Package ‘ECMLE’

May 7, 2026

Type Package

Title Approximating Evidence via Bounded Harmonic Means

Version 0.1.0

Description Implements the Elliptical Covering Marginal Likelihood Estimator (ECMLE), a geometric method for approximating marginal likelihood from posterior draws and log-posterior evaluations. The method constructs a collection of non-overlapping ellipsoids in a high-posterior-density region, computes the covered volume, and combines this with posterior sample coverage to estimate model evidence. It is designed to stabilize harmonic-mean-based evidence approximation and can be applied in multimodal settings. The methodology is described in Naderi et al. (2025) <doi:10.48550/arXiv.2510.20617>.

License GPL-3

Encoding UTF-8

Depends R (>= 3.6.0)

Imports IDPmisc, graphics, stats, withr

Suggests testthat (>= 3.0.0), knitr, rmarkdown

VignetteBuilder knitr

Config/testthat/edition 3

URL `https://github.com/da-na-deri/ECMLE`

BugReports `https://github.com/da-na-deri/ECMLE/issues`

NeedsCompilation no

Author Dana Naderi [aut, cre],
Christian P. Robert [aut],
Kaniav Kamary [aut],
Darren Wraith [aut]

Maintainer Dana Naderi <`naderi@ceremade.dauphine.fr`>

Repository CRAN

Date/Publication 2026-03-26 10:10:12 UTC

Contents

ECMLE-package 2
ecmle 3
ecmle-methods 5
pair_plot 6
plot_ecmle_2d 7
rosenbrock 8

Index **11**

`ECMLE-package` *ECMLE: Elliptical Covering Marginal Likelihood Estimator*

Description

Implements Elliptical Covering Marginal Likelihood Estimator (ECMLE), a geometric method for approximating marginal likelihood from posterior draws and log-posterior evaluations. The approach constructs a collection of non-overlapping ellipsoids within a high-posterior-density region, computes the covered volume, and combines this with posterior sample coverage to estimate model evidence. The method is designed to stabilize harmonic-mean-based evidence approximation and can be applied in multimodal settings. The methodology is described in Naderi et al. (2025) <doi:10.48550/arXiv.2510.20617>.

Details

The package provides the main estimator `ecmle()` with print, summary, and plot methods, visualisation functions `plot_ecmle_2d()` and `draw_ellipse_2d()`, a pair plot utility `pair_plot()`, and Rosenbrock (banana) posterior benchmark functions for testing and reproducible experiments.

Author(s)

Dana Naderi [aut, cre], Christian P Robert [aut], Kaniav Kamary [aut], Darren Wraith [aut]

See Also

`ecmle`

`ecmle` *Estimate the log marginal likelihood with ECMLE*

Description

Computes an ECMLE estimate of the log marginal likelihood from posterior draws, corresponding log unnormalized posterior evaluations, and a function that evaluates the log unnormalized posterior density at arbitrary points.

Usage

```
ecmle(
  post_samples,
  lps,
  log_post_fn,
  hpd_level = 0.75,
  subsample_frac = 0.05,
  r_max = NULL,
  bisect_tol = 1e-04,
  seed = NULL,
  verbose = FALSE
)
```

Arguments

`post_samples`	Numeric matrix of posterior draws; each row is one draw.
`lps`	Numeric vector of log unnormalized posterior values (log prior + log likelihood, up to an additive constant) evaluated at the rows of `post_samples`. Must be consistent with `log_post_fn`; see the note below.
`log_post_fn`	A function that accepts a numeric vector of length d and returns a single finite numeric scalar equal to the log unnormalized posterior density at that point. Called repeatedly during ellipsoid construction, so it should be reasonably fast. Must be consistent with `lps`; see the note below.
`hpd_level`	Fraction in `(0, 1)` used to define the HPD region. Default `0.75`.
`subsample_frac`	Fraction of HPD points retained for candidate ellipsoid centers. Default `0.05`.
`r_max`	Upper bracketing radius used by the directional bisection solver. If `NULL`, estimated from the HPD points.
`bisect_tol`	Tolerance for the directional radius solver. Default `1e-4`.
`seed`	Optional integer seed used for HPD subsampling.
`verbose`	Logical; if `TRUE`, emits basic progress information.

