Package: snp.slicer (via r-universe)

June 3, 2026

Title Bayesian Nonparametric Resolution of Multi-Strain Infections

Version 0.1.0

Author Nianqiao Ju [aut], Maxwell Murphy [cre]

Maintainer Maxwell Murphy <maxwell.murphy@ucsf.edu>

Description Implementation of SNP-Slice, a Bayesian nonparametric method for resolving multi-strain infections using slice sampling with stick-breaking construction. The algorithm simultaneously estimates strain haplotypes and links them to hosts from sequencing data. Supports multiple observation models including categorical, Poisson, binomial, and negative binomial distributions.

License GPL-3 + file LICENSE

Encoding UTF-8

LazyData true

Imports dplyr, readr, tidyr

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Suggests devtools, testthat (>= 3.0.0), knitr, rmarkdown, ggplot2, pkgdown

VignetteBuilder knitr

URL https://github.com/plasmogenepi/snp.slicer, https://plasmogenepi.github.io/snp.slicer/

BugReports https://github.com/plasmogenepi/snp.slicer/issues

Depends R (>= 4.2.0)

Config/pak/sysreqs libicu-dev libx11-dev

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

Date/Publication 2026-06-03 20:12:50 UTC

RemoteUrl https://github.com/PlasmoGenEpi/snp.slicer

RemoteRef HEAD

RemoteSha 303bfd83dd00543f40e9feca2a037c44859e8998

Contents

`calculate_allele_frequencies` *Calculate Allele Frequencies from MCMC Results*

Usage

```
calculate_allele_frequencies(
  results,
  snp_indices,
  use_map = TRUE,
  n_samples = 100,
  interval = 0.95,
  allele_sep = "|"
)
```

Arguments

`results`	A `snp_slice_results` object containing MCMC results
`snp_indices`	A vector of SNP indices to treat as a single allele
`use_map`	Logical, whether to use MAP estimates (TRUE) or sample from MCMC (FALSE)
`n_samples`	Number of MCMC samples to use if use_map = FALSE (default: 100)
`interval`	Numeric in (0, 1). Credible interval width when using MCMC (e.g. 0.95).
`allele_sep`	Separator for allele strings (default: "|")

Value

The structure depends on use_map.

MAP (use_map = TRUE**)** Data frame with columns: allele (string representation of the allele, e.g. "ref|alt|ref" for 3 SNPs), frequency (proportion of total parasites with this allele; sums to 1), count (number of parasites with this allele in the MAP allocation), total_parasites (total parasites in the MAP allocation; same for every row).

MCMC (use_map = FALSE**)** Data frame with columns: allele, frequency (posterior mean proportion), frequency_sd (posterior SD of proportion across samples), frequency_lower and frequency_upper (credible interval, e.g. 2.5\

Calculates allele frequencies for a collection of SNPs treated as a single allele. The function takes SNP indices and calculates the frequency of each possible allele combination across all individuals based on their strain allocations.

With use_map = FALSE, counts are summarized as per-sample means rather than sums, so mean_count and mean_total_parasites are interpretable regardless of n_samples. Frequency uncertainty (frequency_sd, frequency_lower, frequency_upper) is computed from the distribution of allele frequencies across MCMC samples.

Load example results result <- load_example_results()

Calculate allele frequencies for SNPs 1, 5, and 10 (MAP) allele_freqs <- calculate_allele_frequencies(result, c(1, 5, 10)) print(allele_freqs)

With MCMC: posterior mean, SD, credible interval, and per-sample mean count if (!is.null(result$mcmc_samples)) allele_freqs_mcmc <- calculate_allele_frequencies(result, c(1, 5, 10), use_map = FALSE, n_samples = 50) print(allele_freqs_mcmc)

---

calculate_allele_frequencies_by_sets

Calculate allele frequencies for multiple target sets

Description

For each target set, computes allele frequencies from MCMC/MAP results and returns a list of frequency tables (one per set). Each set is a vector of target indices or target names.

Usage

```
calculate_allele_frequencies_by_sets(
  results,
  target_sets,
  use_map = TRUE,
  n_samples = 100,
  interval = 0.95,
  allele_sep = "|"
)
```

Arguments

results	A snp_slice_results object containing MCMC results.
target_sets	List of vectors; each element is target indices (integer) or target names (character) defining one set. If the list is named, those names are used for the returned list.
use_map	Logical; use MAP estimates (TRUE) or sample from MCMC (FALSE).
n_samples	Number of MCMC samples to use if use_map = FALSE (default: 100).
interval	Credible interval width when use_map = FALSE (e.g. 0.95).
allele_sep	Separator for allele strings (default: "|").

