Package 'MYIS'

August 5, 2026

Type Package

Title 'Moreau-Yosida' Importance Sampling for Statistical Inference

Version 0.1.0

Description Implements 'Moreau-Yosida' Markov chain Monte Carlo ('MCMC') importance sampling for parameter estimation and Bayesian inference under smooth, non-differentiable, or light-tailed target posterior distributions and arbitrary probability models with complete or censored data. Users supply user-defined probability density functions, optional distribution functions, parameter ranges, and observations subject to complete, right, left, interval, Type-I, Type-II, progressive Type-II, first-failure, or truncation schemes. Constructs 'Moreau-Yosida' envelopes, gradient-based proposals ('MALA', 'HMC', or 'RWM'), self-normalized importance weights, batch-means asymptotic variance estimates, and Bayesian marginal quantiles. Methodologies are based on 'Shukla', 'Vats', and 'Chi' (2025) <doi:10.48550/arXiv.2501.02228>, 'Pereyra' (2016) <doi:10.1111/sjos.12208>, 'Durmus' and others (2022) <doi:10.1214/22-EJS2027>, 'Chen' and 'Shao' (1999) <doi:10.1214/ss/1009211804>, 'Roberts' and 'Rosenthal' (1998) <doi:10.1214/aoap/1028903378>, 'Geweke' (1989) <doi:10.2307/2290062>, 'Hesterberg' (1995) <doi:10.1080/00031305.1995.10476138>, and 'Balakrishnan' and 'Aggarwala' (2000, ISBN:978-0-8176-4001-9).

License GPL-3

Encoding UTF-8

Language en-US

Depends R (>= 4.1.0)

Imports stats, graphics

Suggests testthat (>= 3.0.0), knitr, rmarkdown

Config/testthat/edition 3

RoxygenNote 7.3.3

VignetteBuilder knitr

NeedsCompilation no

Author Shikhar Tyagi [aut, cre] (ORCID:
<https://orcid.org/0000-0003-1606-0844>),
Arvind Pandey [aut],

Bhupendra Singh [aut],
Vrijesh Tripathi [aut]

Maintainer Shikhar Tyagi <shikhar1093tyagi@gmail.com>

Repository CRAN

Date/Publication 2026-08-05 07:50:12 UTC

Contents

`batch_means_cov` *Compute Batch-Means Asymptotic Covariance Matrix*

Description

Estimates the asymptotic covariance matrix of a Markov chain output using consistent batch-means methodology (Vats et al., 2019).

Usage

```
batch_means_cov(S_matrix, batch_size = NULL)
```

Arguments

`S_matrix`	Numeric matrix of size `n` x `p` containing Markov chain observations.
`batch_size`	Integer batch size. If `NULL`, defaults to `floor(sqrt(n))`.

Value

A `p` x `p` numeric matrix representing the batch-means estimated asymptotic covariance.

Examples

```
set.seed(123)
data_mat <- matrix(rnorm(1000), ncol = 2)
cov_bm <- batch_means_cov(data_mat)
```

`build_negative_log_posterior` *Construct Negative Log-Posterior Potential Function*

Description

Constructs the negative log-posterior potential function $\psi(\theta) = -\log L(\theta; \text{data}) - \log \pi(\theta)$ for a user-supplied PDF (and optional CDF / Survival function), data, censoring scheme, and prior.

Usage

```
build_negative_log_posterior(
  pdf,
  cdf = NULL,
  surv = NULL,
  data,
  censoring = "complete",
  censoring_params = list(),
  par_lower = -Inf,
  par_upper = Inf,
  log_prior = NULL
)
```

Arguments

`pdf`	Function taking variable `x` and parameter vector `theta`, returning the probability density value.
`cdf`	Optional function taking variable `x` and parameter vector `theta`, returning cumulative probability.
`surv`	Optional function taking variable `x` and parameter vector `theta`, returning survival probability.
`data`	Numeric vector, matrix, or list containing the observed dataset.
`censoring`	Character string specifying the censoring scheme. Options include: `"complete"`, `"right"`, `"left"`, `"interval"`, `"type1"`, `"type2"`, `"progressive_type2"`, `"middle"`, or `"truncated"`.
`censoring_params`	Named list of additional parameters for censoring schemes (e.g. `delta`, `R`, `trunc_left`, `trunc_right`).
`par_lower`	Numeric vector of lower parameter bounds.
`par_upper`	Numeric vector of upper parameter bounds.
`log_prior`	Optional function taking parameter vector `theta` and returning scalar log-prior density.

