Package 'ebnm'

May 8, 2026

Encoding UTF-8

Type Package

Version 1.1-42

Date 2025-10-08

Title Solve the Empirical Bayes Normal Means Problem

URL https://github.com/stephenslab/ebnm

BugReports https://github.com/stephenslab/ebnm/issues

Description Provides simple, fast, and stable functions to fit the normal means model using empirical Bayes. For available models and details, see function ebnm(). Our JSS article, Willwerscheid, Carbonetto, and Stephens (2025) <doi:10.18637/jss.v114.i03>, provides a detailed introduction to the package.

License GPL (>= 3)

NeedsCompilation no

Depends R (>= 3.3.0)

Imports stats, ashr, mixsqp, truncnorm, trust, deconvolveR, magrittr, rlang, dplyr, ggplot2

Suggests testthat, REBayes, horseshoe, knitr, rmarkdown, cowplot, mcmc, numDeriv

RoxygenNote 7.3.2

VignetteBuilder knitr

LazyData true

Author Jason Willwerscheid [aut], Matthew Stephens [aut], Peter Carbonetto [aut, cre], Andrew Goldstein [ctb], Yusha Liu [ctb]

Maintainer Peter Carbonetto <peter.carbonetto@gmail.com>

Repository CRAN

Date/Publication 2025-10-10 05:30:25 UTC

Contents

coef.ebnm *Extract posterior means from a fitted EBNM model*

Description

The coef method for class ebnm.

Usage

```
## S3 method for class 'ebnm'
coef(object, ...)
```

Arguments

object	The fitted ebnm object.
...	Not used. Included for consistency as an S3 method.

confint.ebnm *Obtain credible intervals using a fitted EBNM model*

Description

The confint method for class ebnm. Estimates posterior "credible intervals" for each "true mean" θ_i. We define the $(1-\alpha)\%$ credible interval for θ_i as the narrowest continuous interval $[a_i, b_i]$ such that $\theta_i \in [a_i, b_i]$ with posterior probability at least $1-\alpha$, where $\alpha \in (0, 1)$. We estimate these credible intervals using Monte Carlo sampling. Note that by default, ebnm does not return a posterior sampler; one can be added to the ebnm object using function ebnm_add_sampler.

Usage

```
## S3 method for class 'ebnm'
confint(object, parm, level = 0.95, nsim = 1000, ...)
```

Arguments

object	The fitted ebnm object.
parm	A vector of numeric indices specifying which means θ_i are to be given confidence intervals. If missing, all observations are considered.
level	The "confidence level" $1-\alpha$ desired.
nsim	The number of samples to use to estimate confidence intervals.
...	Additional arguments to be passed to the posterior sampler function. Since ebnm_horseshoe returns an MCMC sampler, it takes parameter burn, the number of burn-in samples to discard. At present, no other samplers take any additional parameters.

Value

A matrix with columns giving lower and upper confidence limits for each mean θ_i. These will be labelled as "CI.lower" and "CI.upper".

---

ebnm *Solve the EBNM problem*

Description

Solves the empirical Bayes normal means (EBNM) problem using a specified family of priors. For a detailed introduction to the package, see Willwerscheid, Carbonetto, and Stephens (2025), our JSS article cited in **References** below.

Usage

```
ebnm(
  x,
  s = 1,
 prior_family = c("point_normal", "point_laplace", "point_exponential", "normal",
    "horseshoe", "normal_scale_mixture", "unimodal", "unimodal_symmetric",
   "unimodal_nonnegative", "unimodal_nonpositive", "generalized_binary", "npmle",
    "deconvolver", "flat", "point_mass", "ash"),
  mode = 0,
  scale = "estimate",
  g_init = NULL,
  fix_g = FALSE,
  output = ebnm_output_default(),
  optmethod = NULL,
  control = NULL,
  ...
)

ebnm_output_default()

ebnm_output_all()
```

Arguments

x	A vector of observations. Missing observations (NAs) are not allowed.
s	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed. Two prior families have additional restrictions: when horseshoe priors are used, errors must be homoskedastic; and since function deconv in package deconvolveR takes z-scores, the "deconvolver" family requires that all standard errors be equal to 1.
prior_family	A character string that specifies the prior family G. See **Details** below.

