Package ‘HaploGeno’

December 4, 2025

Title Haplotype-Based Genomic Prediction

Version 0.3.0

Author Zachary Aldiss [aut, cre]

Maintainer Zachary Aldiss <zaldiss@illinois.edu>

Description Implements a high-performance framework for haplotype-
based genomic prediction and QTL discovery.
Leverages file-backed matrices via 'bigstatsr' to handle large-scale genomic datasets efficiently.
Key features include LD-
based haplotype block construction, kernel ridge regression (KRR) for genomic prediction,
and local genomic estimated breeding value (GEBV) analysis for identifying significant genomic regions.

License MIT

Encoding UTF-8

LazyData true

Imports R6, bigstatsr, data.table, Rcpp, methods, future,
future.apply, bigsnpr, progressr, ggplot2

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 7.3.3

Contents

encode_block_fast 2
HaploObject 2
load_demo_data 7
load_haplo_project 7
print.HaploObject 7
summary.HaploObject 8

`encode_block_fast` *Encode Haplotypes (Optimized)*

Description

Maps unique rows of a genotype matrix to integers without string conversion.

Usage

```
encode_block_fast(mat)
```

Arguments

`mat`	A numeric matrix of genotypes (0, 1, 2).

Value

Integer vector of haplotype IDs.

`HaploObject` *HaploObject Class*

Description

HaploObject Class

HaploObject Class

Public fields

`geno` Filebacked Big Matrix of genotypes.

`map` Data.table containing marker map information.

`pheno` Vector of phenotypes.

`blocks` Data.table defining haplotype blocks.

`haplo_mat` Matrix of encoded haplotypes.

`hrm` Haplotype Relationship Matrix.

`marker_effects` Vector of estimated marker effects.

`local_gebv` List containing local GEBV matrix and variances.

`significance` Data.table of significance test results.

`active_markers` Integer vector of indices of markers with non-zero variance.

Methods

Public methods:

- `HaploObject$new()`
- `HaploObject$print()`
- `HaploObject$import_genotypes()`
- `HaploObject$load_map()`
- `HaploObject$load_pheno()`
- `HaploObject$get_subset()`
- `HaploObject$define_blocks_fixed()`
- `HaploObject$define_blocks_ld()`
- `HaploObject$estimate_marker_effects()`
- `HaploObject$encode_haplotypes()`
- `HaploObject$calculate_local_gebv()`
- `HaploObject$test_significance()`
- `HaploObject$compute_hrm()`
- `HaploObject$fit_krr()`
- `HaploObject$cross_validate()`
- `HaploObject$plot_manhattan()`
- `HaploObject$plot_pca()`
- `HaploObject$plot_gebv_image()`
- `HaploObject$identify_superior_haplotypes()`
- `HaploObject$score_stacking()`
- `HaploObject$plot_stacking_trend()`
- `HaploObject$impute_genotypes()`
- `HaploObject$filter_monomorphic()`
- `HaploObject$save_project()`
- `HaploObject$clone()`

Method `new()`**:** Initialize a new HaploObject.

Usage:

```
HaploObject$new(backing_file_path)
```

Arguments:

`backing_file_path` Path to the backing file for the FBM.

Method `print()`**:** Print a summary of the HaploObject.

Usage:

```
HaploObject$print()
```

Method `import_genotypes()`**:** Import genotypes from a file or matrix.

Usage:

```
HaploObject$import_genotypes(matrix_or_path)
```

Arguments:

`matrix_or_path` A matrix or path to a file (bed/vcf/rds).

Method `load_map()`**:** Load marker map.

Usage:

```
HaploObject$load_map(map_data)
```

Arguments:

`map_data` A data.frame or path to a file.

Method `load_pheno()`**:** Load phenotypes.

Usage:

```
HaploObject$load_pheno(pheno_data)
```

Arguments:

`pheno_data` A vector, data.frame, or path to a file.

Method `get_subset()`**:** Get a subset of the genotype matrix.

Usage:

```
HaploObject$get_subset(row_ind, col_ind)
```

Arguments:

`row_ind` Row indices.

`col_ind` Column indices.

Method `define_blocks_fixed()`**:** Define haplotype blocks using fixed window size.

Usage:

```
HaploObject$define_blocks_fixed(window_size)
```

Arguments:

`window_size` Number of markers per block.

Method `define_blocks_ld()`**:** Define haplotype blocks using LD scan.

Usage:

```
HaploObject$define_blocks_ld(
  r2_threshold = 0.1,
  tolerance = 3,
  window_size = 2000
)
```

Arguments:

`r2_threshold` r2 threshold for block definition.

`tolerance` Number of failures allowed before ending a block.

`window_size` Maximum window size to scan.