Value

A function of `theta` returning scalar negative log-posterior $\psi(\theta)$.

Examples

```
my_pdf <- function(x, th) dexp(x, rate = th)
psi_fn <- build_negative_log_posterior(my_pdf, data = c(0.5, 1.2, 2.1), par_lower = 0.001)
psi_fn(1.0)
```

---

effective_sample_size_is

Compute Importance Sampling Effective Sample Size

Description

Calculates the effective sample size (ne) for importance sampling based on Kong (1992).

Usage

```
effective_sample_size_is(weights)
```

Arguments

`weights`	Numeric vector of non-negative importance weights.

Value

A named list containing `ne` (effective sample size) and `ratio` (`ne/n`).

Examples

```
set.seed(123)
w <- runif(100)
ess_info <- effective_sample_size_is(w)
```

`my_envelope` *Moreau-Yosida Envelope and Proximal Mapping Computation*

Description

Computes the Moreau-Yosida envelope value, proximal mapping point, gradient, and unnormalized importance weights for a given convex or log-concave function $\psi(\theta)$ following Shukla, Vats, and Chi (2025).

Usage

```
my_envelope(
  psi_fn,
  theta,
  lambda,
  par_lower = -Inf,
  par_upper = Inf,
  method = NULL
)
```

Arguments

`psi_fn`	Function taking a numeric parameter vector `theta` and returning a scalar numeric value representing $\psi(\theta)$.
`theta`	Numeric vector of parameter values at which to evaluate the envelope.
`lambda`	Positive numeric smoothing parameter $\lambda > 0$.
`par_lower`	Numeric vector of lower bounds for parameters. Defaults to `-Inf`.
`par_upper`	Numeric vector of upper bounds for parameters. Defaults to `Inf`.
`method`	Character string specifying the optimization algorithm for `optim`. Defaults to `"L-BFGS-B"` or `"BFGS"`.

Value

A list with the following components:

`prox`	Numeric vector containing the proximal mapping point $\text{prox}_{\lambda\psi}(\theta)$.
`psi_lambda`	Scalar numeric value of the Moreau-Yosida envelope $\psi_\lambda(\theta)$.
`psi_orig`	Scalar numeric value of the original function $\psi(\theta)$.
`grad_psi_lambda`	Numeric vector representing the gradient of the envelope $\nabla\psi_\lambda(\theta)$.
`grad_log_pi_lambda`	Numeric vector representing $\nabla \log \pi_\lambda(\theta) = -\nabla\psi_\lambda(\theta)$.
`weight`	Scalar numeric value of the unnormalized importance weight $w_\lambda(\theta) \in (0, 1]$.

References

Shukla, A., Vats, D., & Chi, E. C. (2025). MCMC Importance Sampling via Moreau-Yosida Envelopes. arXiv:2501.02228v2.

Examples

```
psi <- function(th) sum(abs(th)) # Laplace potential
res <- my_envelope(psi, theta = c(1.5, -0.8), lambda = 0.5)
res$prox
res$weight
```

`my_is_estimate` *Moreau-Yosida MCMC Importance Sampling Estimator*

Description

Computes Moreau-Yosida MCMC Importance Sampling (MY-IS) estimates, standard errors, asymptotic covariance, Bayesian marginal quantiles, and effective sample size for user-defined probability density functions under complete or censored observations (Shukla, Vats, and Chi, 2025).