mode — A scalar specifying the mode of the prior g or "estimate" if the mode is to be estimated from the data. This parameter is ignored by the NPMLE, the deconvolveR family, and the improper uniform (or "flat") prior. For generalized binary priors, which are bimodal, the mode parameter specifies the mode of the truncated normal component (the location of the point mass is fixed at zero).

scale — A scalar or vector specifying the scale parameter(s) of the prior or "estimate" if the scale parameters are to be estimated from the data. This parameter is ignored by the flat prior and the family of point mass priors.

The interpretation of scale depends on the prior family. For normal and point-normal families, it is a scalar specifying the standard deviation of the normal component. For point-Laplace and point-exponential families, it is a scalar specifying the scale parameter of the Laplace or exponential component. For the horseshoe family, it corresponds to $s\tau$ in the usual parametrization of the horseshoe distribution. For the family of generalized binary priors, it specifies the ratio of the (untruncated) standard deviation of the normal component to its mode. This ratio must be fixed in advance (i.e., argument "estimate" is unavailable for generalized binary priors). For the NPMLE and deconvolveR prior family, scale is a scalar specifying the distance between successive means in the grid of point masses or normal distributions used to estimate g. For all other prior families, which are implemented using the function ash in package ashr, it is a vector specifying the parameter mixsd to be passed to ash or "estimate" if mixsd is to be chosen by ebnm. (Note that ebnm chooses mixsd differently from ash: see functions ebnm_scale_normalmix, ebnm_scale_unimix, and ebnm_scale_npmle for details. To use the ash grid, set scale = "estimate" and pass in gridmult as an additional parameter. See ash for defaults and details.)

g_init — The prior distribution g. Usually this is left unspecified (NULL) and estimated from the data. However, it can be used in conjuction with fix_g = TRUE to fix the prior (useful, for example, to do computations with the "true" g in simulations). If g_init is specified but fix_g = FALSE, g_init specifies the initial value of g used during optimization. For non-parametric priors, this has the side effect of fixing the mode and scale parameters. If g_init is supplied, it should be an object of class normalmix for normal, point-normal, scale mixture of normals, and deconvolveR prior families, as well as for the NPMLE; class laplacemix for point-Laplace families; class gammamix for point-exponential families; class horseshoe for horseshoe families; class unimix for unimodal_ families; or class tnormalmix for generalized binary priors. An object of class ebnm can also be supplied as argument, provided that field fitted_g contains a prior of the correct class (see **Examples** below).

fix_g — If TRUE, fix the prior g at g_init instead of estimating it.

output — A character vector indicating which values are to be returned. Function ebnm_output_default() provides the default return values, while ebnm_output_all() lists all possible return values. See **Value** below.

optmethod — A string specifying which optimization function is to be used.

For parametric families other than the horseshoe and generalized binary (normal, point-normal, point-Laplace, and point-exponential), options include "nlm" (which

calls nlm), "lbfgsb" (which calls optim with method = "L-BFGS-B"), and "trust" (which calls into the trust package). Other options are "nohess_nlm", "nograd_nlm", and "nograd_lbfgsb", which use numerical approximations rather than exact expressions for the Hessian; both of the "nograd" functions use numerical approximations for the gradient as well. The default option is "nohess_nlm".

Since the horseshoe, generalized binary, and point mass families only require one parameter to be estimated (at most), the only available optimization method is optimize, and thus the optmethod parameter is ignored by ebnm_horseshoe, ebnm_generalized_binary, and ebnm_point_mass.

For most nonparametric families (scale mixtures of normals; unimodal, symmetric unimodal, nonnegative unimodal, and nonpositive unimodal families; and the NPMLE), optmethod options are provided by package ashr. The default method uses the mix-SQP algorithm implemented in the mixsqp package. See the ash function documentation for other options. For the NPMLE only, it is also possible to specify optmethod = "REBayes", which uses function GLmix in the REBayes package to estimate the NPMLE rather than using the ashr package. Note that REBayes requires installation of the commercial interior-point solver MOSEK; for details, see the documentation for REBayes function KWDual.

