Package ‘diagis’

June 4, 2020

Type Package
Title Diagnostic Plot and Multivariate Summary Statistics of Weighted Samples from Importance Sampling
Version 0.1.5
Date 2020-06-04
Description Fast functions for effective sample size, weighted multivariate mean and variance computation, and weight diagnostic plot for generic importance sampling type results.
License GPL (>= 2)
BugReports https://github.com/helske/diagis/issues
Suggests testthat, knitr, rmarkdown
Imports Rcpp (>= 0.12.7), ggplot2 (>= 2.1.0), gridExtra, coda
LinkingTo Rcpp, RcppArmadillo
RoxygenNote 7.1.0
VignetteBuilder knitr
Encoding UTF-8
NeedsCompilation yes
Author Jouni Helske [aut, cre] (<https://orcid.org/0000-0001-7130-793X>)
Maintainer Jouni Helske <jouni.helske@iki.fi>
Repository CRAN
Date/Publication 2020-06-04 13:30:02 UTC

R topics documented:

  diagis ................................................................. 2
  ess ................................................................. 3
  running_ess ....................................................... 3
  running_mean ...................................................... 4
  running_var ....................................................... 4
  running_weighted_mean .......................................... 5
# Simple importance sampling example
# true distribution is a standard normal:
p <- function(x) dnorm(x)
# proposal distribution is normal with sd s
q <- function(x, s) dnorm(x, 0, s)

# IS weights have finite variance only if s^2 > 1/2
# variance is s/(2-1/s^2)^(3/2)

# Optimal case
set.seed(42)
s_opt <- sqrt(2)
x_opt <- rnorm(1000, sd = s_opt)
w_opt <- p(x_opt) / q(x_opt, s_opt)
weighted_mean(x_opt, w_opt)
weighted_var(x_opt, w_opt)

s_inf <- 0.25
x_inf <- rnorm(1000, sd = s_inf)
w_inf <- p(x_inf) / q(x_inf, s_inf)
weighted_mean(x_inf, w_inf) #!!
weighted_var(x_inf, w_inf) #!!

# Diagnostic plots
weight_plot(w_inf)
weight_plot(w_opt)
## ess

### Effective sample size

**Description**

Computes the effective sample size (ESS) of importance sampling estimator.

**Usage**

`ess(w, f, x)`

**Arguments**

- `w`: A numeric vector of non-negative weights.
- `f`: A function used in computing $f$-specific ESS.
- `x`: A numeric vector of samples used to generate `w`. Used for computing $f(x)$.

**Value**

An effective sample size estimate.

## running_ess

### Running effective sample size

**Description**

Computes and returns the running estimate of effective sample size (ESS) of importance sampling estimator.

**Usage**

`running_ess(w, f, x)`

**Arguments**

- `w`: A numeric vector of non-negative weights.
- `f`: A function used in computing $f$-specific ESS.
- `x`: A numeric vector of samples used to generate `w`. Used for computing $f(x)$.

**Value**

An effective sample size estimate.
running_mean

Description
Computes running mean of a vector or matrix, returning the values from each step.

Usage
running_mean(x, na.rm)

Arguments
- x: A numeric vector, matrix, three dimensional array, or an mcmc object from the coda package. For matrix, the mean is computed for each column, and for array the sweep is done over the third dimension.
- na.rm: If TRUE, NA values in x are omitted from the computation. Default is FALSE.

Value
A vector containing the recursive mean estimates.

running_var

Description
Computes running variance of a vector, returning the values from each step.

Usage
running_var(x, method = c("moment", "unbiased"), na.rm = FALSE)

Arguments
- x: A numeric vector or object that can be coerced to such.
- method: Estimator type, either "moment" (default) or "unbiased", which is unbiased only in case of frequency weights.
- na.rm: If TRUE, NA values in x are omitted from the computation. Default is FALSE.

Value
A vector containing the recursive variance estimates.
running_weighted_mean  Running weighted mean

Description

Computes running weighted mean of a vector or matrix, returning the values from each step.