Usage

```
my_is_estimate(
  pdf,
  cdf = NULL,
  surv = NULL,
  data,
  initial_theta,
  censoring = "complete",
  censoring_params = list(),
  par_lower = -Inf,
  par_upper = Inf,
  log_prior = NULL,
  sampler = c("mala", "hmc", "rwm"),
  n_samples = 3000,
  burnin = 500,
  lambda = NULL,
  tune_lambda = TRUE,
  target_ne_ratio = 0.6,
  conf_level = 0.95
)
```

Arguments

`pdf`	User-defined function taking variable `x` and numeric parameter vector `theta`, returning probability density value.

`cdf`	Optional function taking variable `x` and parameter vector `theta`, returning cumulative distribution function value.
`surv`	Optional function taking variable `x` and parameter vector `theta`, returning survival function value.
`data`	Numeric vector, matrix, or data frame of observations.
`initial_theta`	Numeric vector of initial parameter values.
`censoring`	Character string specifying censoring scheme. Supported schemes: `"complete"`, `"right"`, `"left"`, `"interval"`, `"type1"`, `"type2"`, `"progressive_type2"`, or `"truncated"`. Defaults to `"complete"`.
`censoring_params`	Named list of scheme options (e.g. `delta`, `R`, `trunc_left`, `trunc_right`).
`par_lower`	Numeric vector of lower parameter bounds. Defaults to `-Inf`.
`par_upper`	Numeric vector of upper parameter bounds. Defaults to `Inf`.
`log_prior`	Optional function taking parameter vector `theta` and returning scalar log-prior.
`sampler`	Character string specifying MCMC algorithm for π_λ: `"mala"` (default), `"hmc"`, or `"rwm"`.
`n_samples`	Integer number of retained MCMC samples. Defaults to `3000`.
`burnin`	Integer number of burn-in samples. Defaults to `500`.
`lambda`	Optional positive numeric smoothing parameter $\lambda > 0$. If `NULL`, automatically tuned.
`tune_lambda`	Logical flag indicating whether to perform automatic λ tuning. Defaults to `TRUE`.
`target_ne_ratio`	Numeric target ratio for importance sampling effective sample size n_e/n. Defaults to `0.6`.
`conf_level`	Numeric confidence level for Bayesian credible intervals in `(0, 1)`. Defaults to `0.95`.

Value

An S3 object of class `"myis"` containing:

`estimates`	Named vector of MY-IS parameter point estimates $\hat{\theta}_n^{\mathrm{MY}}$.
`se`	Named vector of batch-means standard errors.
`cov_mat`	Asymptotic covariance matrix estimate $\hat{\Xi}_{\mathrm{BM}}/n$.
`cred_intervals`	Matrix containing lower and upper bounds of Bayesian credible intervals.
`quantiles`	Matrix of weighted quantiles calculated via Chen & Shao (1999).
`ne_info`	List containing effective sample size `ne` and ratio `ratio` (n_e/n).
`accept_rate`	Numeric MCMC acceptance rate.
`lambda`	Smoothing parameter λ used.
`samples`	Numeric matrix of generated MCMC states.
`weights`	Numeric vector of unnormalized importance weights.
`sampler`	Character string of sampler used.
`censoring`	Character string of censoring scheme used.
`call`	Matched function call.

References

Shukla, A., Vats, D., & Chi, E. C. (2025). MCMC Importance Sampling via Moreau-Yosida Envelopes. arXiv:2501.02228v2. Chen, M. H., & Shao, Q. M. (1999). Monte Carlo estimation of Bayesian credible quantities. Journal of Computational and Graphical Statistics, 8(1), 69-92.

Examples

```
set.seed(42)
sample_data <- rexp(30, rate = 1.5)
my_pdf <- function(x, th) dexp(x, rate = th[1])
fit <- my_is_estimate(
  pdf = my_pdf,
  data = sample_data,
  initial_theta = 1.0,
  par_lower = 0.001,
  n_samples = 100,
  burnin = 20
)
print(fit)
```

---

plot.myis *Diagnostic Plots for Moreau-Yosida Importance Sampling*

Description

Generates trace plots, autocorrelation functions (ACF), posterior density/histograms, and weight distribution plots for MCMC importance sampling outputs.