The nonparametric exception is the the "deconvolveR" family. Since the deconvolveR package only ever uses nlm, ebnm_deconvolver ignores the optmethod parameter.

control — A list of control parameters to be passed to the optimization function specified by parameter optmethod.

... — Additional parameters. When a unimodal_ prior family is used, these parameters are passed to function ash in package ashr. Although it does not call into ashr, the scale mixture of normals family accepts parameter gridmult for purposes of comparison. When gridmult is set, an ashr-style grid will be used instead of the default ebnm grid. When the "deconvolver" family is used, additional parameters are passed to function deconv in package deconvolveR. Families of generalized binary priors take several additional parameters; see ebnm_generalized_binary. In all other cases, additional parameters are ignored.

Details

Given vectors of data x and standard errors s, ebnm solves the "empirical Bayes normal means" (EBNM) problem for various choices of prior family. The model is

$$x_j|\theta_j, s_j \sim N(\theta_j, s_j^2)$$

$$\theta_j \sim g \in G,$$

where g, which is referred to as the "prior distribution" for θ, is to be estimated from among some specified family of prior distributions G. Several options for G are implemented, some parametric and others non-parametric; see below for examples.

Solving the EBNM problem involves two steps. First, $g \in G$ is estimated via maximum marginal likelihood:

$$\hat{g} := \arg\max_{g \in G} L(g),$$

where

$$L(g) := \prod_j \int p(x_j|\theta_j, s_j) g(d\theta_j).$$

Second, posterior distributions $p(\theta_j|x_j, s_j, \hat{g})$ and/or summaries such as posterior means and posterior second moments are computed.

Implemented prior families include:

`point_normal` The family of mixtures where one component is a point mass at μ and the other is a normal distribution centered at μ.

`point_laplace` The family of mixtures where one component is a point mass at μ and the other is a double-exponential distribution centered at μ.

`point_exponential` The family of mixtures where one component is a point mass at μ and the other is a (nonnegative) exponential distribution with mode μ.

`normal` The family of normal distributions.

`horseshoe` The family of horseshoe distributions.

`normal_scale_mixture` The family of scale mixtures of normals.

`unimodal` The family of all unimodal distributions.

`unimodal_symmetric` The family of symmetric unimodal distributions.

`unimodal_nonnegative` The family of unimodal distributions with support constrained to be greater than the mode.

`unimodal_nonpositive` The family of unimodal distributions with support constrained to be less than the mode.

`generalized_binary` The family of mixtures where one component is a point mass at zero and the other is a truncated normal distribution with lower bound zero and nonzero mode. See Liu et al. (2023), cited in **References** below.

`npmle` The family of all distributions.

`deconvolver` A non-parametric exponential family with a natural spline basis. Like `npmle`, there is no unimodal assumption, but whereas `npmle` produces spiky estimates for g, `deconvolver` estimates are much more regular. See `deconvolveR-package` for details and references.

`flat` The "non-informative" improper uniform prior, which yields posteriors

$$\theta_j|x_j, s_j \sim N(x_j, s_j^2).$$

`point_mass` The family of point masses δ_μ. Posteriors are point masses at μ.

`ash` Calls into function `ash` in package `ashr`. Can be used to make direct comparisons of `ebnm` and `ashr` implementations of prior families such as scale mixtures of normals and the NPMLE.

Value

An `ebnm` object. Depending on the argument to `output`, the object is a list containing elements:

`data` A data frame containing the observations `x` and standard errors `s`.

`posterior` A data frame of summary results (posterior means, standard deviations, second moments, and local false sign rates).

`fitted_g` The fitted prior $\hat{g}$ (an object of class `normalmix`, `laplacemix`, `gammamix`, `unimix`, `tnormalmix`, or `horseshoe`).

`log_likelihood` The optimal log likelihood attained, $L(\hat{g})$.

`posterior_sampler` A function that can be used to produce samples from the posterior. For all prior families other than the horseshoe, the sampler takes a single parameter `nsamp`, the number of posterior samples to return per observation. Since `ebnm_horseshoe` returns an MCMC sampler, it additionally takes parameter `burn`, the number of burn-in samples to discard.