Usage

running_weighted_mean(x, w, na.rm)

Arguments

x A numeric vector, matrix, three dimensional array, or an mcmc object from the coda package. For matrix, the mean is computed for each column, and for array the sweep is done over the third dimension.

w A numeric vector of non-negative weights. Will be automatically normalised to sum to one.

na.rm If TRUE, NA values in x (and corresponding weights in w) are omitted from the computation. Default is FALSE. Only used in vector methods.

Value

A vector containing the recursive weighted mean estimates.

running_weighted_var  Running weighted variance of a vector

Description

Computes running weighted variance of a vector, returning the values from each step.

Usage

running_weighted_var(x, w, method = c("moment", "unbiased"), na.rm = FALSE)

Arguments

x A numeric vector or object that can be coerced to such.

w A numeric vector of non-negative weights. Will be automatically normalised to sum to one.

method Estimator type, either "moment" (default) or "unbiased", which is unbiased only in case of frequency weights.

na.rm If TRUE, NA values in x (and corresponding weights in w) are omitted from the computation. Default is FALSE.
**Value**

A vector containing the recursive weighted variance estimates.

---

**weighted_mean**

*Weighted mean*

**Description**

Computes a weighted mean of a vector, matrix, or a three dimensional array.

**Usage**

`weighted_mean(x, w, na.rm)`

**Arguments**

- `x`: A numeric vector, matrix, three dimensional array, or an `mcmc` object from the `coda` package. For matrix, the mean is computed for each column, and for array the sweep is done over the third dimension.
- `w`: A numeric vector of non-negative weights. Will be automatically normalised to sum to one.
- `na.rm`: If `TRUE`, NA values in `x` (and corresponding weights in `w`) are omitted from the computation. Default is `FALSE`. Only used in vector methods.

**Value**

A weighted mean.

---

**weighted_se**

*Weighted standard error*

**Description**

Computes a weighted standard error of a vector or matrix.

**Usage**

`weighted_se(x, w, na.rm)`

**Arguments**

- `x`: A numeric vector or matrix.
- `w`: A numeric vector of non-negative weights. Will be automatically normalised to sum to one.
- `na.rm`: If `TRUE`, NA values in `x` (and corresponding weights in `w`) are omitted from the computation. Default is `FALSE`. 
**weighted_var**

**Value**

A weighted variance.

**Note**

Compared to some other R functions, here the weights are regarded as probability weights, not frequency weights.

---

### Description

Computes a weighted variance/covariance of a vector, matrix or a three dimensional array.

### Usage

```r
weighted_var(x, w, method, na.rm)
```

### Arguments

- **x**
  A numeric vector, matrix or three dimensional array. For matrix, covariances are computed between columns. For array, marginal covariances are computed for each column, i.e. for $m \times n \times k$ array function returns $m \times m \times n$ array.

- **w**
  A numeric vector of non-negative weights. Will be automatically normalised to sum to one.

- **method**
  Estimator type, either "moment" (default) or "unbiased", which is unbiased only in case of frequency weights.

- **na.rm**
  If TRUE, NA values in x (and corresponding weights in w) are omitted from the computation. Default is FALSE.

### Value

A weighted variance.

### Note

Compared to some other R functions, here the weights are regarded as probability weights, not frequency weights.
weight_plot

Diagnostic plot of importance sampling weights

Description

Function weight_plot plots four figures given the weight vector \( w \): Plot of largest weights, sorted graph of all weights, running variance estimate of weights, and running effective sample size estimate of weights.

Usage

weight_plot(w)

Arguments

\( w \)

Vector of weights.

Examples

```r
# importance sampling from too narrow distribution
# weights have infinite variance
set.seed(1)
x_inf <- rnorm(1000, sd = 0.1)
w_inf <- dnorm(x_inf) / dnorm(x_inf, 0, 0.1)
weight_plot(w_inf)
x_opt <- rnorm(1000, sd = sqrt(2))
w_opt <- dnorm(x_opt) / dnorm(x_opt, 0, sqrt(2))
weight_plot(w_opt)
```
Index

diagis, 2
ess, 3
running_ess, 3
running_mean, 4
running_var, 4
running_weighted_mean, 5
running_weighted_var, 5
weight_plot, 8
weighted_mean, 6
weighted_se, 6
weighted_var, 7