Usage

```
## S3 method for class 'myis'
plot(
  x,
  type = c("all", "trace", "acf", "density", "weights"),
  par_idx = 1,
  ...
)
```

Arguments

x	An S3 object of class "myis".
type	Character string specifying plot type: "all" (default), "trace", "acf", "density", or "weights".
par_idx	Integer parameter index to plot. Defaults to 1.
...	Additional graphical parameters.

Value

No return value, called for side effects.

Examples

```
set.seed(123)
dat <- rexp(20, rate = 2)
fit <- my_is_estimate(
  dexp, data = dat, initial_theta = 1,
  n_samples = 100, burnin = 20
)
plot(fit, type = "all")
```

---

print.myis *Print Moreau-Yosida Importance Sampling Results*

Description

Print method for S3 objects of class `"myis"`.

Usage

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

Arguments

`x`	An S3 object of class `"myis"`.
`...`	Additional arguments passed to print.

Value

Returns `x` invisibly.

Examples

```
set.seed(123)
dat <- rexp(20, rate = 2)
fit <- my_is_estimate(
  dexp, data = dat, initial_theta = 1,
  n_samples = 100, burnin = 20
)
print(fit)
```

print.summary.myis *Print Summary of Moreau-Yosida Importance Sampling Object*

Description

Print method for summary objects of class "summary.myis".

Usage

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

Arguments

x	An S3 object of class "summary.myis".
...	Additional arguments.

Value

Returns x invisibly.

safe_exp *Numerically Safe Exponential*

Description

Computes exponential with an upper cap to prevent numerical overflow to Inf.

Usage

```
safe_exp(x, max_val = 700)
```

Arguments

x	Numeric scalar, vector, or matrix.
max_val	Numeric maximum allowed exponent. Defaults to 700.

Value

Numeric output after applying safe exponential.

Examples

```
safe_exp(c(-5, 10, 1000))
```

safe_log *Numerically Safe Logarithm*

Description

Computes the logarithm with a lower bound threshold to avoid `-Inf` or NaN errors.

Usage

```
safe_log(x, min_val = 1e-300)
```

Arguments

`x`	Numeric scalar, vector, or matrix.
`min_val`	Numeric threshold value. Defaults to `1e-300`.

Value

Numeric output after applying safe logarithm.

Examples

```
safe_log(c(0, 0.5, 2))
```

sample_pi_lambda_hmc *Hamiltonian Monte Carlo targeting Moreau-Yosida Density*

Description

Simulates a Markov chain targeting the Moreau-Yosida importance density $\pi_\lambda(\theta) \propto \exp(-\psi_\lambda(\theta))$ using Hamiltonian Monte Carlo (Livingstone et al., 2019; Chaari et al., 2016).

Usage

```
sample_pi_lambda_hmc(
  psi_fn,
  initial_theta,
  lambda,
  n_samples = 2000,
  burnin = 500,
  step_size = NULL,
  n_leapfrog = 10,
  par_lower = -Inf,
  par_upper = Inf
)
```

Arguments

psi_fn	Function taking parameter vector theta and returning scalar $\psi(\theta)$.
initial_theta	Numeric vector of initial parameter values.
lambda	Positive numeric smoothing parameter $\lambda > 0$.
n_samples	Integer number of MCMC samples to retain.
burnin	Integer number of initial burn-in iterations.
step_size	Optional positive leapfrog step size epsilon. If NULL, automatically tuned.
n_leapfrog	Integer number of leapfrog steps L. Defaults to 10.
par_lower	Numeric vector of lower parameter bounds.
par_upper	Numeric vector of upper parameter bounds.

Value

A list containing samples, weights, accept_rate, and step_size.

Examples

```
psi <- function(th) sum(th^2)
res <- sample_pi_lambda_hmc(
  psi, initial_theta = c(1, 1),
  lambda = 0.1, n_samples = 100, burnin = 20
)
```

---

sample_pi_lambda_mala *Metropolis-Adjusted Langevin Algorithm targeting Moreau-Yosida Density*

Description

Simulates a Markov chain targeting the Moreau-Yosida importance density $\pi_\lambda(\theta) \propto \exp(-\psi_\lambda(\theta))$ using the MALA proposal (Shukla, Vats, and Chi, 2025; Pereyra, 2016).