S3 methods `coef`, `confint`, `fitted`, `logLik`, `nobs`, `plot`, `predict`, `print`, `quantile`, `residuals`, `simulate`, `summary`, and `vcov` have been implemented for `ebnm` objects. For details, see the respective help pages, linked below under **See Also**.

Functions

- `ebnm_output_default()`: Lists the default return values.
- `ebnm_output_all()`: Lists all valid return values.

References

Jason Willwerscheid, Peter Carbonetto, and Matthew Stephens (2025). `ebnm`: an `R` Package for solving the empirical Bayes normal means problem using a variety of prior families. *Journal of Statistical Software*, **114**(3), 1–33. doi:10.18637/jss.v114.i03.

Yusha Liu, Peter Carbonetto, Jason Willwerscheid, Scott A Oakes, Kay F Macleod, and Matthew Stephens (2025). Dissecting tumor transcriptional heterogeneity from single-cell RNA-seq data by generalized binary covariance decomposition. *Nature Genetics* **57**, 263–273. doi:10.1038/s41588-02401997z.

See Also

A plotting method is available for `ebnm` objects: see `plot.ebnm`.

For other methods, see `coef.ebnm`, `confint.ebnm`, `fitted.ebnm`, `logLik.ebnm`, `nobs.ebnm`, `predict.ebnm`, `print.ebnm`, `print.summary.ebnm`, `quantile.ebnm`, `residuals.ebnm`, `simulate.ebnm`, `summary.ebnm`, and `vcov.ebnm`.

Calling into functions `ebnm_point_normal`, `ebnm_point_laplace`, `ebnm_point_exponential`, `ebnm_normal`, `ebnm_horseshoe`, `ebnm_normal_scale_mixture`, `ebnm_unimodal`, `ebnm_unimodal_symmetric`, `ebnm_unimodal_nonnegative`, `ebnm_unimodal_nonpositive`, `ebnm_generalized_binary`, `ebnm_npmle`, `ebnm_deconvolver`, `ebnm_flat`, `ebnm_point_mass`, and `ebnm_ash` is equivalent to calling into `ebnm` with `prior_family` set accordingly.

Examples

```
theta <- c(rep(0, 100), rexp(100))
s <- 1
x <- theta + rnorm(200, 0, s)

# The following are equivalent:
pn.res <- ebnm(x, s, prior_family = "point_normal")
pn.res <- ebnm_point_normal(x, s)
```

```
# Inspect results:
logLik(pn.res)
plot(pn.res)

# Fix the scale parameter:
pl.res <- ebnm_point_laplace(x, s, scale = 1)

# Estimate the mode:
normal.res <- ebnm_normal(x, s, mode = "estimate")
plot(normal.res) # posterior means shrink to a value different from zero

# Use an initial g (this fixes mode and scale for ash priors):
normalmix.res <- ebnm_normal_scale_mixture(x, s, g_init = pn.res)

# Fix g and get different output (including a posterior sampler):
pn.res <- ebnm_point_normal(x, s, g_init = pn.res, fix_g = TRUE,
                            output = ebnm_output_all())

# Sample from the posterior:
pn.samp <- simulate(pn.res, nsim = 100)

# Quantiles and HPD confidence intervals can be obtained via sampling:
set.seed(1)
pn.quantiles <- quantile(pn.res, probs = c(0.1, 0.9))
pn.quantiles[1:5, ]
confint(pn.res, level = 0.8, parm = 1:5)

# Examples of usage of control parameter:
#  point_normal uses nlm:
pn.res <- ebnm_point_normal(x, s, control = list(print.level = 1))
#  unimodal uses mixsqp:
unimodal.res <- ebnm_unimodal(x, s, control = list(verbose = TRUE))
```

`ebnm_add_sampler` *Add sampler to an ebnm_object*

Description

Adds a posterior sampler to a fitted `ebnm` object.

Usage

```
ebnm_add_sampler(ebnm_res)
```

Arguments

`ebnm_res`	The fitted ebnm object.