Details

The algorithm splits posterior draws into two halves. The first half is used to define and pack ellipsoids inside a high-posterior-density region. The second half is used to estimate the coverage-adjusted harmonic-mean quantity that yields the final estimate.

Value

An object of class `"ecmle"`, a list with components:

`log_marginal_likelihood`	Final log marginal likelihood estimate.
`log_marginal_likelihood_iter`	Running estimate over the second half-sample.
`ellipsoids`	List of fitted ellipsoids.
`total_volume`	Total ellipsoid volume.
`n_ellipsoids`	Number of ellipsoids.
`points_in_ellipsoids`	Number of evaluation points inside the ellipsoids.
`n_samples`	Number of evaluation points in the second half-sample.
`coverage_rate`	Fraction of evaluation points covered by the ellipsoids.
`hpd_level`	HPD level used in the fit.

Note

Consistency of `lps` and `log_post_fn` is critical. Both must evaluate the same log unnormalized posterior (log prior + log likelihood) up to the same additive constant. If they differ by more than a constant the marginal likelihood estimate will be incorrect.

Examples

```
set.seed(1)
d <- 2
b <- rep(-1, d - 1)
a <- rep(0,  d - 1)

Y_bar <- rosenbrock_generate_data(
  theta_true = seq(1, 2.8, length.out = d),
  n = 20, b = b, a = a, sigma = 8
)
samps <- rosenbrock_exact_posterior(Y_bar, n = 20,
  b = b, a = a, sigma = 8, n_samples = 500L)
lps <- rosenbrock_log_post_vec(samps, Y_bar,
  n = 20, b = b, a = a, sigma = 8)
log_post_fn <- function(theta)
  rosenbrock_log_post(theta, Y_bar, n = 20, b = b, a = a, sigma = 8)

fit <- ecmle(samps, lps, log_post_fn, hpd_level = 0.75, seed = 1L)
fit
summary(fit)
plot_ecmle_2d(fit, post_samples = samps)
```

`ecmle-methods` *Methods for ECMLE objects*

Description

S3 methods for inspecting ECMLE fits.

Usage

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

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

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

## S3 method for class 'ecmle'
plot(
  x,
  y,
  xlab = "Iteration",
  ylab = "Running log marginal likelihood",
  main = "ECMLE running estimate",
  ...
)
```

Arguments

`x`	An object of class `"ecmle"` or `"summary.ecmle"`.
`object`	An object of class `"ecmle"`.
`y`	Unused.
`xlab`	X-axis label for the plot.
`ylab`	Y-axis label for the plot.
`main`	Plot title.
`...`	Unused or additional graphical parameters.

Value

`summary.ecmle()` returns an object of class `"summary.ecmle"`. The print and plot methods return their input invisibly.

`pair_plot` *Pair plot of posterior samples*

Description

Produces a lower-triangular scatter/density pair plot of a posterior sample matrix. Diagonal panels display scaled marginal histograms. Lower off-diagonal panels display a 2-D pixel-density image using a blue colour ramp. Upper panels are left blank.

Usage

```
pair_plot(samples, pixs = 1, labels = NULL)
```

Arguments

`samples`	Numeric matrix with one row per posterior draw and one column per parameter. Column names are used as axis labels when `labels` is `NULL`.
`pixs`	Positive scalar controlling the pixel size of the 2-D density image. Smaller values produce finer resolution at higher computational cost. Default `1`.
`labels`	Optional character vector of length `ncol(samples)`. When `NULL` (default), column names of `samples` are used; if there are no column names, labels default to `theta1`, `theta2`, etc.

Value

Invisibly returns `NULL`. Called for its side-effect (drawing a plot).

Examples

