromSizes(assembly)
```

## Arguments

| | |
|---|---|
| `assembly` | string, reference assembly identifier to chromosome sizes for |

## Value

named int, sizes of chromosome with respective names

## Examples

```
## Not run:
buildChromSizes("hg19")

## End(Not run)
```

---

`buildGeneModels` *Build gene models*

---

## Description

Build gene models

## Usage

```
buildGeneModels(refAssembly)
```

## Arguments

| | |
|---|---|
| `refAssembly` | string, reference assembly identifier to build gene model for |

## Value

a list of four GRanges objects: genesGR, exonsGR, threeUTRGR, fiveUTRGR

## Examples

```
## Not run:
hg19models = buildGeneModels("hg19")

## End(Not run)
```

---

## `buildOpenSignalMatrix` *Build open signal matrix*

---

### Description

Build open signal matrix

### Usage

```
buildOpenSignalMatrix(assembly)
```

### Arguments

| | |
|---|---|
| `assembly` | string, reference assembly identifier to build gene model for |

### Value

data.frame, rows represent whole selection of open chromatin regions across all cell types defined by ENCODE, columns are individual cell types and values are normalized open chromatin signal values

### Examples

```
## Not run:
buildOpenSignalMatrix("hg19")

## End(Not run)
```

---

## `buildTSS` *Build TSS*

---

### Description

Build TSS

### Usage

```
buildTSS(assembly)
```

### Arguments

| | |
|---|---|
| `assembly` | string, reference assembly identifier to TSS for |

### Value

GRanges object that consists of transcription start sites locations

### Examples

```
## Not run:
tss = buildTSS("hg19")

## End(Not run)
```

---

## `cellTypeMetadata` *Table the maps cell types to tissues and groups*

---

### Description

Table the maps cell types to tissues and groups

### Format

data.table with 3 columns (cellType, tissue and group) and 74 rows (one per cellType)

### Source

self-curated dataset

---

## `chromSizes_hg19` *hg19 chromosome sizes*

---

### Description

A dataset containing chromosome sizes for Homo sapiens hg19 genome assembly

### Usage

```
chromSizes_hg19(metadata=FALSE)
```

### Arguments

| | |
|---|---|
| `metadata` | `logical` value indicating whether only metadata should be returned or if the resource should be loaded |

### Format

A named vectors of lengths with one item per chromosome

### Value

A vector of lengths

### Source

BSgenome.Hsapiens.UCSC.hg19 package

### Examples

```
hg19c = chromSizes_hg19()
```

---

`chromSizes_hg38` *hg38 chromosome sizes*

---

### Description

A dataset containing chromosome sizes for Homo sapiens hg38 genome assembly

### Usage

```
chromSizes_hg38(metadata=FALSE)
```

### Arguments

| | |
|---|---|
| `metadata` | `logical` value indicating whether only metadata should be returned or if the resource should be loaded |

### Format

A named vectors of lengths with one item per chromosome

### Value

A vector of lengths

### Source

BSgenome.Hsapiens.UCSC.hg38 package

### Examples

```
hg38c = chromSizes_hg38()
```

---

`chromSizes_mm10` *mm10 chromosome sizes*

---

### Description

A dataset containing chromosome sizes for Mus musculus mm10 genome assembly

### Usage

```
chromSizes_mm10(metadata=FALSE)
```

### Arguments

| | |
|---|---|
| `metadata` | `logical` value indicating whether only metadata should be returned or if the resource should be loaded |

## Format

A named vectors of lengths with one item per chromosome

## Value

A vector of lengths

## Source

BSgenome.MMusculus.UCSC.mm10 package

## Examples

```
mm10c = chromSizes_mm10()
```

---

# `chromSizes_mm9` *mm9 chromosome sizes*

## Description

A dataset containing chromosome sizes for Mus musculus mm9 genome assembly

## Usage

```
chromSizes_mm9(metadata=FALSE)
```

## Arguments

| | |
|---|---|
| `metadata` | `logical` value indicating whether only metadata should be returned or if the resource should be loaded |

## Format

A named vectors of lengths with one item per chromosome

## Value

A vector of lengths

## Source

BSgenome.MMusculus.UCSC.mm9 package

## Examples

```
mm9c = chromSizes_mm9()
```

---

## `geneModels_hg19` *hg19 gene models*

---

### Description

A dataset containing gene models for Homo sapiens hg19 genome assembly

### Usage

```
geneModels_hg19(metadata=FALSE)
```

### Arguments

| | |
|---|---|
| `metadata` | `logical` value indicating whether only metadata should be returned or if the resource should be loaded |

### Format

A list of two GRanges objects, with genes and exons locations

### Value

A list with two GRanges objects.

### Source

EnsDb.Hsapiens.v75 package

### Examples