Method `estimate_marker_effects()`**:** Estimate marker effects using Ridge Regression.

Usage:

```
HaploObject$estimate_marker_effects(lambda = 1)
```

Arguments:

`lambda` Regularization parameter.

Method `encode_haplotypes()`**:** Encode haplotypes into integer IDs.

Usage:

```
HaploObject$encode_haplotypes(n_cores = 1)
```

Arguments:

`n_cores` Number of cores to use.

Method `calculate_local_gebv()`**:** Calculate local GEBVs.

Usage:

```
HaploObject$calculate_local_gebv(n_cores = 1)
```

Arguments:

`n_cores` Number of cores to use.

Method `test_significance()`**:** Test significance of local GEBVs.

Usage:

```
HaploObject$test_significance()
```

Returns: Data.table of p-values.

Method `compute_hrm()`**:** Compute Haplotype Relationship Matrix.

Usage:

```
HaploObject$compute_hrm(n_cores = 1)
```

Arguments:

`n_cores` Number of cores to use.

Method `fit_krr()`**:** Fit Kernel Ridge Regression model.

Usage:

```
HaploObject$fit_krr(lambda = 0.1, use_cg = NULL, tol = 1e-05, max_iter = 1000)
```

Arguments:

`lambda` Regularization parameter.

`use_cg` Whether to use Conjugate Gradient solver.

`tol` Tolerance for CG.

`max_iter` Maximum iterations for CG.

Method `cross_validate()`**:** Cross-validate KRR model.

Usage:

```
HaploObject$cross_validate(k = 5, lambdas = NULL, n_cores = 1, folds = NULL)
```

Arguments:

`k` Number of folds.

`lambdas` Vector of lambdas to test.

`n_cores` Number of cores to use.

`folds` Optional vector of fold assignments.

Method `plot_manhattan()`**:** Plot Manhattan plot of local GEBV significance.

Usage:

```
HaploObject$plot_manhattan(threshold = 0.05)
```

Arguments:

`threshold` Significance threshold (p-value).

Method `plot_pca()`**:** Plot PCA of HRM.

Usage:

```
HaploObject$plot_pca(groups = NULL)
```

Arguments:

`groups` Optional vector of groups for coloring.

Method `plot_gebv_image()`**:** Plot heatmap of local GEBVs.

Usage:

```
HaploObject$plot_gebv_image(block_range = NULL)
```

Arguments:

`block_range` Optional range of blocks to plot.

Method `identify_superior_haplotypes()`**:** Identify superior haplotypes.

Usage:

```
HaploObject$identify_superior_haplotypes(top_n = 50)
```

Arguments:

`top_n` Number of top blocks to consider.

Method `score_stacking()`**:** Calculate stacking scores.

Usage:

```
HaploObject$score_stacking(superior_haplos)
```

Arguments:

`superior_haplos` Data.table of superior haplotypes.

Method `plot_stacking_trend()`**:** Plot stacking trend.

Usage:

```
HaploObject$plot_stacking_trend(scores = NULL, superior_haplos = NULL)
```

Arguments:

`scores` Vector of stacking scores.

`superior_haplos` Optional table of superior haplotypes (if scores not provided).

Method `impute_genotypes()`**:** Impute missing genotypes.

Usage:

```
HaploObject$impute_genotypes(method = "mean")
```

Arguments:

`method` Imputation method. Currently only "mean" is supported.

Method `filter_monomorphic()`**:** Filter monomorphic markers (zero variance).

Usage:

```
HaploObject$filter_monomorphic()
```

Method `save_project()`**:** Save the project to an RDS file.

Usage:

```
HaploObject$save_project(path)
```

Arguments:

`path` Path to the output .rds file.

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
HaploObject$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`load_demo_data` *Load Demo Dataset*

Description

Loads a pre-processed demo dataset for testing and visualization. The dataset contains 50 samples and 500 markers, with blocks defined, haplotypes encoded, and local GEBVs calculated.

Usage

```
load_demo_data()
```

Value

A HaploObject instance.

`load_haplo_project` *Load HaploGeno Project*

Description

Load HaploGeno Project

Usage

```
load_haplo_project(path)
```

Arguments

`path`	Path to the .rds file.

Value

A HaploObject instance.

`print.HaploObject` *Print method for HaploObject*

Description

Print method for HaploObject

Usage

```
## S3 method for class 'HaploObject'
print(x, ...)
```

Arguments

`x`	A HaploObject
`...`	Additional arguments

`summary.HaploObject` *Summary method for HaploObject*

Description

Summary method for HaploObject

Usage

```
## S3 method for class 'HaploObject'
summary(object, ...)
```

Arguments

`object`	A HaploObject
`...`	Additional arguments