Value

The ebnm object with an additional field posterior_sampler.

---

ebnm_ash *Solve the EBNM problem using an ash family of distributions*

Description

A wrapper to function ash in package ashr. Identical to function ebnm with argument prior_family = "ash".

Usage

```
ebnm_ash(
  x,
  s = 1,
  mode = 0,
  scale = "estimate",
  g_init = NULL,
  fix_g = FALSE,
  output = ebnm_output_default(),
  control = NULL,
  ...
)
```

Arguments

x	A vector of observations. Missing observations (NAs) are not allowed.
s	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed.
mode	Passed to ash as parameter mode.
scale	Passed to ash as parameter mixsd.
g_init	Passed to ash as parameter g.
fix_g	Passed to ash as parameter fixg.
output	A character vector indicating which values are to be returned. Function ebnm_output_default() provides the default return values, while ebnm_output_all() lists all possible return values. See **Value** below.
control	Passed to ash as parameter control.
...	Additional parameters to be passed to ash.

Value

An ebnm object. Depending on the argument to output, the object is a list containing elements:

data A data frame containing the observations x and standard errors s.

posterior A data frame of summary results (posterior means, standard deviations, second moments, and local false sign rates).

fitted_g The fitted prior $\hat{g}$.

log_likelihood The optimal log likelihood attained, $L(\hat{g})$.

posterior_sampler A function that can be used to produce samples from the posterior. The sampler takes a single parameter nsamp, the number of posterior samples to return per observation.

S3 methods coef, confint, fitted, logLik, nobs, plot, predict, print, quantile, residuals, simulate, summary, and vcov have been implemented for ebnm objects. For details, see the respective help pages, linked below under **See Also**.

See Also

See ebnm for examples of usage and model details.

Available S3 methods include coef.ebnm, confint.ebnm, fitted.ebnm, logLik.ebnm, nobs.ebnm, plot.ebnm, predict.ebnm, print.ebnm, print.summary.ebnm, quantile.ebnm, residuals.ebnm, simulate.ebnm, summary.ebnm, and vcov.ebnm.

---

ebnm_check_fn *Check a custom ebnm function*

Description

Checks inputs and outputs of an ebnm-style function. Designed to troubleshoot custom functions, especially those that are intended for use in the flashier package (e.g., as argument to the ebnm_fn parameter in function flash).

Usage

```
ebnm_check_fn(fn, x, s)
```

Arguments

fn	The function to be checked.
x	A test set (vector) of observations.
s	A test set of standard errors. Typically, ebnm-style functions should be able to accept a vector of standard errors or a scalar if all standard errors are identical. This is not always the case; for example, the horseshoe prior family requires homoskedastic standard errors.

Value

Prints a success message and silently returns 1 if all checks pass. Otherwise the function errors out.

Examples

```
ebnm_check_fn(ebnm_normal, x = rnorm(10, sd = 2), s = 1)
```

`ebnm_deconvolver` *Solve the EBNM problem using the "deconvolveR" family of distributions*

Description

Solves the empirical Bayes normal means (EBNM) problem using a non-parametric exponential family with a natural spline basis. Like `ebnm_npmle`, there is no unimodal assumption, but whereas `ebnm_npmle` produces spiky estimates for g, `ebnm_deconvolver` estimates are much more regular. See `deconvolveR-package` for details and references. Identical to function `ebnm` with argument `prior_family = "deconvolver"`.

Usage

```
ebnm_deconvolver(
  x,
  s = 1,
  scale = "estimate",
  g_init = NULL,
  fix_g = FALSE,
  output = ebnm_output_default(),
  control = NULL,
  ...
)
```

Arguments

`x`	A vector of observations. Missing observations (`NA`s) are not allowed.
`s`	Standard errors, which must be uniformly equal to 1 (i.e., `s = 1`) since the deconvolveR method takes z-scores as input.
`scale`	A deconvolveR prior is a finite mixture of point masses $$\pi_1 \delta_{\mu_1} + \ldots + \pi_K \delta_{\mu_K},$$ where parameters π_k are estimated and the point masses are evenly spaced over (μ_1, μ_K).The distance between successive point masses can be specified by the user via parameter `scale`, in which case the argument should be a scalar specifying the distance $d = \mu_2 - \mu_1 = \cdots = \mu_K - \mu_{K-1}$; alternatively, if `scale = "estimate"`, then ebnm sets the grid via function `ebnm_scale_npmle`.

