

Package ‘DEmixR’

September 25, 2026

Type Package

Title Fit Two-Component Normal and Lognormal Mixture Models

Version 0.3.0

Description Fits, bootstraps, and evaluates two-component normal and lognormal mixture models. Parameters are searched within data-derived bounds by differential-evolution global optimization, as implemented in the 'DEoptim' package (Mullen, Ardia, Gil, Windover and Cline, 2011) [doi:10.18637/jss.v040.i06](https://doi.org/10.18637/jss.v040.i06), followed by a local 'L-BFGS-B' refinement step via optim(). Also provides preliminary diagnostic plots, diagnostic plots for fitted mixtures, automatic normal-versus-lognormal model selection by the Bayesian or Akaike information criterion, and parametric or nonparametric bootstrap confidence intervals for the fitted parameters.

Depends R (>= 4.0.0)

Imports DEoptim (>= 2.0.0), pbapply (>= 1.0.0), graphics, stats, utils

License MIT + file LICENSE

URL <https://github.com/FHabibzadeh/DEmixR>

BugReports <https://github.com/FHabibzadeh/DEmixR/issues>

Encoding UTF-8

Suggests knitr, rmarkdown, testthat (>= 3.1.7), parallelly

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Config/roxygen2/version 8.1.0

Author Farrokh Habibzadeh [aut, cre] (ORCID:
<https://orcid.org/0000-0001-5360-2900>)

Maintainer Farrokh Habibzadeh <farrokh.habibzadeh@gmail.com>

Repository CRAN

Date/Publication 2026-09-25 06:10:02 UTC

Contents

bootstrap_mix2	2
evaluate_init	4
fit_lognorm2	5
fit_norm2	6
mix2_bounds	7
plot.demixr_fit	8
prelim_plots	9
print.demixr_boot	10
print.demixr_fit	11
print.demixr_select	12
print.summary.demixr_fit	12
select_best_mixture	13
summary.demixr_fit	14
Index	15

bootstrap_mix2	<i>Bootstrap mixture parameters</i>
----------------	-------------------------------------

Description

Parametric or nonparametric bootstrap of the five parameters of a fitted two-component mixture, with percentile confidence intervals.

Usage

```
bootstrap_mix2(  
  fit = NULL,  
  x = NULL,  
  par = NULL,  
  family = NULL,  
  B = 1000,  
  parametric = TRUE,  
  boot_size = NULL,  
  parallelType = 0,  
  quiet = 2,  
  ci_level = 0.95,  
  refit = c("local", "de"),  
  center = c("median", "mean"),  
  de_control = list(),  
  scale_ratio = NULL  
)
```

Arguments

fit	fitted object from fit_lognorm2 or fit_norm2
x	numeric vector (if fit not provided)
par	numeric vector of parameters (if fit not provided)
family	"lognormal" or "normal" (if fit not provided)
B	number of bootstrap replicates
parametric	logical, parametric bootstrap if TRUE
boot_size	size or fraction (if between 0 and 1) of bootstrap sample
parallelType	values 0 or 1 run serially. An integer greater than 1 is passed to pblapply as the number of worker processes (forking, so Unix-alikes only: on Windows an integer value is ignored by pbapply and the bootstrap runs serially). It is <i>not</i> the DEoptim parallelType used by the fitting functions.
quiet	0/1/2 for verbosity (0 suppresses the progress bar)
ci_level	confidence level
refit	"local" (default) or "de"; see Details
center	summary reported in central: "median" (default) or "mean" of the bootstrap distribution
de_control	list of DE settings used when refit = "de"; any of NP (default 50), itermax (2000), n_runs (1), reltol, steptol, F, CR, strategy
scale_ratio	relative scale constraint, taken from fit unless given; see fit_norm2

Details

Two refitting strategies are available. With `refit = "local"` (the default, fast) every replicate is refitted by a local L-BFGS-B optimization started at the original estimate. This does *not* repeat the global search, so it does not propagate uncertainty arising from multimodality of the likelihood: a replicate whose likelihood has a different dominant mode may be pulled back to the original solution. With `refit = "de"` every replicate is refitted with the same procedure as the original fit (a differential-evolution search followed by L-BFGS-B; the DE settings are controlled by `de_control`); this is much slower. Percentile intervals of mixture parameters can behave poorly with weakly separated components, small mixing proportions or solutions on the bounds; their coverage should not be taken for granted.

