Package 'statgen'

May 27, 2026

Type Package

Title Statistical Genetics Data Objects and Loaders

Version 0.3.2

Description Loads and manages statistical genetics data objects including reference panels, genotypes, LD matrices, annotations, and summary statistics. Follows the 'statgen' specification for use across 'Python', 'R', and 'MATLAB'/'Octave' runtimes.

License MIT + file LICENSE

URL https://github.com/precimed/statgen

BugReports https://github.com/precimed/statgen/issues

Encoding UTF-8

Depends R (>= 4.1)

Imports Matrix, jsonlite, digest, bit64, methods, data.table

Suggests R.utils, testthat (>= 3.0.0), knitr, rmarkdown

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author Oleksandr Frei [aut, cph],
Andrew Morris [cre, ctb]

Maintainer Andrew Morris <a.h.morris@mn.uio.no>

Repository CRAN

Date/Publication 2026-05-27 20:10:17 UTC

Contents

`statgen-package` *Statistical Genetics Data Objects and Loaders*

Description

CRAN-compatible R runtime for statgen statistical genetics data objects.

Details

statgen provides R objects and loaders for reference-aligned statistical genetics data: reference variants, genotypes, linkage disequilibrium (LD), annotations, and GWAS summary statistics. The package is designed for analysis workflows where genome build, contig labels, and allele orientation have already been harmonized upstream.

The central object is a `ReferencePanel`, which defines the ordered SNP axis used by the other panel types. `LDPanel`, `GenotypePanel`, `AnnotationPanel`, and `Sumstats` objects expose data in those reference coordinates, so operations can combine objects without silently renaming contigs, swapping alleles, or dropping unmatched variants.

Typical workflows start by loading or caching a reference panel with `load_reference`, then loading aligned data with functions such as `load_ld`, `load_genotype`, `load_annotations`, or `load_sumstats`. Panel objects provide accessors for aligned vectors and matrices, `select_shards` for canonical chromosome subsetting, and cache helpers for repeated analyses.

LD distributions are directory-based runtime artifacts discovered through `ld_manifest.json`; they are loaded with `load_ld` rather than as individual shard files. R reads the Python `.npz` LD shard format directly after `prepare_ld_npz_for_r` adds an R reference-cache sidecar. R-native object caches use RDS files, while portable source inputs such as PLINK BIM/BED/FAM, BED annotations, and summary-statistics TSV files remain external inputs.

See Also

`load_reference`, `load_ld`, `load_genotype`, `load_annotations`, `load_sumstats`, `version`, `set_verbosity`

annotations *AnnotationPanel objects*

Description

Paint BED interval annotations onto a reference panel, inspect sparse annotation matrices, and save or load R-native annotation caches.

Usage

```
load_annotations(bed_paths, reference)
create_annotations(reference, annotation_matrix, annotation_names)
create_annotation(reference, annovec, annoname)
save_annotations_cache(panel, path)
load_annotations_cache(path, shards = NULL)

annomat(x, ...)
annonames(x, ...)
num_annot(x, ...)
select_annotations(x, names, ...)
union_annotations(x, other, mode = "by_name", ...)
```

Arguments

bed_paths	Character path or character vector of BED files.
reference	A ReferencePanel.
annotation_matrix	A binary matrix with SNPs as rows and annotations as columns.
annotation_names, names	Character annotation names.
annovec	Binary annotation vector aligned to reference.
annoname	Character scalar annotation name.
panel, x, other	Annotation panel objects.
path	Character scalar cache path.
shards	Optional character vector of shard labels in canonical order.
mode	Union mode. Currently only "by_name" is supported.
...	Reserved for S3 methods.

Details

BED intervals are interpreted as 0-based half-open intervals and are matched to reference chromosome labels exactly. The annotation matrix is binary and reference-aligned.