```
hg19GeneModels = geneModels_hg19()
```

---

## `geneModels_hg38` *hg38 gene models*

---

### Description

A dataset containing gene models for Homo sapiens hg38 genome assembly

### Usage

```
geneModels_hg38(metadata=FALSE)
```

### Arguments

| | |
|---|---|
| `metadata` | `logical` value indicating whether only metadata should be returned or if the resource should be loaded |

## Format

A list of two GRanges objects, with genes and exons locations

## Value

A list with two GRanges objects.

## Source

EnsDb.Hsapiens.v86 package

## Examples

```
hg38GeneModels = geneModels_hg38()
```

---

# `geneModels_mm10` *mm10 gene models*

## Description

A dataset containing gene models for Mus musculus mm10 genome assembly

## Usage

```
geneModels_mm10(metadata=FALSE)
```

## Arguments

| | |
|---|---|
| `metadata` | `logical` value indicating whether only metadata should be returned or if the resource should be loaded |

## Format

A list of two GRanges objects, with genes and exons locations

## Value

A list with two GRanges objects.

## Source

EnsDb.Mmusculus.v79 package

## Examples

```
mm10GeneModels = geneModels_mm10()
```

---

## `geneModels_mm9` *mm9 gene models*

---

### Description

A dataset containing gene models for Mus musculus mm9 genome assembly

### Usage

```
geneModels_mm9(metadata=FALSE)
```

### Arguments

| | |
|---|---|
| `metadata` | `logical` value indicating whether only metadata should be returned or if the resource should be loaded |

### Format

A list of two GRanges objects, with genes and exons locations

### Value

A list with two GRanges objects.

### Source

TxDb.Mmusculus.UCSC.mm9.knownGene package

### Examples

```
mm9GeneModels = geneModels_mm9()
```

---

## `GenomicDistributionsData` *GenomicDIstributionsData*

---

### Description

Data (hg19, hg38, mm9, mm10) for the GenomicDistributions package.

### Details

The vignette details how to access the data from ExperimentHub: browseVignettes("GenomicDistributionsData") Details on how the data files were created can be found under the R directory in utils.R and build.R

### Source

UCSC hg19, hg38, mm9, mm10 genome packages and ensembldb

## Examples

```
## Not run:
library(ExperimentHub)
hub = ExperimentHub()
q = query(hub, "GenomicDistributionsData")
q[[1]]
q[["EH3472"]]


## End(Not run)
```

---

## `loadBSgenome` *Loads BSgenome objects from UCSC-style character vectors.*

## Description

This function will let you use a simple character vector (e.g. 'hg19') to load and then return BSgenome objects. This lets you avoid having to use the more complex annotation for a complete BSgenome object (e.g. BSgenome.Hsapiens.UCSC.hg38.masked)

## Usage

```
loadBSgenome(genomeBuild, masked = TRUE)
```

## Arguments

| | |
|---|---|
| `genomeBuild` | One of 'hg19', 'hg38', 'mm10', 'mm9', or 'grch38' |
| `masked` | Should we used the masked version? Default:TRUE |

## Value

a BSgenome object according to the specified genome assembly

## Examples

```
## Not run:
bsg = loadBSgenome('hg19')

## End(Not run)
```

---

## `loadEnsDb` *Load selected EnsDb library*

---

### Description

Load selected EnsDb library

### Usage

```
loadEnsDb(genomeBuild)
```

### Arguments

| | |
|---|---|
| `genomeBuild` | string, genome identifier |

### Value

loaded library

### Examples

```
## Not run:
loadEnsDb("hg19")

## End(Not run)
```

---

## `loadTxDb` *Load selected TxDb library*

---

### Description

Load selected TxDb library

### Usage

```
loadTxDb(genomeBuild)
```

### Arguments

| | |
|---|---|
| `genomeBuild` | string, genome identifier |

### Value

loaded library

### Examples

```
## Not run:
loadTxDb("hg19")

## End(Not run)
```

## `openSignalMatrix_hg19` *A dataset containing open chromatin regions across all cell types defined by ENCODE for Homo Sapiens hg19*

### Description

Preparation steps:

1. made a universe of regions by merging regions across cell types defined as opened in ENCODE
2. took bigwig files from ENCODE for individual cell types, merged replicates, filtered out blacklisted sites
3. evaluated the signal above regions defined by previous step
4. performed quantile normalization

### Usage

```
openSignalMatrix_hg19(metadata=FALSE)
```

### Arguments

| | |
|---|---|
| `metadata` | `logical` value indicating whether only metadata should be returned or if the resource should be loaded |

### Format

data.frame, rows represent whole selection of open chromatin regions across all cell types defined by ENCODE, columns are individual cell types and values are normalized open chromatin signal values.

### Value

A data.frame with hg19 open chromatin regions.

### Source

`http://big.databio.org/open_chromatin_matrix/openSignalMatrix_hg19_percentile99_01_quantNormalized_round4d.txt.gz`

### Examples