Value

A named list of data frames, one per target set. List names come from names(target_sets) or "set_1", "set_2", etc. Each data frame has the same structure as the return value of calculate_allele_frequencies: with MAP, columns allele, frequency, count, total_parasites; with MCMC, columns allele, frequency, frequency_sd, frequency_lower, frequency_upper, mean_count, n_samples, and attribute mean_total_parasites. See that function's help for the meaning of each column.

Examples

```
result <- load_example_results()
target_sets <- list(locus_a = c(1, 5), locus_b = c(10))
freqs <- calculate_allele_frequencies_by_sets(result, target_sets)
print(freqs$locus_a)
```

---

calculate_individual_coi

Calculate estimated individual COI with uncertainty

Description

Returns per-host complexity of infection (COI). With use_map = TRUE you get a point estimate (MAP); with use_map = FALSE and MCMC samples, posterior mean, SD, and credible interval are computed.

Usage

```
calculate_individual_coi(
  results,
  use_map = TRUE,
  n_samples = 100,
  interval = 0.95
)
```

Arguments

results	A snp_slice_results object.
use_map	If TRUE (default), use MAP only; uncertainty columns are NA. If FALSE, use MCMC samples for mean, SD, and interval.
n_samples	When use_map = FALSE, number of MCMC samples to use (capped at available post-burnin samples).
interval	Numeric in (0, 1). Credible interval width when using MCMC (e.g. 0.95 for 2.5 and 97.5 percent quantiles).

Value

A data frame with one row per host: host_index, host_id, coi_estimate, coi_sd, coi_lower, coi_upper. Uncertainty columns are NA when using MAP or when no MCMC samples are available.

Examples

```
result <- load_example_results()
coi_map <- calculate_individual_coi(result, use_map = TRUE)
head(coi_map)
if (!is.null(result$mcmc_samples)) {
  coi_post <- calculate_individual_coi(result, use_map = FALSE, n_samples = 50)
  head(coi_post)
}
```

---

clear_performance_log *Clear performance log*

Description

Clear performance log

Usage

```
clear_performance_log()
```

`effective_sample_size` *Calculate effective sample size for MCMC diagnostics*

Description

Calculate effective sample size for MCMC diagnostics

Usage

```
effective_sample_size(
  results,
  parameter = "logpost",
  method = "autocorrelation"
)
```

Arguments

`results`	SNP-Slice results object
`parameter`	Parameter to analyze. Options include: • "logpost": Log posterior probability • "kstar": Number of active strains • "n_strains": Number of strains with non-zero allocations • "ktrunc": Truncation level • "mu": Stick-breaking weights (returns ESS for each weight) • "A": Allocation matrix (returns overall ESS) • "D": Dictionary matrix (returns overall ESS) • "all": Calculate ESS for all available parameters
`method`	Method for ESS calculation. Options: • "autocorrelation": Standard autocorrelation-based ESS (default) • "batch_means": Batch means method • "spectral": Spectral density method

Value

Effective sample size(s) and diagnostic information

example_read0 *Example Read Count Data - Reference Alleles*

Description

Example reference allele read count data for testing and demonstration purposes.

Usage

```
example_read0
```

Format

A matrix of reference allele read counts with 200 hosts and 96 SNPs

Source

Simulated data for package testing

example_read1 *Example Read Count Data - Alternate Alleles*

Description

Example alternate allele read count data for testing and demonstration purposes.

Usage

```
example_read1
```

Format

A matrix of alternate allele read counts with 200 hosts and 96 SNPs

Source

Simulated data for package testing

example_snp_data *Example SNP Data*

Description

Example SNP data for testing and demonstration purposes.

Usage

```
example_snp_data
```

Format

A list containing:

read0 Matrix of reference allele read counts

read1 Matrix of alternate allele read counts

Source

Simulated data for package testing

extract_allocations *Extract allocation information from SNP-Slice results*

Description

Extract allocation information from SNP-Slice results

Usage

```
extract_allocations(results)
```

Arguments

results	SNP-Slice results object

Value

List containing allocation information

`extract_strains` *Extract strain information from SNP-Slice results*

Description

Extract strain information from SNP-Slice results

Usage

```
extract_strains(results)
```

Arguments

`results`	SNP-Slice results object

Value

List containing strain information

`get_performance_summary` *Get performance summary*

Description

Get performance summary

Usage

```
get_performance_summary()
```

Value

Performance summary data frame

load_example_results *Load Example Analysis Results*

Description

Loads pre-computed SNP-Slice analysis results from the example data. These results were generated using the negative binomial model with 2000 MCMC iterations.