Value

`load_annotations()`, `create_annotations()`, `create_annotation()`, and `load_annotations_cache()` return `AnnotationPanel` S3 objects. `annomat(x)` returns a sparse `Matrix::lgCMatrix` with column names equal to `annonames(x)`.

Examples

```
ref_path <- system.file("extdata", "reference_chr1.bim", package = "statgen")
bed_path <- system.file("extdata", "anno1.bed", package = "statgen")
ref <- load_reference(ref_path)
ann <- load_annotations(bed_path, ref)
dim(annomat(ann))

cache <- tempfile(fileext = ".rds")
save_annotations_cache(ann, cache)
ann_cached <- load_annotations_cache(cache)
dim(annomat(ann_cached))
```

---

genotype *GenotypePanel objects*

Description

Load PLINK bfile-backed genotype metadata, inspect reference-aligned genotype panels, cache metadata, and fetch hardcall genotype slices from BED files.

Usage

```
load_genotype(bfile_prefix, reference)
save_genotype_cache(panel, path)
load_genotype_cache(path, shards = NULL)

num_sample(x, ...)
fid(x, ...)
iid(x, ...)
father_id(x, ...)
mother_id(x, ...)
sex(x, ...)
is_male(x, ...)
is_female(x, ...)
ploidy_male(x, ...)
ploidy_female(x, ...)
source_layout(x, ...)
source_row0(x, ...)
is_subject_present(x, shard, ...)

fetch_genotypes_int8(x, snp_indices, bed_path = NULL, ...)
fetch_genotypes(x, snp_indices, bed_path = NULL,
  haploid_mode = NULL, ...)
```

Arguments

bfile_prefix	Character scalar PLINK bfile prefix, without suffixes. Use @ as a shard-label placeholder for sharded input.
reference	A ReferencePanel.
panel, x	A GenotypePanel.
path	Character scalar path to an RDS genotype metadata cache.
shards	Optional character vector of shard labels in canonical order.
shard	A loaded shard label for subject-presence lookup.
snp_indices	One-based panel-global SNP indices.
bed_path	Optional BED path override. A sharded override may contain @.
haploid_mode	Optional output mode, "raw" or "ploidy_scaled". The default is "raw".
...	Reserved for S3 methods.

Details

Genotype panels are reference-aligned metadata handles for PLINK BED payloads. Hardcalls are decoded on demand; caches store only metadata needed to locate and map BED rows.

Value

load_genotype() and load_genotype_cache() return GenotypePanel S3 objects. Metadata accessors return ordinary R vectors. fetch_genotypes_int8() returns an integer matrix with samples as rows and requested SNPs as columns, using -1 for missing hardcalls. fetch_genotypes() returns a numeric matrix with missing calls as NaN.

Examples

```
ref_path <- system.file("extdata", "reference_chr1.bim", package = "statgen")
geno_bed <- system.file("extdata", "genotype_1.bed", package = "statgen")
geno_prefix <- sub("\\.bed$", "", geno_bed)
ref <- load_reference(ref_path)
g <- load_genotype(geno_prefix, ref)
geno <- fetch_genotypes(g, c(1, 2))
dim(geno)
head(geno)

cache <- tempfile(fileext = ".rds")
save_genotype_cache(g, cache)
g_cached <- load_genotype_cache(cache)
num_sample(g_cached)
sum(is_present(g_cached))
```

ld *LD panel loading, preparation, and operations*

Description

Prepare Python LD .npz distributions for R loading, validate their manifests and sparse CSC payloads, load LD panels, and run the core LD operations.

