Package ‘avseqmc’

May 7, 2026

Title Anytime-Valid Sequential Estimation of Monte-Carlo p-Values

Version 1.0.2

Description Anytime-valid sequential estimation of the p-value of a test calibrated by Monte-Carlo simulation, as described in Stoepker & Castro (2024) <doi:10.48550/arXiv.2409.18908>.

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.2

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Author Ivo V. Stoepker [aut, cre],
Rui M. Castro [aut]

Maintainer Ivo V. Stoepker <`i.v.stoepker@tue.nl`>

NeedsCompilation no

Repository CRAN

Date/Publication 2025-09-01 13:10:02 UTC

Contents

`avseqmc` *Anytime-valid sequential estimation of the p-value of a Monte-Carlo test*

Description

`avseqmc()` performs anytime-valid sequential estimation of the p-value of a Monte-Carlo test as described in Stoepker and Castro (2024, Definition 6). Subsequent references to equations and sections in this section of the reference manual pertain to this paper. The sequential p-value estimate is based on the construction of Definition 6 (i.e. through the confidence sequence by Robbins (1970)).

For first-time usage, it may be helpful to follow the examples in the package vignette via `vignette("Introduction",packag`

Usage

```
avseqmc(
  sample_G,
  epsilon = NULL,
  stopcrit = list(type = "futility", param = 0.05),
  min_samples = 0,
  max_samples = max(1000, min_samples),
  compute_lower = FALSE
)
```

Arguments

`sample_G`	Either: a function (without arguments) that draws one (or a batch of) zero/one samples from the distribution G as in Equation (5), where the function returns a vector of zeroes and ones; or an object from class 'avseqmc_progress' containing earlier progress from anytime-valid estimation of the p-value. The function `avseqmc()` returns such an object, or the object can be constructed manually using function `init_avseqmc_progress()`.
`epsilon`	The desired risk of overestimated significance. Ignored if sample_G is an object of class `avseqmc_progress` and mandatory otherwise.
`stopcrit`	The desired stopping criterion. Can use one of the two pre-defined stopping criteria from Section 4.1 as follows (with respect to the notation used in that section): • `list("type" = "futility", "param" = alpha)` • `list("type" = "convergence", "param" = c(gamma, n_0))` Alternatively, a custom function may be provided that takes an `avseqmc_progress` object as input and returns `FALSE` if sampling should continue for another batch.
`min_samples`	Minimum number of Monte-Carlo samples before returning the current p-value estimate. Defaults to 0.
`max_samples`	Maximum number of Monte-Carlo samples before returning the current p-value estimate. Defaults to `max(1000, min_samples)`.

`compute_lower`	Boolean; if `TRUE`, the lower confidence sequence limit with significance level epsilon is computed after each batch of Monte-Carlo samples, based on the construction by Robbins (1970). Since it is used in the evaluation of the futility stopping criterion (i.e. `stopcrit = list("type"="futility","param"=...)`) it is automatically computed when this stopping criterion is selected.

Value

An object of class `avseqmc_progress` containing the progress of the sequentially estimated p-value. The object is a list containing the following elements:

- `$epsilon`: risk of overestimated significance used in the sequential estimation.
- `$sample_G`: function that samples (batches) from the Monte-Carlo distribution $G^*(X)$ as in Equation (5).
- `$ptilde`: sequence of sequential p-value estimates. The final value in this sequence is the most recent estimate of the p-value.
- `$Ltilde`: sequence of lower bounds of the confidence sequence based on the construction by Robbins (1970). Contains NA values if these were not computed by default through `stopcrit = list("type"="futility","param"=...)` or requested using `compute_lower=TRUE`.
- `$n`: total number of samples drawn from the MC sampler.
- `$S`: total number of ones observed from the MC sampler.
- `$B`: sequence of number of ones observed at each sampling timepoint (which can be greater than 1 if `sample_G` samples in batches)
- `$Bn`: sequence of number of samples drawn from MC sampler at each timepoint (which can be greater than 1 if `sample_G` samples in batches)

References

Stoepker, I. V., and R. M. Castro. 2024. Inference with Sequential Monte-Carlo Computation of p-Values: Fast and Valid Approaches. https://doi.org/10.48550/arXiv.2409.18908.

Robbins, H. (1970). Statistical Methods Related to the Law of the Iterated Logarithm. The Annals of Mathematical Statistics, 41(5):1397–1409. http://dx.doi.org/10.1214/aoms/1177696786

See Also

`init_avseqmc_progress` which can be used if one wishes to resume progress based on earlier reported p-values estimated by Monte-Carlo simulation.