```
set.seed(1)
d <- 2
b <- rep(-1, d - 1)
a <- rep(0, d - 1)
Y_bar <- rosenbrock_generate_data(seq(1, 2.8, length.out = d),
                                  n = 20, b = b, a = a, sigma = 8)
samps <- rosenbrock_exact_posterior(Y_bar, n = 20, b = b, a = a,
                                    sigma = 8, n_samples = 2000L)
pair_plot(samps, pixs = 0.5)
```

`plot_ecmle_2d` *2-D visualisation of ECMLE ellipsoids and posterior samples*

Description

`plot_ecmle_2d` visualises the output of `ecmle()` for a two-dimensional posterior. The second-half evaluation samples are colour-coded by ellipsoid membership (navy = inside, grey = outside) and the fitted ellipsoid boundaries are overlaid in red. For $d > 2$ the function emits an informative message and returns `fit` invisibly without plotting.

`draw_ellipse_2d` is the low-level helper that traces a single ellipse boundary on the current graphics device. It can also be called directly to add ellipses to an existing plot.

Usage

```
plot_ecmle_2d(
  fit,
  post_samples,
  col_inside = "navy",
  col_outside = "grey",
  col_ellipse = "red",
  pch = 19,
  cex = 0.3,
  lwd_ellipse = 1.5,
  npoints = 100L,
  xlab = expression(theta[1]),
  ylab = expression(theta[2]),
  main = NULL,
  legend_pos = "topright",
  ...
)

draw_ellipse_2d(center, Sigma, npoints = 100L, col = "red", lwd = 1.5, ...)
```

Arguments

`fit`	An object of class `"ecmle"` returned by `ecmle()`.
`post_samples`	Numeric matrix of posterior draws (N by d); the same matrix passed to `ecmle()`. Only the second half-sample is used for the scatter.
`col_inside`	Colour for samples inside at least one ellipsoid. Default `"navy"`.
`col_outside`	Colour for samples outside all ellipsoids. Default `"grey"`.
`col_ellipse`	Colour for ellipsoid boundaries. Default `"red"`.
`pch`	Point character; default `19` (solid circle).
`cex`	Point size; default `0.3`.
`lwd_ellipse`	Line width for ellipse boundaries; default `1.5`.

`npoints`	Integer; number of boundary points per ellipse. Default `100`.
`xlab`	X-axis label; default `expression(theta[1])`.
`ylab`	Y-axis label; default `expression(theta[2])`.
`main`	Plot title. When `NULL` (default) a title of the form "ECMLE: k ellipsoids" is constructed automatically.
`legend_pos`	Legend position string passed to `legend()`, or `NULL` to suppress the legend. Default `"topright"`.
`center`	Numeric vector of length 2; ellipse center.
`Sigma`	2 by 2 symmetric positive-definite covariance matrix defining the ellipse boundary.
`col`	Line colour for `draw_ellipse_2d`; default `"red"`.
`lwd`	Line width for `draw_ellipse_2d`; default `1.5`.
`...`	Additional graphical parameters passed to `plot()` or `lines()`.

Value

`plot_ecmle_2d` returns `fit` invisibly. `draw_ellipse_2d` returns `NULL` invisibly. Both are called for their side-effect (drawing on the active graphics device).

Examples

```
set.seed(42)
post_samps <- cbind(rnorm(400), rnorm(400))
lps <- apply(post_samps, 1, function(z) sum(dnorm(z, log = TRUE)))
log_post_fn <- function(theta) sum(dnorm(theta, log = TRUE))

fit <- ecmle(post_samps, lps, log_post_fn, hpd_level = 0.75, seed = 1L)
plot_ecmle_2d(fit, post_samples = post_samps)