g_init	The prior distribution g. Usually this is left unspecified (NULL) and estimated from the data. However, it can be used in conjuction with fix_g = TRUE to fix the prior (useful, for example, to do computations with the "true" g in simulations). If g_init is specified but fix_g = FALSE, g_init specifies the initial value of g used during optimization. This has the side effect of fixing the scale parameter. When supplied, g_init should be an object of class normalmix or an ebnm object in which the fitted prior is an object of class normalmix.
fix_g	If TRUE, fix the prior g at g_init instead of estimating it.
output	A character vector indicating which values are to be returned. Function ebnm_output_default() provides the default return values, while ebnm_output_all() lists all possible return values. See **Value** below.
control	A list of control parameters to be passed to optimization function nlm.
...	Additional parameters to be passed to function deconv in package deconvolveR.

Value

An ebnm object. Depending on the argument to output, the object is a list containing elements:

data A data frame containing the observations x and standard errors s.

posterior A data frame of summary results (posterior means, standard deviations, second moments, and local false sign rates).

fitted_g The fitted prior $\hat{g}$.

log_likelihood The optimal log likelihood attained, $L(\hat{g})$.

posterior_sampler A function that can be used to produce samples from the posterior. The sampler takes a single parameter nsamp, the number of posterior samples to return per observation.

S3 methods coef, confint, fitted, logLik, nobs, plot, predict, print, quantile, residuals, simulate, summary, and vcov have been implemented for ebnm objects. For details, see the respective help pages, linked below under **See Also**.

See Also

See ebnm for examples of usage and model details.

Available S3 methods include coef.ebnm, confint.ebnm, fitted.ebnm, logLik.ebnm, nobs.ebnm, plot.ebnm, predict.ebnm, print.ebnm, print.summary.ebnm, quantile.ebnm, residuals.ebnm, simulate.ebnm, summary.ebnm, and vcov.ebnm.

---

ebnm_flat *Solve the EBNM problem using a flat prior*

Description

Solves the empirical Bayes normal means (EBNM) problem using a "non-informative" improper uniform prior, which yields posteriors

$$\theta_j | x_j, s_j \sim N(x_j, s_j^2).$$

Identical to function ebnm with argument prior_family = "flat". For details about the model, see ebnm.

Usage

```
ebnm_flat(
  x,
  s = 1,
  g_init = NULL,
  fix_g = FALSE,
  output = ebnm_output_default()
)
```

Arguments

x	A vector of observations. Missing observations (NAs) are not allowed.
s	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed.
g_init	Not used by ebnm_flat, but included for consistency with other ebnm functions.
fix_g	Not used by ebnm_flat, but included for consistency with other ebnm functions.
output	A character vector indicating which values are to be returned. Function ebnm_output_default() provides the default return values, while ebnm_output_all() lists all possible return values. See **Value** below.

Value

An ebnm object. Depending on the argument to output, the object is a list containing elements:

data A data frame containing the observations x and standard errors s.

posterior A data frame of summary results (posterior means, standard deviations, second moments, and local false sign rates).

fitted_g The fitted prior $\hat{g}$.

log_likelihood The optimal log likelihood attained, $L(\hat{g})$.

posterior_sampler A function that can be used to produce samples from the posterior. The sampler takes a single parameter nsamp, the number of posterior samples to return per observation.

S3 methods coef, confint, fitted, logLik, nobs, plot, predict, print, quantile, residuals, simulate, summary, and vcov have been implemented for ebnm objects. For details, see the respective help pages, linked below under **See Also**.

See Also

See ebnm for examples of usage and model details.