Examples

```
# Minimal example using defaults:
set.seed(123)
library(avseqmc)
G1 <- function(){runif(1) < 0.01} # A mock MC function to demonstrate functionality
R1 <- avseqmc(sample_G = G1, epsilon = 0.001)
print(R1)

# Minimal example to resuming earlier estimation:
G2 <- function(){runif(1) < 0.03}
```

```
R2a <- avseqmc(sample_G = G2, epsilon = 0.001)
print(R2a)
R2b <- avseqmc(R2a)
print(R2b)

# Using built-in convergence stopping time:
G3 <- function(){runif(1) < 0.04}
R3 <- avseqmc(sample_G = G3,
              epsilon = 0.001,
              stopcrit = list("type" = "convergence", param = c(1e-5, 100)))

# Batch sampling example (drawing batches of size 50)
G4 <- function(){runif(50) < 0.04}
R4 <- avseqmc(sample_G = G4, epsilon = 0.001)
print(R4)
```

`init_avseqmc_progress` *Initialize an* `avseqmc_progress` *Object*

Description

Initialize an object of class `avseqmc_progress`. This object contains the progress of an earlier sequentially anytime-valid estimated p-value from a Monte-Carlo simulation as described in Stoepker and Castro (2024, Definition 6). Subsequent references to equations and sections in this section of the reference manual pertain to this paper.

The object can be passed to `avseqmc()` to resume sampling.

Usage

```
init_avseqmc_progress(sample_G, epsilon, ptilde = NULL, n = 0, S = 0)
```

Arguments

`sample_G`	A function (without arguments) that draws one (or a batch of) zero/one samples from the distribution G as in Equation (5), where the function returns a vector of zeroes and ones.
`epsilon`	The desired risk of overestimated significance.
`ptilde`	Optional: a previously computed anytime-valid p-value estimate (via the construction in Definition 6, based on Robbins (1970)). If supplied, `n` must also be provided. The number of previously observed ones observed from the Monte-Carlo sampler `sample_G` (`S`) is inferred from `ptilde` and `n` unless given explicitly. If `ptilde` is rounded, supplying `S` is recommended to avoid numerical inaccuracies in inferring it.
`n`	Optional: The number of Monte-Carlo samples used to compute the supplied `ptilde`. Must be given if `ptilde` or `S` is provided.
`S`	Optional: The number of ones observed from `sample_G`. Must be given when `n` is supplied. Can be used instead of inferring it from `ptilde` and `n`.

Value

An object of class `avseqmc_progress` containing the progress of the sequentially estimated p-value. The object is a list containing the following elements:

- `$epsilon`: risk of overestimated significance used in the sequential estimation.
- `$sample_G`: function that samples (batches) from the Monte-Carlo distribution $G^*(X)$ as in Equation (5).
- `$ptilde`: sequence of sequential p-value estimates. The final value in this sequence is the most recent estimate of the p-value.
- `$Ltilde`: sequence of lower bounds of the confidence sequence based on the construction by Robbins (1970). Contains NA values if these were not computed by default through `stopcrit = list("type"="futility","param"=...)` or requested using `compute_lower=TRUE`.
- `$n`: total number of samples drawn from the MC sampler.
- `$S`: total number of ones observed from the MC sampler.
- `$B`: sequence of number of ones observed at each sampling timepoint (which can be greater than 1 if `sample_G` samples in batches)
- `$Bn`: sequence of number of samples drawn from MC sampler at each timepoint (which can be greater than 1 if `sample_G` samples in batches)

If `ptilde`, `n`, and `S` are all empty, the object is initialized but empty and can be used as a starting point for sequential estimation.

References

Stoepker, I. V., and R. M. Castro. 2024. Inference with Sequential Monte-Carlo Computation of p-Values: Fast and Valid Approaches. https://doi.org/10.48550/arXiv.2409.18908.

Robbins, H. (1970). Statistical Methods Related to the Law of the Iterated Logarithm. The Annals of Mathematical Statistics, 41(5):1397–1409. http://dx.doi.org/10.1214/aoms/1177696786

See Also

`avseqmc` which can be used to resume the anytime-valid sampling as logged in the constructed object.

Examples

```
# Minimal example to construct an object based on earlier values of n and S
G1 <- function(){runif(1) < 0.04}
R1 <- init_avseqmc_progress(sample_G = G1,
                            epsilon = 0.001,
                            n = 1000,
                            S = 44)
```

`plot.avseqmc_progress` *Plot the results of the sequential p-value estimation.*

Description

Plot the results of the sequential p-value estimation.

Usage

```
## S3 method for class 'avseqmc_progress'
plot(x, ...)
```

Arguments

`x`	an object of class avseqmc_progress containing progress on the sequential estimation.
`...`	Ignored.

Value

No return value, called for side effects.

`print.avseqmc_progress` *Print the results of the sequential p-value estimation.*

Description

Print the results of the sequential p-value estimation.

Usage

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

Arguments

`x`	an object of class avseqmc_progress containing progress on the sequential estimation.
`...`	Ignored.

Value

No return value, called for side effects.

Index