Package 'UniIS'

August 6, 2026

Type Package

Title Importance Sampling Inference for Censored Univariate Data

Version 0.1.0

Description Distribution-independent framework for importance-sampling inference with univariate observations subject to censoring or truncation. Users provide probability functions and a proposal over model parameters. Constructs observed-data likelihood contributions, computes numerically stable importance weights, and supplies posterior, likelihood, predictive, diagnostic, and model-comparison summaries. Covers complete, right, left, interval, Type-I, Type-II, progressive Type-II, first-failure, progressive first-failure, doubly Type-II, middle-censored, and left/right-truncated data. Methods for importance sampling and censoring schemes are described in Geweke (1989) <doi:10.2307/2290062>, Hesterberg (1995) <doi:10.1080/00031305.1995.10476138>, Robert and Casella (2004, ISBN:978-0-387-21617-1), Kundu and Joarder (2006) <doi:10.1016/j.csda.2005.05.002>, Banerjee and Kundu (2008) <doi:10.1109/TR.2008.916890>, Iyer, Jammalamadaka, and Kundu (2008) <doi:10.1016/j.jspi.2007.03.062>, Wu and Kus (2009) <doi:10.1016/j.csda.2009.03.010>, Prajapati, Mitra, and Kundu (2019) <doi:10.1007/s13571-018-0167-0>, Mondal and Kundu (2020) <doi:10.1080/03610926.2018.1554128>, Balakrishnan and Aggarwala (2000, ISBN:980-1-4612-1334-5), Ding and Gui (2023) <doi:10.3390/math11092003>, Nagar, Kumar, and Krishna (2026) <doi:10.59467/IJASS.2026.22.1>, Goel and Krishna (2026) <doi:10.1007/s13198-026-03208-w>, Yadav, Jaiswal, and Yadav (2026) <doi:10.1007/s11135-026-02647-8>, and Goel, Kumar, and Krishna (2026, ``Estimation in power Lindley distributions using balanced joint progressively Type-II censored data").

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 2026-08-06 10:10:02 UTC

Contents

is_control 2
is_diagnostics 3
is_dist 4
is_fit 4
is_loglik 6
is_model_comparison 8
is_posterior_expectation 8
is_predictive 9
is_proposal_defensive 9
is_proposal_normal 10
is_proposal_t 10
is_proposal_uniform 11
simulate_complete 11
simulate_hybrid 12
simulate_interval 12
simulate_left 13
simulate_progressive 13
simulate_right 14

Index **15**

`is_control` *Control settings for importance sampling*

Description

Control settings for importance sampling

Usage

```
is_control(
  n_draws = 10000L,
  method = c("self_normalized", "ordinary", "adaptive"),
  adapt_fraction = 0.25,
  min_ess = 50,
  truncate_weights = Inf,
  seed = NULL,
  optimize = TRUE,
  optim_control = list()
)
```

Arguments

n_draws	number of parameter draws.
method	one of "self_normalized", "ordinary", or "adaptive".
adapt_fraction	pilot fraction for adaptive deterministic-mixture IS.
min_ess	minimum ESS required before a warning is issued.
truncate_weights	optional positive cap on normalized weights, as a multiple of 1 / n_draws; use Inf to disable truncation.
seed	optional integer seed.
optimize	whether to refine the maximum-likelihood draw using optim.
optim_control	control list passed to optim.

Value

A list of class is_control.

---

is_diagnostics *Importance-sampling diagnostics*

Description

Importance-sampling diagnostics

Usage

```
is_diagnostics(object)
```

Arguments

object	an isfit object.

Value

A list containing ESS, weight dispersion, entropy, and Monte Carlo standard errors for parameter means.