Value

an object of class `demixr_boot`: a list with the converged bootstrap estimates (`ests_clean`), the chosen summary (`central`), the mean and median, percentile confidence intervals (`ci`), the number of converged (`n_success`) and failed (`n_failed`) replicates and the `refit` strategy. Has a `print` method.

Reproducibility

No seed argument is provided. Call `set.seed()` before the call for reproducible resampling (only guaranteed for `parallelType` 0 or 1, i.e. serial execution; forked workers do not generally reproduce from a single seed).

Examples

```
set.seed(1)
x <- c(rnorm(150, 0, 1), rnorm(50, 4, 1))
fit <- fit_norm2(x, n_runs = 1, NP = 20, itermax = 200, quiet = 0)
boot <- bootstrap_mix2(fit, B = 20, quiet = 0)
boot
```

evaluate_init	<i>Evaluate initial parameter values for mixture fitting</i>
---------------	--

Description

Runs a single L-BFGS-B optimization from the supplied starting values inside the same bounds used by the fitting functions.

Usage

```
evaluate_init(
  par_init,
  x,
  family = c("lognormal", "normal"),
  lower = NULL,
  upper = NULL,
  pgtol = 1e-08,
  scale_ratio = 0
)
```

Arguments

par_init	numeric vector of initial parameters (p, m1, s1, m2, s2)
x	numeric vector of data
family	"lognormal" or "normal"
lower	numeric vector of lower bounds
upper	numeric vector of upper bounds
pgtol	numeric, gradient tolerance for optim
scale_ratio	relative scale constraint, see fit_norm2 (default 0 = none)

Value

list with success flag (TRUE when L-BFGS-B reports convergence code 0), the optimized parameters (ordered so that $m1 \leq m2$), log-likelihood, and convergence code and message

Examples

```
set.seed(1)
x <- c(rnorm(150, 0, 1), rnorm(50, 4, 1))
ev <- evaluate_init(par_init = c(0.7, 0, 1, 4, 1), x = x, family = "normal")
ev$success
```

fit_lognorm2

*Fit 2-component lognormal mixture***Description**

Finds the best two-component lognormal mixture inside the bounded parameter region returned by [mix2_bounds](#), by differential evolution followed by an L-BFGS-B refinement. The result is the best solution found within those bounds, *not* a global maximum of the unrestricted likelihood (which is unbounded). The parameters m_1 , s_1 , m_2 , s_2 are the meanlog and sdlog of each component; components are ordered so that $m_1 \leq m_2$.

Usage

```
fit_lognorm2(x, ...)
```

Arguments

<code>x</code>	numeric vector of data to fit (positive, finite, at least 10 observations)
<code>...</code>	additional arguments passed to .fit_mix2_core (for example <code>n_runs</code> , <code>NP</code> , <code>lower</code> , <code>upper</code> , <code>scale_ratio</code>)

Value

an object of class `demixr_fit`: a list with the fitted parameters (`par`), `logLik`, `AIC`, `BIC`, convergence information (`convergence`, `source`, `local_convergence`), the bounds used, the log-likelihood reached by each DE run (`de_logLik`) and diagnostics. Has `print`, `summary`, and `plot` methods.

See Also

[mix2_bounds](#), [bootstrap_mix2](#)

Examples

```
set.seed(1)
x <- c(rlnorm(150, 0, 0.5), rlnorm(50, 2, 0.3))
fit <- fit_lognorm2(x, n_runs = 1, NP = 20, itermax = 200, quiet = 0)
fit

plot(fit)
```

fit_norm2	<i>Fit 2-component normal mixture</i>
-----------	---------------------------------------

Description

Finds the best two-component normal mixture inside the bounded parameter region returned by [mix2_bounds](#), by differential evolution followed by an L-BFGS-B refinement. The result is the best solution found within those bounds, *not* a global maximum of the unrestricted likelihood (which is unbounded for component-specific variances). Components are ordered so that $m1 \leq m2$.

