Package: TxParq.Hs.gencode.v49 (via r-universe)

May 14, 2026

Title Parquet-based representation of GENCODE gene models v49 for Homo sapiens

Version 0.99.2

Description This is a parquet-based representation of GENCODE gene models v49 for Homo sapiens. Parquet is chosen to reduce footprint, to support tidyverse-oriented operations natively, and to provide opportunities for cloud-backed annotation services. Community contributions to functionality and architecture are welcome.

Depends R (>= 4.5.0), utils

Suggests knitr, BiocStyle, testthat, rmarkdown

Imports dplyr, arrow, GenomicRanges, GenomeInfoDb, stats, GenomicFeatures, IRanges, methods, S4Vectors

License MIT + file LICENSE

VignetteBuilder knitr

biocViews Infrastructure

Encoding UTF-8

URL https://github.com/vjcitn/TxParq.Hs.gencode.v49

BugReports https://github.com/vjcitn/TxParq.Hs.gencode.v49/issues

Config/roxygen2/version 8.0.0

Config/pak/sysreqs cmake make libbz2-dev liblzma-dev libpng-dev libxml2-dev libssl-dev xz-utils zlib1g-dev

Repository https://biocstaging.r-universe.dev

Date/Publication 2026-05-14 12:27:56 UTC

RemoteUrl https://github.com/BiocStaging/TxParq.Hs.gencode.v49

RemoteRef HEAD

RemoteSha 101fb299961cb78034487b23a3da39749380ab4d

Contents

`biotype-queries` *List available gene or transcript types*

Description

Query the available biotypes in the annotation and their counts.

Usage

```
gene_types(x)

transcript_types(x)
```

Arguments

x	A `GTFParquet` object.

Value

A table of biotype counts.

Examples

```
gtf <- GTFParquet(system.file("gc49", package="TxParq.Hs.gencode.v49"))
gene_types(gtf)
# protein_coding      lncRNA     pseudogene ...
#         19950        16880          15200 ...

transcript_types(gtf)
```

convenience-functions *Convenience functions for common gene types*

Description

Helper functions to quickly extract genes of common biotypes.

Usage

```
protein_coding_genes(x, ...)

lncRNA_genes(x, ...)
```

Arguments

x	A GTFParquet object.
...	Additional arguments passed to genes,GTFParquet-method.

Value

A GRanges object.

Examples

```
gtf <- GTFParquet(system.file("gc49", package="TxParq.Hs.gencode.v49"))
pc <- protein_coding_genes(gtf)
lnc <- lncRNA_genes(gtf)
```

genes *Extract genomic features from a GTFParquet object*

Description

Methods to extract genomic features from a GTFParquet object as GRanges. Unlike TxDb methods, these preserve all GTF attributes as metadata columns.

Usage

```
## S4 method for signature 'GTFParquet'
genes(x, columns=NULL, filter=NULL, use_versioned_ids=FALSE)

## S4 method for signature 'GTFParquet'
transcripts(x, columns=NULL, filter=NULL, use_versioned_ids=FALSE)

## S4 method for signature 'GTFParquet'
```

```
exons(x, columns=NULL, filter=NULL, use_versioned_ids=FALSE)

## S4 method for signature 'GTFParquet'
cds(x, columns=NULL, filter=NULL)

## S4 method for signature 'GTFParquet'
transcripts(x, columns = NULL, filter = NULL, use_versioned_ids = FALSE)

## S4 method for signature 'GTFParquet'
exons(x, columns = NULL, filter = NULL, use_versioned_ids = FALSE)

## S4 method for signature 'GTFParquet'
cds(x, columns = NULL, filter = NULL)
```

Arguments

x	A GTFParquet object.
columns	Character vector of columns to include in mcols. If NULL (default), includes all available attribute columns. For genes: gene_name, gene_type, source, level, tags, havana_gene. For transcripts: transcript_name, transcript_type, gene_id, gene_name, transcript_support_level, ccdsid, protein_id.
filter	Optional named list for filtering features. Names should be column names, values are vectors of acceptable values. Example: filter = list(gene_type = "protein_coding", chrom = "chr1")
use_versioned_ids	Logical. If TRUE, use full versioned IDs (e.g., ENSG00000141510.18). If FALSE (default), use stripped IDs (e.g., ENSG00000141510).

Details

These methods return GRanges objects with feature IDs as names and rich metadata columns from the original GTF file.

The filter argument enables efficient server-side filtering through Arrow/Parquet predicate pushdown, which can dramatically improve performance compared to subsetting after loading.