`is_dist` *Define a distribution specification*

Description

Define a distribution specification

Usage

```
is_dist(
  pdf,
  cdf = NULL,
  survival = NULL,
  support = c(-Inf, Inf),
  name = "custom"
)
```

Arguments

`pdf`	function of `x` and `theta` returning a density.
`cdf`	optional function of `x` and `theta` returning a CDF.
`survival`	optional survival function of `x` and `theta`.
`support`	numeric length-two support bounds for the observations.
`name`	descriptive label.

Value

An object of class `is_dist`.

`is_fit` *Fit a censored univariate model by importance sampling*

Description

`is_fit()` samples parameter values from a proposal distribution, evaluates the observed-data likelihood for each draw, and computes stable normalized importance weights. With `prior` (or `log_prior`) it returns a Bayesian posterior approximation; without one, it approximates the normalized likelihood under a flat reference measure. The reported `par` is an observed likelihood maximum refined from the best importance draw when requested.

Usage

```
is_fit(
  data,
  pdf,
  cdf = NULL,
  survival = NULL,
  theta0,
  proposal = NULL,
  proposal_density = NULL,
  prior = NULL,
  log_prior = NULL,
  support = c(-Inf, Inf),
  scheme = "complete",
  status = NULL,
  time2 = NULL,
  censor_params = list(),
  control = is_control()
)
```

Arguments

`data`	observed data, a numeric vector, data frame, or list.
`pdf`	density function `pdf(x, theta)`.
`cdf`	CDF function `cdf(x, theta)`.
`survival`	survival function `survival(x, theta)`.
`theta0`	initial parameter vector, used to validate dimension and make a default normal proposal when `proposal` is omitted.
`proposal`	proposal object, or a function `proposal(n)`.
`proposal_density`	density function for a function-valued proposal.
`prior`	prior density `prior(theta)`.
`log_prior`	log prior function `log_prior(theta)`.
`support`	observation support bounds retained in the fitted object.
`scheme`	censoring or truncation scheme.
`status`	event status indicator.
`time2`	upper interval endpoints.
`censor_params`	scheme-specific parameters.
`control`	settings from `is_control()`.

Value

An object of class `isfit`.

References

Geweke, J. (1989). Bayesian inference in econometric models using Monte Carlo integration. *Econometrica*, 57(6), 1317-1339. doi:10.2307/2290062

Hesterberg, T. (1995). Weighted average importance sampling and defensive mixture distributions. *Technometrics*, 37(2), 185-194. doi:10.1080/00031305.1995.10476138

Robert, C. P., & Casella, G. (2004). *Monte Carlo Statistical Methods* (2nd ed.). Springer. ISBN: 978-0-387-21617-1.

Examples

```
x <- rexp(40, 1.5)
proposal <- is_proposal_normal(log(1.5), 0.5)
fit <- is_fit(x,
  pdf = function(x, theta) dexp(x, rate = exp(theta[1])),
  cdf = function(x, theta) pexp(x, rate = exp(theta[1])),
  survival = function(x, theta) pexp(x, rate = exp(theta[1]), lower.tail = FALSE),
  theta0 = log(1), proposal = proposal, scheme = "complete",
  control = is_control(n_draws = 500)
)
exp(coef(fit))
```

---

is_loglik *Observed-data log likelihood*

Description

Computes a log likelihood for an arbitrary univariate distribution. The density, CDF, and survival functions all use the convention `fun(x, theta)`.

Usage

```
is_loglik(
  data,
  theta,
  pdf,
  cdf = NULL,
  survival = NULL,
  scheme = "complete",
  status = NULL,
  time2 = NULL,
  censor_params = list()
)
```

Arguments

`data`	observed data.
`theta`	numeric parameter vector.
`pdf`	density function.
`cdf`	CDF function; required for finite or left-censored intervals.
`survival`	survival function; an alternative to `cdf` for right tails.
`scheme`	censoring/truncation scheme.
`status`	event status.
`time2`	interval upper endpoints.
`censor_params`	additional scheme parameters.

Value

A scalar log likelihood.