Usage

```
load_ld(path, shards = NULL, default_chrX_sex = NULL, retain_ld_r = TRUE)
load_ld_reference(path, shards = NULL)
validate_ld_distribution(path, check_payload_structure = FALSE)
prepare_ld_npz_for_r(npz_root, extract_npz = FALSE)
a1freq(x, chrX_sex = NULL, ...)
multiply_r2(ld_panel, M, chrX_sex = NULL, ...)
fast_prune(logpvec, ld_panel, r2_threshold = 0.2, chrX_sex = NULL)
reference(x, ...)
default_chrX_sex(x, ...)
```

Arguments

path	Path to a Python .npz LD distribution root prepared for R loading.
shards	Optional character vector of canonical shard labels to load.
default_chrX_sex	Default chrX LD shard selector, one of "female", "male", or "combined".
retain_ld_r	Logical; retain raw signed LD r matrices in loaded shards. Operations keep working when this is FALSE.
check_payload_structure	Logical; perform expensive sparse payload checks during distribution validation.
npz_root	Path to a Python .npz LD distribution root.
extract_npz	Logical; when TRUE, create or refresh sibling *.npz.d directories containing extracted .npy arrays for faster R loading.
x	An LD panel or shard object.
ld_panel	An LDPanel.
M	A numeric vector or matrix with rows aligned to the LD panel reference.
logpvec	Numeric vector of signed or unsigned log-p values aligned to the LD panel reference.
r2_threshold	Finite non-negative pruning threshold; defaults to 0.2.
chrX_sex	Optional chrX LD shard selector for operations. Ignored when chrX is absent.
...	Reserved for S3 method dispatch.

Details

R reads Python LD .npz shard files directly. R-loadable distributions use ld_manifest.json, bundled reference BIM files, Python .npz LD shards, and an R reference cache sidecar recorded in manifest fields r_reference_cache and r_reference_cache_md5. load_ld() constructs Matrix::dgCMatrix objects in memory and derives the elementwise squared LD matrix used by multiply_r2() and fast_prune().

prepare_ld_npz_for_r() builds or refreshes the R reference cache sidecar from the bundled BIM files for all shards described in the existing Python manifest and updates that manifest in place. With extract_npz = TRUE, it also creates or refreshes extracted *.npz.d cache directories next to the manifest-declared shard files. The operation is idempotent.

Value

load_ld() returns an LDPanel. load_ld_reference() returns the paired ReferencePanel. validate_ld_distribution(returns a list with ok = TRUE on success. prepare_ld_npz_for_r() returns the updated manifest invisibly. a1freq() returns aligned allele frequencies, multiply_r2() returns a vector or matrix with the same shape as M, and fast_prune() returns a numeric vector or one-dimensional matrix with pruned entries set to NaN.

Examples

```
ld_root <- system.file("extdata", "ld", package = "statgen")
ld <- load_ld(ld_root)

num_snp(ld)
head(a1freq(ld))

scores <- seq_len(num_snp(ld))
multiply_r2(ld, scores)

logp <- rev(seq_len(num_snp(ld)))
fast_prune(logp, ld, r2_threshold = 0.35)
```

---

reference *ReferencePanel objects*

Description

Load PLINK BIM reference panels as statgen ReferencePanel S3 objects, inspect aligned reference vectors, and save or load R-native reference caches.

Usage

```
# Constructors and cache I/O
load_reference(path, shards = NULL)
load_reference_cache(path, shards = NULL)
save_reference_cache(panel, path)
```

```
# Accessors
num_snp(x, ...)
shards(x, ...)
chr(x, ...)
snp(x, ...)
bp(x, ...)
a1(x, ...)
a2(x, ...)
a1_hash64(x, ...)
a2_hash64(x, ...)
is_single_nucleotide_variant(x, ...)
is_strand_ambiguous(x, ...)
shard_offsets(x, ...)

# Operations
select_shards(x, shards, ...)
is_object_compatible(reference, object, ...)
validate_checksums(x, ...)
save_cache(x, path, ...)
```

Arguments

path	Character scalar path. For load_reference(), this is a BIM file or sharded BIM template containing @. For load_reference_cache(), this is an RDS reference cache. For cache writers, this is the output cache path.
shards	Optional character vector of shard labels in canonical order.
panel, reference, x, object	Reference panel or compatible statgen object.
...	Reserved for S3 methods.