Usage

```
load_example_results()
```

Value

A `snp_slice_results` object containing the analysis results

Examples

```
# Load the pre-computed results
result <- load_example_results()
print(result)
```

performance_env *Performance Logger for SNP-Slice*

Description

Utility functions for tracking performance and identifying bottlenecks

Usage

```
performance_env
```

Format

An object of class `environment` of length 0.

plot_convergence *Plot convergence diagnostics*

Description

Plot convergence diagnostics

Usage

```
plot_convergence(results, type = "logpost")
```

Arguments

results	SNP-Slice results object
type	Type of plot ("logpost", "kstar", "n_strains")

Value

Plot object (if ggplot2 is available)

print.ess_all_results *Print comprehensive ESS results for all parameters*

Description

Print comprehensive ESS results for all parameters

Usage

```
## S3 method for class 'ess_all_results'
print(x, digits = 2, ...)
```

Arguments

x	List of ESS results
digits	Number of digits to display
...	Additional arguments

Value

Formatted ESS results

print.ess_result *Print effective sample size results*

Description

Print effective sample size results

Usage

```
## S3 method for class 'ess_result'
print(x, digits = 2, ...)
```

Arguments

x	ESS result object
digits	Number of digits to display
...	Additional arguments

Value

Formatted ESS results

print.snp_slice_results *Print SNP-Slice results*

Description

Print SNP-Slice results

Usage

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

Arguments

x	SNP-Slice results object
...	Additional arguments

Value

Print information

print_performance_summary *Print performance summary*

Description

Print performance summary

Usage

```
print_performance_summary()
```

snp_slice *Bayesian Nonparametric Resolution of Multi-Strain Infections*

Description

SNP-Slice is a Bayesian nonparametric method for resolving multi-strain infections using slice sampling with stick-breaking construction. The algorithm simultaneously unveils strain haplotypes and links them to hosts from sequencing data.

Usage

```
snp_slice(
  data,
  model = "negative_binomial",
  n_mcmc = 10000,
  burnin = NULL,
  alpha = 2.6,
  rho = if (model == "categorical") 0.5 else NULL,
  threshold = 0.001,
  gap = NULL,
  seed = NULL,
  verbose = TRUE,
  log_performance = FALSE,
  store_mcmc = FALSE,
  ...
)

snp_slice_categorical(data, e1 = 0.05, e2 = 0.05, ...)

snp_slice_poisson(data, ...)

snp_slice_binomial(data, ...)

snp_slice_negative_binomial(data, ...)
```

Arguments

data	Input data. Can be a matrix, data.frame, or file path. For read count data, should be a list with elements read1 and read0 (or total). For categorical data, can be a matrix with values 0, 0.5, or 1; or a long-format data.frame with columns specimen_id, target_id, target_value, and target_count. For a categorical data.frame, counts are converted to categories: ref-only -> 0, alt-only -> 1, both present -> 0.5, zero total -> NA. Matrix and categorical file inputs (e.g. *_cat.txt) remain supported.
model	Observation model to use. Options: "categorical", "poisson", "binomial", "negative_binomial" (default).
n_mcmc	Number of MCMC iterations (default: 10000).
burnin	Burn-in period. If NULL, defaults to n_mcmc/2.
alpha	IBP concentration parameter (default: 2.6).
rho	Dictionary sparsity parameter (default: 0.5 for categorical model and NULL otherwise, which means use the global minor allele frequency)
threshold	Threshold for identifying single infections (default: 0.001).
gap	Early stopping threshold. If NULL, runs for full n_mcmc iterations.
seed	Random seed for reproducibility.
verbose	Whether to print progress information (default: TRUE).
log_performance	Whether to log performance metrics (default: FALSE).
store_mcmc	Whether to store full MCMC samples (default: FALSE).
...	Additional model-specific parameters.
e1	Error parameter for categorical model (default: 0.05)
e2	Error parameter for categorical model (default: 0.05)

Value

An object of class snp_slice_results containing:

- allocation_matrix: Binary allocation matrix (A)
- dictionary_matrix: Binary dictionary matrix (D)
- mcmc_samples: MCMC samples (if store_mcmc = TRUE)
- diagnostics: Convergence diagnostics
- parameters: Model parameters used
- model_info: Model specification

Examples

```
## Not run:
# Example with read count data
data <- list(
  read1 = matrix(c(10, 5, 15, 8), nrow = 2),
```

```
  read0 = matrix(c(90, 95, 85, 92), nrow = 2)
)

result <- snp_slice(data, model = "negative_binomial", n_mcmc = 1000)

# Extract results
strains <- extract_strains(result)
allocations <- extract_allocations(result)

## End(Not run)
```

`summary.snp_slice_results` *Print summary of SNP-Slice results*

Description

Print summary of SNP-Slice results

Usage

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

Arguments

`object`	SNP-Slice results object
`...`	Additional arguments

Value

Summary information

Index