References

Balakrishnan, N., & Aggarwala, R. (2000). *Progressive Censoring: Theory, Methods, and Applications*. Birkhauser. ISBN: 978-1-4612-1334-5.

Banerjee, A., & Kundu, D. (2008). Inference based on Type-II hybrid censored data from Weibull distribution. *IEEE Transactions on Reliability*, 57(2), 369-378. doi:10.1109/TR.2008.916890

Ding, C., & Gui, W. (2023). Statistical inference of power Lindley distribution under progressive first-failure censoring. *Mathematics*, 11(9), 2003. doi:10.3390/math11092003

Goel, C., & Krishna, H. (2026). Reliability estimation in inverse Weibull distribution under progressive Type-II censoring. *Journal of Reliability and Statistical Studies*. doi:10.1007/s13198026-03208w

Goel, C., Kumar, M., & Krishna, H. (2026). Estimation in power Lindley distributions using balanced joint progressively Type-II censored data. *Preprint*.

Iyer, S. K., Jammalamadaka, S. R., & Kundu, D. (2008). Analysis of middle-censored data with exponential lifetime distribution. *Journal of Statistical Planning and Inference*, 138(11), 3550-3560. doi:10.1016/j.jspi.2007.03.062

Kundu, D., & Joarder, A. (2006). Analysis of Type-II progressively hybrid censored data. *Computational Statistics & Data Analysis*, 50(10), 2509-2528. doi:10.1016/j.csda.2005.05.002

Mondal, S., & Kundu, D. (2020). Point and interval estimation of parameters of Weibull distribution under middle censoring scheme. *Communications in Statistics - Theory and Methods*, 49(8), 1984-2003. doi:10.1080/03610926.2018.1554128

Nagar, S., Kumar, M., & Krishna, H. (2026). Bayesian estimation under censoring. *International Journal of Agricultural and Statistical Sciences*, 22(1). doi:10.59467/IJASS.2026.22.1

Prajapati, A., Mitra, S., & Kundu, D. (2019). On progressive first-failure censoring scheme. *Journal of Statistical Theory and Practice*, 13(1), 16. doi:10.1007/s1357101801670

Wu, S. J., & Kus, C. (2009). On estimation methods for the Weibull distribution under progressive first-failure censored data. *Computational Statistics & Data Analysis*, 53(10), 3617-3626. doi:10.1016/j.csda.2009.03.010

Yadav, A. S., Jaiswal, S., & Yadav, S. K. (2026). Statistical properties and inference for censoring schemes. *Quality & Quantity*. doi:10.1007/s11135026026478

`is_model_comparison` *Model comparison for multiple IS fits*

Description

Model comparison for multiple IS fits

Usage

```
is_model_comparison(...)
```

Arguments

`...`	one or more `isfit` objects.

Value

A data frame comparing logLik, AIC, AICc, BIC, CAIC, and HQIC.

`is_posterior_expectation`
Compute an expectation from an IS fit

Description

Compute an expectation from an IS fit

Usage

```
is_posterior_expectation(object, fun)
```

Arguments

`object`	an `isfit` object.
`fun`	function mapping one parameter vector to a numeric scalar or fixed-length numeric vector.

Value

Weighted posterior or normalized-likelihood expectation.

is_predictive *Posterior predictive functionals*

Description

Posterior predictive functionals

Usage

```
is_predictive(object, x, type = c("density", "cdf", "survival"))
```

Arguments

object	an isfit object.
x	values at which to evaluate the predictive quantity.
type	"density", "cdf", or "survival".

Value

A weighted predictive estimate at x.

is_proposal_defensive *Defensive mixture proposal*

Description

Constructs a deterministic mixture of two proposals. The defensive component protects against under-dispersed initial proposals and is evaluated exactly in the resulting mixture density.

Usage

```
is_proposal_defensive(primary, defensive, epsilon = 0.1)
```

Arguments

primary	proposal object.
defensive	proposal object of the same dimension.
epsilon	mixture weight assigned to the defensive proposal.