Details

Reference panels are represented as ReferencePanel S3 objects backed by an ordered list of reference shards.

Most exported functions in this API are S3 generics operating on ReferencePanel objects and other reference-aligned statgen objects. Users should access panel data through accessor functions such as bp(x) and a1(x) rather than directly accessing internal fields.

is_object_compatible(reference, object) dispatches on the first argument, which should be a ReferencePanel. Other reference-aligned panel classes participate by providing shards() methods that return their ordered shard objects.

shards(x) returns the ordered shard objects. save_cache() is the cross-object cache-writing generic; save_reference_cache() is the explicit reference-panel cache writer.

Value

load_reference() and load_reference_cache() return a ReferencePanel S3 object. Accessors return ordinary R vectors or data frames; allele hashes are bit64::integer64 vectors.

Examples

```
path <- system.file("extdata", "reference_chr1.bim", package = "statgen")
ref <- load_reference(path)
num_snp(ref)
head(snp(ref))
sum(is_strand_ambiguous(ref))

cache <- tempfile(fileext = ".rds")
save_reference_cache(ref, cache)
ref_cached <- load_reference_cache(cache)
num_snp(ref_cached)

save_cache(ref, cache)
```

`sumstats` *Sumstats objects*

Description

Load, create, inspect, and cache reference-aligned GWAS summary statistics.

Usage

```
load_sumstats(path, reference)
create_sumstats(reference, p, z = NULL, n = NULL,
  beta = NULL, se = NULL, eaf = NULL, info = NULL)
save_sumstats_cache(sumstats, path)
load_sumstats_cache(path, shards = NULL)

logpvec(x, ...)
zvec(x, ...)
nvec(x, ...)
is_present(x, ...)
beta_vec(x, ...)
se_vec(x, ...)
eaf_vec(x, ...)
info_vec(x, ...)
```

Arguments

`path`	Character scalar path to a summary-statistics TSV or RDS cache.
`reference`	A `ReferencePanel`.
`sumstats`, `x`	A `Sumstats` object.
`p`, `z`, `n`, `beta`, `se`, `eaf`, `info`	Aligned numeric vectors. Optional vectors may be `NULL`.
`shards`	Optional character vector of shard labels in canonical order.
`...`	Reserved for S3 methods.

Details

Rows are projected onto the supplied reference by exact chromosome, position, and allele-hash matching. Missing reference variants are represented by `NaN` in `logpvec(x)` and by `is_present(x) == FALSE`. Gzipped TSV input requires the suggested **R.utils** package so `data.table::fread()` can read compressed files portably.

Value

`load_sumstats()`, `create_sumstats()`, and `load_sumstats_cache()` return `Sumstats` S3 objects. Accessors return ordinary R vectors, with `NULL` for absent optional fields.

Examples

```
ref_path <- system.file("extdata", "reference_chr1.bim", package = "statgen")
sum_path <- system.file("extdata", "traits_complete.tsv.gz", package = "statgen")
ref <- load_reference(ref_path)
s <- load_sumstats(sum_path, ref)
head(logpvec(s))

cache <- tempfile(fileext = ".rds")
save_sumstats_cache(s, cache)
s_cached <- load_sumstats_cache(cache)
head(logpvec(s_cached))
```

`verbosity` *Get or Set Runtime Verbosity*

Description

Controls the package-wide statgen runtime verbosity setting.

Usage

```
get_verbosity()
set_verbosity(level)
```

Arguments

`level`	Character scalar, either `"quiet"` or `"info"`.

Value

`get_verbosity()` returns the current verbosity level. `set_verbosity()` returns the selected level invisibly.

Examples

```
old <- get_verbosity()
set_verbosity("quiet")
get_verbosity()
set_verbosity(old)
```

`version` *Return the statgen Package Version*

Description

Returns the installed statgen R package version.

Usage

```
version()
```

Value

A character scalar containing the installed package version.

Examples

```
version()
```

Index