Available filter columns include:

- chrom: Chromosome name
- gene_type: Gene biotype (e.g., "protein_coding", "lncRNA")
- transcript_type: Transcript biotype
- level: Annotation confidence (1=verified, 2=manual, 3=automatic)
- source: Annotation source ("HAVANA", "ENSEMBL")

Value

A GRanges object with:

- Feature IDs as names

- Genomic coordinates (`seqnames`, `ranges`, `strand`)
- Genome build in `seqinfo` (e.g., "GRCh38")
- Rich metadata in `mcols`

See Also

- `GTFParquet-class` for the class definition
- `transcriptsBy,GTFParquet-method` for grouped extraction
- `genes` for the generic

Examples

```
gtf <- GTFParquet(system.file("gc49", package="TxParq.Hs.gencode.v49"))

# Extract all genes with full attributes
gr <- genes(gtf)
S4Vectors::mcols(gr)  # gene_name, gene_type, level, tags, source, havana_gene

# Filter by gene type
pc <- genes(gtf, filter = list(gene_type = "protein_coding"))
lnc <- genes(gtf, filter = list(gene_type = "lncRNA"))

# Combine filters
pc_chr1 <- genes(gtf, filter = list(gene_type = "protein_coding", chrom = "chr1"))

# Select specific columns only
gr <- genes(gtf, columns = c("gene_name", "gene_type"))

# Use versioned IDs
gr <- genes(gtf, use_versioned_ids = TRUE)
names(gr)[1]  # "ENSG00000141510.18"

# Transcripts with support level
tx <- transcripts(gtf)
# note that transcript_support_level is frequently missing
high_conf <- tx[na.omit(S4Vectors::mcols(tx)$transcript_support_level) == "1"]

# Exons
ex <- exons(gtf, filter = list(chrom = "chr1"))

# CDS with protein IDs
cds_gr <- cds(gtf)
S4Vectors::mcols(cds_gr)$protein_id
```

gtf_metadata *Get GTF metadata*

Description

Retrieve metadata from the GTF file header, including provider, version, date, and genome build.

Usage

```
gtf_metadata(x)
```

Arguments

x	A GTFParquet object.

Value

A named character vector of metadata key-value pairs.

Examples

```
gtf <- GTFParquet(system.file("gc49", package="TxParq.Hs.gencode.v49"))
gtf_metadata(gtf)
#      provider         format           date         genome
#      "GENCODE"          "gtf"   "2025-07-08"       "GRCh38"
```

GTFParquet *Create a GTFParquet object*

Description

Create a GTFParquet object

Usage

```
GTFParquet(path)
```

Arguments

path	Path to directory containing Parquet files from gtf_to_parquet.py

Value

A GTFParquet S4 object

Examples

```
gtf <- GTFParquet(system.file("gc49", package="TxParq.Hs.gencode.v49"))
genes(gtf)
genes(gtf, filter = list(gene_type = "protein_coding"))
```

`GTFParquet-class` *GTFParquet class*

Description

An S4 class for accessing GTF annotations stored in Parquet format. Unlike TxDb, preserves all GTF attributes (gene_type, gene_name, transcript_support_level, tags, etc.)

Usage

```
## S4 method for signature 'GTFParquet'
genome(x)

## S4 method for signature 'GTFParquet'
seqinfo(x)
```

Arguments

x	A `GTFParquet` object.

Details

GTFParquet objects are created by the `GTFParquet` constructor function from a directory of Parquet files generated by `gtf_to_parquet.py`.

The class implements methods for GenomicFeatures generics including `genes`, `transcripts`, `exons`, `cds`, `exonsBy`, `cdsBy`, and `transcriptsBy`.

All methods support a `filter` argument for efficient querying (e.g., `filter = list(gene_type = "protein_coding")`).

Value

A Seqinfo object containing chromosome names and genome build.

Slots

`path` Character. Path to the Parquet directory.

`files` List. Paths to individual Parquet files.

`available` Logical vector. Which files are present.

`is_partitioned` Logical. Whether genes are partitioned by chromosome.

`.genome` Character. Reference genome build (e.g., "GRCh38").

See Also

- GTFParquet for the constructor function
- genes,GTFParquet-method for extracting genes
- transcriptsBy,GTFParquet-method for grouped extractors
- TxDb for comparison with TxDb objects

seqinfo

Examples

```
# Create from Parquet directory
gtf <- GTFParquet(system.file("gc49", package="TxParq.Hs.gencode.v49"))

# Extract genes with full attributes
gr <- genes(gtf)
S4Vectors::mcols(gr)  # gene_name, gene_type, level, tags, etc.

# Filter by gene type
pc <- genes(gtf, filter = list(gene_type = "protein_coding"))
```

---

region-queries *Find features overlapping a genomic region*

Description

Efficient region queries that use chromosome-based filtering before computing overlaps.

Usage

```
genes_in_region(x, region, ...)

transcripts_in_region(x, region, ...)
```

Arguments

x	A GTFParquet object.
region	A GRanges object specifying the query region(s).
...	Additional arguments passed to genes() or transcripts().

Details

These functions first filter by chromosome (using Parquet predicate pushdown for efficiency), then compute overlaps using findOverlaps.

Value

A GRanges object containing features that overlap the query region.