Usage

```
sample_pi_lambda_mala(
  psi_fn,
  initial_theta,
  lambda,
  n_samples = 2000,
  burnin = 500,
  step_size = NULL,
  par_lower = -Inf,
  par_upper = Inf
)
```

Arguments

psi_fn	Function taking parameter vector theta and returning scalar $\psi(\theta)$.
initial_theta	Numeric vector of initial parameter values.
lambda	Positive numeric smoothing parameter $\lambda > 0$.
n_samples	Integer number of MCMC samples to retain.
burnin	Integer number of initial burn-in iterations.
step_size	Optional positive step size h. If NULL, automatically tuned.
par_lower	Numeric vector of lower parameter bounds.
par_upper	Numeric vector of upper parameter bounds.

Value

A list containing samples (matrix of MCMC states), weights (vector of importance weights), accept_rate (acceptance rate), and step_size (used step size).

Examples

```
psi <- function(th) sum(th^2)
res <- sample_pi_lambda_mala(
  psi, initial_theta = c(1, 1),
  lambda = 0.1, n_samples = 100, burnin = 20
)
```

---

sample_pi_lambda_rwm *Random Walk Metropolis targeting Moreau-Yosida Density*

Description

Simulates a Markov chain targeting the Moreau-Yosida importance density $\pi_\lambda(\theta)$ using Random Walk Metropolis proposal.

Usage

```
sample_pi_lambda_rwm(
  psi_fn,
  initial_theta,
  lambda,
  n_samples = 2000,
  burnin = 500,
  proposal_sd = NULL,
  par_lower = -Inf,
  par_upper = Inf
)
```

Arguments

psi_fn	Function taking parameter vector theta and returning scalar $\psi(\theta)$.
initial_theta	Numeric vector of initial parameter values.
lambda	Positive numeric smoothing parameter $\lambda > 0$.
n_samples	Integer number of MCMC samples to retain.
burnin	Integer number of initial burn-in iterations.
proposal_sd	Optional positive proposal standard deviation. If NULL, tuned automatically.
par_lower	Numeric vector of lower parameter bounds.
par_upper	Numeric vector of upper parameter bounds.

Value

A list containing samples, weights, accept_rate, and proposal_sd.

Examples

```
psi <- function(th) sum(th^2)
res <- sample_pi_lambda_rwm(
  psi, initial_theta = c(1, 1),
  lambda = 0.1, n_samples = 100, burnin = 20
)
```

---

summary.myis *Summary of Moreau-Yosida Importance Sampling Object*

Description

Summary method for S3 objects of class "myis".

Usage

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

Arguments

object	An S3 object of class "myis".
...	Additional arguments.

Value

An S3 object of class "summary.myis".

Examples

```
set.seed(123)
dat <- rexp(20, rate = 2)
fit <- my_is_estimate(
  dexp, data = dat, initial_theta = 1,
  n_samples = 100, burnin = 20
)
summary(fit)
```

`weighted_quantiles` *Compute Weighted Quantiles (Chen and Shao Method)*

Description

Calculates weighted marginal quantiles from MCMC importance sampling outputs following the estimator proposed by Chen and Shao (1999).

Usage

```
weighted_quantiles(samples, weights, probs = c(0.025, 0.5, 0.975))
```

Arguments

`samples`	Numeric matrix of samples (iterations x components) or numeric vector.
`weights`	Numeric vector of non-negative importance weights corresponding to each row.
`probs`	Numeric vector of desired quantile probabilities in `(0, 1)`.

Value

A matrix or vector of estimated weighted quantiles.

Examples

```
set.seed(123)
x <- rnorm(100)
w <- exp(-0.1 * x^2)
q <- weighted_quantiles(x, w, probs = c(0.025, 0.5, 0.975))
```

Index