Usage

```
fit_norm2(x, ...)
```

Arguments

x	numeric vector of data to fit (finite, at least 10 observations)
...	additional arguments passed to .fit_mix2_core (for example n_runs, NP, lower, upper, scale_ratio)

Value

an object of class `demixr_fit`: a list with the fitted parameters (`par`), `logLik`, `AIC`, `BIC`, convergence information (`convergence`, `source`, `local_convergence`), the bounds used, the log-likelihood reached by each DE run (`de_logLik`) and diagnostics. Has `print`, `summary`, and `plot` methods.

See Also

[mix2_bounds](#), [bootstrap_mix2](#)

Examples

```
set.seed(1)
x <- c(rnorm(150, 0, 1), rnorm(50, 4, 1))
fit <- fit_norm2(x, n_runs = 1, NP = 20, itermax = 200, quiet = 0)
fit

plot(fit)
```

mix2_bounds

Parameter bounds used by the DEMixR fitting functions

Description

DEMixR does not maximize the likelihood over the whole natural parameter space. The likelihood of a heteroscedastic normal (or lognormal) mixture is unbounded – a component can collapse onto a single observation – so the differential-evolution search and the local refinement are carried out inside a box whose limits are derived from the data. `mix2_bounds()` returns that box so that it can be inspected, modified and passed back to `fit_norm2` / `fit_lognorm2` through lower and upper.

Usage

```
mix2_bounds(
  x,
  family = c("normal", "lognormal"),
  scale_range = c(0.1, 10),
  p_range = c(0.001, 0.999)
)
```

Arguments

<code>x</code>	numeric vector of data (strictly positive for family = "lognormal").
<code>family</code>	"normal" or "lognormal".
<code>scale_range</code>	numeric length 2; lower and upper limits of the component standard deviations, expressed as multiples of the sample standard deviation on the model's working scale (the original scale for the normal family and the log scale for the lognormal family).
<code>p_range</code>	numeric length 2; lower and upper limit of the mixing proportion p .

Details

For a normal mixture, with s_x the sample standard deviation of x , the default limits are

- mixing proportion p : `p_range` (default 0.001 to 0.999);
- component means m_1, m_2 : $[\min(x) - 2s_x, \max(x)]$ and $[\min(x), \max(x) + 2s_x]$, respectively. The two means are *not* restricted to opposite sides of the median; component labels are assigned after estimation by ordering the means;
- component standard deviations: s_x times `scale_range` (default 0.1 to 10).

For a lognormal mixture the same construction is applied on the log scale ($y = \log x$, $s_y = \text{sd}(y)$), with the mean limits padded by 2 log units instead of $2s_x$. The parameters m_j and s_j of the lognormal model are the meanlog and sdlog of component j .

Because of these limits the fitted values are the best solution found *within the box*, not a global maximum of the unrestricted likelihood. The factors 0.1 and 10 are pragmatic regularization choices;

a solution that lies on a bound (reported by the fitting functions) indicates that the bounds, not the data, determined that estimate.

A relative constraint on the ratio of the component scales (Hathaway, 1985) is available separately through the `scale_ratio` argument of the fitting functions.

Value

A list with numeric vectors lower and upper (named `p`, `m1`, `s1`, `m2`, `s2`).

References

Hathaway, R. J. (1985). A constrained formulation of maximum-likelihood estimation for normal mixture distributions. *The Annals of Statistics*, 13(2), 795-800.

Examples

```
set.seed(1)
x <- c(rnorm(150, 0, 1), rnorm(50, 4, 1))
mix2_bounds(x, "normal")
# tighter scale bounds
mix2_bounds(x, "normal", scale_range = c(0.2, 5))
```

plot.demixr_fit

Plot a fitted DEmixR mixture model

Description

Three views of a fitted mixture: the fitted density over a histogram (which = "density", default), the histogram of the probability integral transform ("pit"; approximately uniform when the fitted mixture describes the data; the fit is evaluated on the same data it was estimated from), and a Q-Q plot of the data against the quantiles of the fitted mixture ("qq").