Examples

```
gtf <- GTFParquet(system.file("gc49", package="TxParq.Hs.gencode.v49"))

# Define a query region
region <- GenomicRanges::GRanges("chr1", IRanges::IRanges(1000000, 2000000))

# Find overlapping genes
genes_in_region(gtf, region)

# Find overlapping transcripts (protein-coding only)
transcripts_in_region(gtf, region,
                      filter = list(transcript_type = "protein_coding"))
```

---

show,GTFParquet-method *printer for GTFParquet*

Description

printer for GTFParquet

Usage

```
## S4 method for signature 'GTFParquet'
show(object)
```

Arguments

object	instance of GTFParquet

---

transcriptsBy *Extract and group genomic features from a GTFParquet object*

Description

Generic functions to extract genomic features of a given type grouped based on another type of genomic feature. These methods extend the GenomicFeatures generics for GTFParquet objects.

Usage

```
## S4 method for signature 'GTFParquet'
transcriptsBy(x, by="gene", filter=NULL)

## S4 method for signature 'GTFParquet'
exonsBy(x, by=c("tx", "gene"), filter=NULL)

## S4 method for signature 'GTFParquet'
cdsBy(x, by=c("tx", "gene"), filter=NULL)

## S4 method for signature 'GTFParquet'
cdsBy(x, by = c("tx", "gene"), filter = NULL)

## S4 method for signature 'GTFParquet'
transcriptsBy(x, by = "gene", filter = NULL)
```

Arguments

x	A GTFParquet object.
by	One of "gene", "tx" (transcript). Determines the grouping. For transcriptsBy, only "gene" is currently supported.
filter	Optional named list for filtering features before grouping. Names should be column names (e.g., gene_type, chrom), values are vectors of acceptable values. Example: filter = list(gene_type = "protein_coding", chrom = "chr1")

Details

These functions return a GRangesList object where the ranges within each of the elements are ordered according to the following rule:

When using exonsBy or cdsBy with by = "tx", the returned exons or CDS are ordered by ascending exon number for each transcript, that is, by their position in the transcript. In all other cases, the ranges will be ordered by chromosome, strand, start, and end values.

Unlike TxDb methods, GTFParquet methods preserve rich metadata columns including transcript_name, transcript_type, exon_number, protein_id, and frame.

The filter argument allows efficient server-side filtering before data is loaded into R, which can dramatically improve performance for large annotation files.

Value

A GRangesList object. The names of the list elements are the IDs of the grouping features (gene IDs or transcript IDs).

For GTFParquet objects, the names use stripped (unversioned) IDs by default (e.g., ENSG00000141510 rather than ENSG00000141510.18).

See Also

- GTFParquet-class for the class definition
- genes,GTFParquet-method for extracting ungrouped features
- transcriptsBy for the generic
- exonsBy for the generic
- cdsBy for the generic

Examples

```
gtf <- GTFParquet(system.file("gc49", package="TxParq.Hs.gencode.v49"))

# Exons grouped by transcript (sorted by exon_number)
ebt <- exonsBy(gtf, by = "tx")
ebt[[1]]  # Exons for first transcript

# Exons grouped by gene
ebg <- exonsBy(gtf, by = "gene")

# CDS grouped by transcript
cbt <- cdsBy(gtf, by = "tx")

# Transcripts grouped by gene
tbg <- transcriptsBy(gtf, by = "gene")

## Filter to protein-coding only - no, gene_type not available - FIXME?
#pc_exons <- exonsBy(gtf, by = "tx",
#                    filter = list(gene_type = "protein_coding"))

# Filter by chromosome
chr1_cds <- cdsBy(gtf, by = "tx", filter = list(chrom = "chr1"))
```

---

utrs *Extract UTR and codon features from a GTFParquet object*

Description

Functions to extract UTR (untranslated region) and codon features from a GTFParquet object. These features are stored in the features.parquet file generated by gtf_to_parquet.py.

Usage

```
utrs(x, type = "both", filter = NULL)

## S4 method for signature 'GTFParquet'
utrs(x, type = c("both", "5prime", "3prime"), filter = NULL)
```

```
codons(x, type = "both", filter = NULL)

## S4 method for signature 'GTFParquet'
codons(x, type = c("both", "start", "stop"), filter = NULL)
```

Arguments

x	A GTFParquet object.
type	For utrs: one of "both", "5prime", or "3prime". For codons: one of "both", "start", or "stop".
filter	Optional named list for filtering features.

Value

A GRanges object with metadata columns including feature_type, transcript_id, and gene_id.

Examples

```
gtf <- GTFParquet(system.file("gc49", package="TxParq.Hs.gencode.v49"))

# 5' UTRs
utr5 <- utrs(gtf, type = "5prime")

# 3' UTRs
utr3 <- utrs(gtf, type = "3prime")

# Start codons
start <- codons(gtf, type = "start")

# Stop codons
stop <- codons(gtf, type = "stop")
```

Index