Value

A proposal object accepted by is_fit().

is_proposal_normal *Multivariate normal proposal for model parameters*

Description

This base-R proposal is convenient for unconstrained parameters. For constrained parameters, provide a proposal on a suitable transformed scale and incorporate the Jacobian in `proposal_density`.

Usage

```
is_proposal_normal(mean, covariance)
```

Arguments

`mean`	proposal mean.
`covariance`	positive-definite covariance matrix, or positive variances.

Value

A proposal object accepted by `is_fit()`.

is_proposal_t *Student t proposal for model parameters*

Description

Student t proposal for model parameters

Usage

```
is_proposal_t(mean, scale, df = 4)
```

Arguments

`mean`	proposal location.
`scale`	positive-definite scale matrix or variances.
`df`	degrees of freedom.

Value

A proposal object accepted by `is_fit()`.

is_proposal_uniform *Uniform proposal for model parameters*

Description

Uniform proposal for model parameters

Usage

```
is_proposal_uniform(lower, upper)
```

Arguments

lower	lower bounds for parameters.
upper	upper bounds for parameters.

Value

A proposal object accepted by `is_fit()`.

simulate_complete *Simulate complete univariate data*

Description

Simulate complete univariate data

Usage

```
simulate_complete(n, rfun, theta)
```

Arguments

n	sample size.
rfun	random-generation function `rfun(n, theta)`.
theta	parameter vector passed to `rfun`.

Value

A numeric sample.

simulate_hybrid *Simulate hybrid-censored data*

Description

Simulate hybrid-censored data

Usage

```
simulate_hybrid(n, rfun, theta, censor_time, m = NULL)
```

Arguments

n	initial sample size.
rfun	random-generation function rfun(n, theta).
theta	model parameters.
censor_time	fixed censoring time.
m	target failure count (1 <= m <= n).

Value

A data frame with x and event status.

simulate_interval *Simulate interval-censored data*

Description

Simulate interval-censored data

Usage

```
simulate_interval(n, rfun, theta, inspection_times)
```

Arguments

n	sample size.
rfun	random-generation function rfun(n, theta).
theta	model parameters.
inspection_times	strictly increasing finite inspection times.

Value

A data frame with interval endpoints L and R.

simulate_left *Simulate left-censored data*

Description

Simulate left-censored data

Usage

```
simulate_left(n, rfun, theta, censoring)
```

Arguments

n	sample size.
rfun	random-generation function rfun(n, theta).
theta	model parameters.
censoring	either one censoring time or a function censoring(n).

Value

A data frame with x and event status.

simulate_progressive *Simulate progressively Type-II-censored data*

Description

Simulate progressively Type-II-censored data

Usage

```
simulate_progressive(rfun, theta, R)
```

Arguments

rfun	random-generation function rfun(n, theta).
theta	model parameters.
R	non-negative removals immediately after successive observed failures.

Value

A list suitable for scheme = "progressive_type2".

`simulate_right` *Simulate right-censored data*

Description

Simulate right-censored data

Usage

```
simulate_right(n, rfun, theta, censoring)
```

Arguments

`n`	sample size.
`rfun`	random-generation function `rfun(n, theta)`.
`theta`	model parameters.
`censoring`	either one censoring time or a function `censoring(n)`.

Value

A data frame with `x` and event `status`.

Index

is_control, 2
is_control(), *5*
is_diagnostics, 3
is_dist, 4
is_fit, 4
is_fit(), *9–11*
is_loglik, 6
is_model_comparison, 8
is_posterior_expectation, 8
is_predictive, 9
is_proposal_defensive, 9
is_proposal_normal, 10
is_proposal_t, 10
is_proposal_uniform, 11

simulate_complete, 11
simulate_hybrid, 12
simulate_interval, 12
simulate_left, 13
simulate_progressive, 13
simulate_right, 14