Usage

```
## S3 method for class 'demixr_fit'
plot(
  x,
  hist_bins = 60,
  col_hist = "gray85",
  col_total = "black",
  col_comp1 = "steelblue",
  col_comp2 = "firebrick",
  which = c("density", "pit", "qq"),
  ...
)
```

Arguments

<code>x</code>	an object of class <code>demixr_fit</code>
<code>hist_bins</code>	number of histogram bins
<code>col_hist</code>	histogram fill color
<code>col_total</code>	color of the fitted overall mixture density
<code>col_comp1</code>	color of the first fitted component density
<code>col_comp2</code>	color of the second fitted component density
<code>which</code>	"density" (default), "pit" or "qq"
<code>...</code>	further arguments passed to <code>hist</code> ("density" and "pit" only)

Value

no return value, called for side effects (generating a plot)

Examples

```
set.seed(1)
x <- c(rnorm(150, 0, 1), rnorm(50, 4, 1))
fit <- fit_norm2(x, n_runs = 1, NP = 20, itemax = 200, quiet = 0)
plot(fit)
plot(fit, which = "pit")
plot(fit, which = "qq")
```

```
prelim_plots
```

```
Preliminary diagnostic plots
```

Description

Exploratory displays of the data before fitting. The Q-Q and P-P panels compare the data with a *single* normal (or lognormal, for the "log" panels) distribution. A two-component mixture is not itself normal, so curvature in these plots is expected and they do not validate or refute the assumed component family. To assess a fitted mixture use `plot(fit, which = "pit")` or `plot(fit, which = "qq")`.

Usage

```
prelim_plots(
  x,
  which = c("hist"),
  hist_bins = 60,
  col_hist = "gray85",
  col_density = "darkorange",
  col_qq = "gray60",
  col_line = "darkorange"
)
```

Arguments

x	numeric vector
which	character vector: "hist", "qq", "pp", "logqq", "logpp"
hist_bins	number of bins for histogram
col_hist	color for histogram
col_density	color for the kernel-density curve drawn over the histogram. Set to NA to omit the density curve entirely; in that case the histogram title is "Histogram" instead of "Histogram with density".
col_qq	color for qq points
col_line	color for lines in "qq", "pp", "logqq" plots

Value

no return value, called for side effects (generating plots)

Examples

```
set.seed(1)
x <- c(rnorm(150, 0, 1), rnorm(50, 4, 1))
prelim_plots(x, c("hist", "qq"), hist_bins = 15)

# Histogram without the kernel-density overlay (title becomes "Histogram")
prelim_plots(x, "hist", col_density = NA)
```

print.demixr_boot	<i>Print bootstrap results from a DEmixR mixture fit</i>
-------------------	--

Description

Print bootstrap results from a DEmixR mixture fit

Usage

```
## S3 method for class 'demixr_boot'
print(x, digits = 4, ...)
```

Arguments

x	an object of class demixr_boot, as returned by bootstrap_mix2
digits	number of significant digits to print
...	further arguments passed to or from other methods (currently unused)

Value

x, invisibly

Examples

```
set.seed(1)
x <- c(rnorm(150, 0, 1), rnorm(50, 4, 1))
fit <- fit_norm2(x, n_runs = 1, NP = 20, itermax = 200, quiet = 0)
boot <- bootstrap_mix2(fit, B = 20, quiet = 0)
print(boot)
```

print.demixr_fit	<i>Print a fitted DEmixR mixture model</i>
------------------	--

Description

Print a fitted DEmixR mixture model

Usage

```
## S3 method for class 'demixr_fit'
print(x, digits = 4, ...)
```

Arguments

x	an object of class demixr_fit, as returned by fit_norm2 or fit_lognorm2
digits	number of significant digits to print
...	further arguments passed to or from other methods (currently unused)

Value

x, invisibly

Examples

```
set.seed(1)
x <- c(rnorm(150, 0, 1), rnorm(50, 4, 1))
fit <- fit_norm2(x, n_runs = 1, NP = 20, itermax = 200, quiet = 0)
print(fit)
```

```
print.demixr_select
```

Print model-selection results from DEmixR

Description

Print model-selection results from DEmixR

Usage

```
## S3 method for class 'demixr_select'  
print(x, digits = 4, ...)
```

Arguments

x	an object of class demixr_select, as returned by select_best_mixture
digits	number of significant digits to print
...	further arguments passed to or from other methods (currently unused)