Available S3 methods include coef.ebnm, confint.ebnm, fitted.ebnm, logLik.ebnm, nobs.ebnm, plot.ebnm, predict.ebnm, print.ebnm, print.summary.ebnm, quantile.ebnm, residuals.ebnm, simulate.ebnm, summary.ebnm, and vcov.ebnm.

ebnm_generalized_binary *Solve the EBNM problem using generalized binary priors*

Description

Solves the empirical Bayes normal means (EBNM) problem using the family of nonnegative distributions consisting of mixtures where one component is a point mass at zero and the other is a truncated normal distribution with lower bound zero and nonzero mode. Typically, the mode is positive, with the ratio of the mode to the standard deviation taken to be large, so that posterior estimates are strongly shrunk towards one of two values (zero or the mode of the normal component). Identical to function ebnm with argument prior_family = "generalized_binary". For details, see Liu et al. (2023), cited in **References** below.

Usage

```
ebnm_generalized_binary(
  x,
  s = 1,
  mode = "estimate",
  scale = 0.1,
  g_init = NULL,
  fix_g = FALSE,
  output = ebnm_output_default(),
  control = NULL,
  ...
)
```

Arguments

x	A vector of observations. Missing observations (NAs) are not allowed.
s	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed.
mode	A scalar specifying the mode of the truncated normal component, or "estimate" if the mode is to be estimated from the data (the location of the point mass is fixed at zero).
scale	A scalar specifying the ratio of the (untruncated) standard deviation of the normal component to its mode. This ratio must be fixed in advance (i.e., it is not possible to set scale = "estimate" when using generalized binary priors).
g_init	The prior distribution g. Usually this is left unspecified (NULL) and estimated from the data. However, it can be used in conjuction with fix_g = TRUE to fix the prior (useful, for example, to do computations with the "true" g in simulations). If g_init is specified but fix_g = FALSE, g_init specifies the initial value of g used during optimization. When supplied, g_init should be an object of class tnormalmix or an ebnm object in which the fitted prior is an object of class tnormalmix.

`fix_g` If `TRUE`, fix the prior g at `g_init` instead of estimating it.

`output` A character vector indicating which values are to be returned. Function `ebnm_output_default()` provides the default return values, while `ebnm_output_all()` lists all possible return values. See **Value** below.

`control` A list of control parameters to be passed to function `optimize`.

`...` The following additional arguments act as control parameters for the outer EM loops in the fitting algorithm. Each loop iteratively updates parameters w (the mixture proportion corresponding to the truncated normal component) and μ (the mode of the truncated normal component):

`wlist` A vector defining intervals of w for which optimal solutions will separately be found. For example, if `wlist = c(0, 0.5, 1)`, then two optimal priors will be found: one such that w is constrained to be less than 0.5 and one such that it is constrained to be greater than 0.5.

`maxiter` A scalar specifying the maximum number of iterations to perform in each outer EM loop.

`tol` A scalar specifying the convergence tolerance parameter for each outer EM loop.

`mu_init` A scalar specifying the initial value of μ to be used in each outer EM loop.

`mu_range` A vector of length two specifying lower and upper bounds for possible values of μ.

Value

An `ebnm` object. Depending on the argument to `output`, the object is a list containing elements:

`data` A data frame containing the observations `x` and standard errors `s`.

`posterior` A data frame of summary results (posterior means, standard deviations, second moments, and local false sign rates).

`fitted_g` The fitted prior $\hat{g}$.

`log_likelihood` The optimal log likelihood attained, $L(\hat{g})$.

`posterior_sampler` A function that can be used to produce samples from the posterior. The sampler takes a single parameter `nsamp`, the number of posterior samples to return per observation.

S3 methods `coef`, `confint`, `fitted`, `logLik`, `nobs`, `plot`, `predict`, `print`, `quantile`, `residuals`, `simulate`, `summary`, and `vcov` have been implemented for `ebnm` objects. For details, see the respective help pages, linked below under **See Also**.

References

Yusha Liu, Peter Carbonetto, Jason Willwerscheid, Scott A Oakes, Kay F Macleod, and Matthew Stephens (2025). Dissecting tumor transcriptional heterogeneity from single-cell RNA-seq data by generalized binary covariance decomposition. *Nature Genetics* **57**, 263–273. doi:10.1038/s41588-02401997z.