# draw_ellipse_2d standalone usage
plot(0, 0, xlim = c(-3, 3), ylim = c(-3, 3), type = "n", asp = 1,
     xlab = "", ylab = "")
draw_ellipse_2d(center = c(0, 0),
                Sigma  = matrix(c(1, 0.4, 0.4, 1), 2, 2))
```

`rosenbrock` *Rosenbrock (banana) posterior*

Description

Functions for the Rosenbrock (banana-shaped) posterior, providing data generation, exact posterior sampling, and log-posterior evaluation.

`rosenbrock_generate_data` draws one sufficient statistic vector. `rosenbrock_exact_posterior` draws exact posterior samples. `rosenbrock_log_post` evaluates the log-posterior at a single parameter vector (suitable as the `log_post_fn` argument to `ecmle()`). `rosenbrock_log_post_vec` is the vectorised version over a sample matrix.

Usage

```
rosenbrock_generate_data(theta_true, n, b, a, sigma)

rosenbrock_exact_posterior(Y_bar, n, b, a, sigma, n_samples = 10000L)

rosenbrock_log_post(theta, Y_bar, n, b, a, sigma)

rosenbrock_log_post_vec(theta_matrix, Y_bar, n, b, a, sigma)
```

Arguments

`theta_true`	Numeric vector of length d; true parameter values.
`n`	Positive integer; number of observations used to form the sufficient statistic.
`b`	Numeric vector of length $d - 1$; curvature (banana) parameters.
`a`	Numeric vector of length $d - 1$; shift parameters.
`sigma`	Positive scalar; observation standard deviation.
`Y_bar`	Numeric vector of length d; sufficient statistic, typically the output of `rosenbrock_generate_data()`.
`n_samples`	Positive integer; number of posterior draws to generate. Default `10000`.
`theta`	Numeric vector of length d; a single parameter value.
`theta_matrix`	Numeric matrix with d columns; one parameter vector per row.

Value

`rosenbrock_generate_data`	Numeric vector of length d.
`rosenbrock_exact_posterior`	Numeric matrix of dimensions `n_samples` by d, with column names `theta1`, ..., `thetad`.
`rosenbrock_log_post`	A single numeric value.
`rosenbrock_log_post_vec`	Numeric vector of length `nrow(theta_matrix)`.

Examples

```
set.seed(1)
d <- 2
b <- rep(-1, d - 1)
a <- rep(0,  d - 1)

# 1. Generate one dataset
Y_bar <- rosenbrock_generate_data(
  theta_true = seq(1, 2.8, length.out = d),
  n = 20, b = b, a = a, sigma = 8
)

# 2. Draw exact posterior samples
samps <- rosenbrock_exact_posterior(Y_bar, n = 20,
  b = b, a = a, sigma = 8, n_samples = 500L)
```

```
# 3. Evaluate the log-posterior
lps <- rosenbrock_log_post_vec(samps, Y_bar,
  n = 20, b = b, a = a, sigma = 8)

# 4. Build a log_post_fn closure for ecmle()
log_post_fn <- function(theta)
  rosenbrock_log_post(theta, Y_bar, n = 20, b = b, a = a, sigma = 8)

fit <- ecmle(samps, lps, log_post_fn, hpd_level = 0.75, seed = 1L)
fit
```

Index

∗ **package**
    `ECMLE-package`, 2

`draw_ellipse_2d` *(`plot_ecmle_2d`)*, 7

`ecmle`, *2*, 3, *7, 8*
`ecmle-methods`, 5
`ECMLE-package`, 2

`legend`, *8*
`lines`, *8*

`pair_plot`, 6
`plot`, *8*
`plot.ecmle` *(`ecmle-methods`)*, 5
`plot_ecmle_2d`, 7
`print.ecmle` *(`ecmle-methods`)*, 5
`print.summary.ecmle` *(`ecmle-methods`)*, 5

`rosenbrock`, 8
`rosenbrock_exact_posterior` *(`rosenbrock`)*, 8
`rosenbrock_generate_data` *(`rosenbrock`)*, 8
`rosenbrock_log_post` *(`rosenbrock`)*, 8
`rosenbrock_log_post_vec` *(`rosenbrock`)*, 8

`summary.ecmle` *(`ecmle-methods`)*, 5