Value

x, invisibly

Examples

```
set.seed(1)  
x <- c(rnorm(150, 0, 1), rnorm(50, 4, 1))  
mix <- select_best_mixture(x, n_runs = 1, NP = 20, itermax = 200, quiet = 0)  
print(mix)
```

```
print.summary.demixr_fit
```

Print a summary of a fitted DEmixR mixture model

Description

Print a summary of a fitted DEmixR mixture model

Usage

```
## S3 method for class 'summary.demixr_fit'  
print(x, digits = 4, ...)
```

Arguments

<code>x</code>	an object of class <code>summary.demixr_fit</code> , as returned by <code>summary.demixr_fit</code>
<code>digits</code>	number of significant digits to print
<code>...</code>	further arguments passed to or from other methods (currently unused)

Value

`x`, invisibly

Examples

```
set.seed(1)
x <- c(rnorm(150, 0, 1), rnorm(50, 4, 1))
fit <- fit_norm2(x, n_runs = 1, NP = 20, itermax = 200, quiet = 0)
print(summary(fit))
```

<code>select_best_mixture</code>	<i>Select best mixture model (lognormal or normal) based on AIC or BIC</i>
----------------------------------	--

Description

Fits a two-component normal mixture and, if all observations are positive, a two-component lognormal mixture, and returns the family with the lower information criterion. Both densities are defined on the original scale of x , so their likelihoods and information criteria are comparable. When x contains non-positive values the lognormal family is not possible and the choice is dictated by the support of the data, not by the criterion.

Usage

```
select_best_mixture(
  x,
  n_runs = 10,
  NP = 100,
  itermax = 10000,
  quiet = 2,
  criterion = c("BIC", "AIC"),
  ...
)
```

Arguments

<code>x</code>	numeric vector
<code>n_runs</code>	number of DEoptim runs (default 10, as in <code>fit_norm2</code>)
<code>NP</code>	population size for DEoptim (default 100, as in <code>fit_norm2</code>)
<code>itermax</code>	maximum iterations
<code>quiet</code>	verbosity
<code>criterion</code>	"BIC" (default) or "AIC"
<code>...</code>	further arguments passed to <code>fit_norm2</code> and <code>fit_lognorm2</code> , e.g. <code>scale_ratio</code>

Value

an object of class `demixr_select`: a list with the best-fitting model (`best`), all fitted models (`all`), the BIC of each (BICs), the AIC of each (AICs), the criterion used and `best_family`. Has a print method.

Examples

```
set.seed(1)
x <- c(rnorm(150, 0, 1), rnorm(50, 4, 1))
mix <- select_best_mixture(x, n_runs = 1, NP = 20, itermax = 200, quiet = 0)
mix
mix$best$family
```

<code>summary.demixr_fit</code>	<i>Summarize a fitted DEmixR mixture model</i>
---------------------------------	--

Description

Summarize a fitted DEmixR mixture model

Usage

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

Arguments

<code>object</code>	an object of class <code>demixr_fit</code>
<code>...</code>	further arguments passed to or from other methods (currently unused)

Value

an object of class `summary.demixr_fit`: a list holding the per-component weight/location/scale table together with the family, sample size, log-likelihood, AIC, BIC, and convergence code. Has a print method.

Examples

```
set.seed(1)
x <- c(rnorm(150, 0, 1), rnorm(50, 4, 1))
fit <- fit_norm2(x, n_runs = 1, NP = 20, itermax = 200, quiet = 0)
summary(fit)
```

Index

`.fit_mix2_core`, [5](#), [6](#)
`bootstrap_mix2`, [2](#), [5](#), [6](#), [10](#)
`evaluate_init`, [4](#)
`fit_lognorm2`, [5](#), [7](#), [11](#), [13](#)
`fit_norm2`, [3](#), [4](#), [6](#), [7](#), [11](#), [13](#)
`hist`, [9](#)
`mix2_bounds`, [5](#), [6](#), [7](#)
`pblapply`, [3](#)
`plot.demixr_fit`, [8](#)
`prelim_plots`, [9](#)
`print.demixr_boot`, [10](#)
`print.demixr_fit`, [11](#)
`print.demixr_select`, [12](#)
`print.summary.demixr_fit`, [12](#)
`select_best_mixture`, [12](#), [13](#)
`set.seed`, [3](#)
`summary.demixr_fit`, [13](#), [14](#)