```
## Not run:
hg19OpenSignal = openSignalMatrix_hg19()

## End(Not run)
```

---

`openSignalMatrix_hg38` *A dataset containing open chromatin regions across all cell types defined by ENCODE for Homo sapiens hg38*

---

## Description

Preparation steps:

1. made a universe of regions by merging regions across cell types defined as opened in ENCODE
2. took bigwig files from ENCODE for individual cell types, merged replicates, filtered out blacklisted sites
3. evaluated the signal above regions defined by previous step
4. performed quantile normalization

## Usage

```
openSignalMatrix_hg38(metadata=FALSE)
```

## Arguments

| | |
|---|---|
| `metadata` | `logical` value indicating whether only metadata should be returned or if the resource should be loaded |

## Format

data.frame, rows represent whole selection of open chromatin regions across all cell types defined by ENCODE, columns are individual cell types and values are normalized open chromatin signal values.

## Value

A data.frame with hg38 open chromatin regions.

## Source

`http://big.databio.org/open_chromatin_matrix/openSignalMatrix_hg38_percentile99_01_quantNormalized_round4d.txt.gz`

## Examples

```
## Not run:
hg38OpenSignal = openSignalMatrix_hg38()

## End(Not run)
```

---

## `openSignalMatrix_mm10` *A dataset containing open chromatin regions across all cell types defined by ENCODE for Mus musculus mm10*

---

### Description

Preparation steps:

1. made a universe of regions by merging regions across cell types defined as opened in ENCODE
2. took bigwig files from ENCODE for individual cell types, merged replicates, filtered out blacklisted sites
3. evaluated the signal above regions defined by previous step
4. performed quantile normalization

### Usage

```
openSignalMatrix_mm10(metadata=FALSE)
```

### Arguments

| | |
|---|---|
| `metadata` | `logical` value indicating whether only metadata should be returned or if the resource should be loaded |

### Format

data.frame, rows represent whole selection of open chromatin regions across all cell types defined by ENCODE, columns are individual cell types and values are normalized open chromatin signal values.

### Value

A data.frame with mm10 open chromatin regions.

### Source

`http://big.databio.org/open_chromatin_matrix/openSignalMatrix_mm10_percentile99_01_quantNormalized_round4d.txt.gz`

### Examples

```
## Not run:
mm10OpenSignal = openSignalMatrix_mm10()

## End(Not run)
```

---

## `TSS_hg19` *hg19 TSS locations*

---

### Description

A dataset containing Transcription Start Sites for Homo sapiens hg19 genome assembly

### Usage

```
TSS_hg19(metadata=FALSE)
```

### Arguments

| | |
|---|---|
| `metadata` | `logical` value indicating whether only metadata should be returned or if the resource should be loaded |

### Format

A GRanges object with Transcription Start Site locations

### Value

A GRanges object with hg19 TSS locations.

### Source

EnsDb.Hsapiens.v75 package

### Examples

```
hg19TSS = TSS_hg19()
```

---

## `TSS_hg38` *hg38 TSS locations*

---

### Description

A dataset containing Transcription Start Sites for the Homo sapiens hg38 genome assembly

### Usage

```
TSS_hg38(metadata=FALSE)
```

### Arguments

| | |
|---|---|
| `metadata` | `logical` value indicating whether only metadata should be returned or if the resource should be loaded |

## Format

A GRanges object with Transcription Start Site locations

## Value

A GRanges object with hg38 TSS locations.

## Source

EnsDb.Hsapiens.v86 package

## Examples

```
hg38TSS = TSS_hg38()
```

---

# `TSS_mm10` *mm10 TSS locations*

## Description

A dataset containing Transcription Start Sites for the Mus musculus mm10 genome assembly

## Usage

```
TSS_mm10(metadata=FALSE)
```

## Arguments

| | |
|---|---|
| `metadata` | `logical` value indicating whether only metadata should be returned or if the resource should be loaded |

## Format

A named vectors of lengths with one item per chromosome

## Value

A GRanges object with mm10 TSS locations.

## Source

EnsDb.Mmusculus.v79 package

## Examples

```
mm10TSS = TSS_mm10()
```

---

`TSS_mm9` *mm9 TSS locations*

---

## Description

A dataset containing Transcription Start Sites for the Mus musculus mm9 genome assembly

## Usage

```
TSS_mm9(metadata=FALSE)
```

## Arguments

| | |
|---|---|
| `metadata` | `logical` value indicating whether only metadata should be returned or if the resource should be loaded |

## Format

A named vectors of lengths with one item per chromosome

## Value

A GRanges object with mm9 TSS locations.

## Source

TxDb.Mmusculus.UCSC.mm9.knownGene package

## Examples

```
mm9TSS = TSS_mm9()
```

# Index