See Also

See ebnm for examples of usage and model details.

Available S3 methods include coef.ebnm, confint.ebnm, fitted.ebnm, logLik.ebnm, nobs.ebnm, plot.ebnm, predict.ebnm, print.ebnm, print.summary.ebnm, quantile.ebnm, residuals.ebnm, simulate.ebnm, summary.ebnm, and vcov.ebnm.

---

ebnm_group *Solve the EBNM problem for grouped data*

Description

Solves the empirical Bayes normal means (EBNM) problem for observations belonging to distinct groups.

Usage

```
ebnm_group(
  x,
  s = 1,
  group,
  prior_family = "point_normal",
  mode = 0,
  scale = "estimate",
  g_init = NULL,
  fix_g = FALSE,
  output = ebnm_output_default(),
  ...
)
```

Arguments

x	A vector of observations. Missing observations (NAs) are not allowed.
s	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed. Two prior families have additional restrictions: when horseshoe priors are used, errors must be homoskedastic; and since function deconv in package deconvolveR takes z-scores, the "deconvolver" family requires that all standard errors be equal to 1.
group	A vector of character strings that gives the group to which each observation belongs. It must have the same length as argument x. For an example of usage, see Examples below.
prior_family	A named vector that specifies the prior family G for each group. If the same prior family is to be used for all groups, then a character string may be used instead.

mode — A named list that specifies, for each group, the mode of the respective prior g, or "estimate" if the mode is to be estimated from the data. If the mode is the same across groups, then a scalar may be used instead. If all modes are to be estimated, then mode = "estimate" may be used.

scale — A named list that specifies, for each group, the scale parameter(s) of the respective prior, or "estimate" if the scale parameters are to be estimated from the data. If the scale parameter is the same across groups, then a scalar may be used instead. If all scales are to be estimated, then scale = "estimate" may be used.

g_init — The prior distributions g. Usually this is left unspecified (NULL) and estimated from the data. However, it can be used in conjuction with fix_g = TRUE to fix the prior (useful, for example, to do computations with the "true" g in simulations). If g_init is specified but fix_g = FALSE, g_init specifies the initial value of g used during optimization. If g_init is supplied, it should be a named list that specifies, for each group, a prior of the appropriate class (normalmix for normal, point-normal, scale mixture of normals, and deconvolveR prior families, as well as for the NPMLE; class laplacemix for point-Laplace families; class gammamix for point-exponential families; class horseshoe for horseshoe families; and class unimix for unimodal_ families).

fix_g — If TRUE, fix the prior g at g_init instead of estimating it.

output — A character vector indicating which values are to be returned. Function ebnm_output_default() provides the default return values, while ebnm_output_all() lists all possible return values. See **Value** below.

... — Additional parameters. When a unimodal_ prior family is used, these parameters are passed to function ash in package ashr. Although it does not call into ashr, the scale mixture of normals family accepts parameter gridmult for purposes of comparison. When gridmult is set, an ashr-style grid will be used instead of the default ebnm grid. When the "deconvolver" family is used, additional parameters are passed to function deconv in package deconvolveR. Families of generalized binary priors take several additional parameters; see ebnm_generalized_binary. In all other cases, additional parameters are ignored.

Details

The EBNM model for grouped data, with observations x_j belonging to groups $k = 1, ..., K$, is

$$x_j|\theta_j, s_j \sim N(\theta_j, s_j^2)$$

$$\theta_j \sim g_{k(j)} \in G_{k(j)}.$$

Solving the EBNM problem for grouped data is equivalent to solving a separate EBNM problem for each group $k = 1, ..., K$, with the optimal log likelihood equal to the sum of the optimal log likelihoods for each separate problem.

Value

An ebnm object. Depending on the argument to output, the object is a list containing elements:

data A data frame containing the observations x and standard errors s.

posterior A data frame of summary results (posterior means, standard deviations, second moments, and local false sign rates).

fitted_g The fitted prior $\hat{g}$ (an object of class normalmix, laplacemix, gammamix, unimix, tnormalmix, or horseshoe).

log_likelihood The optimal log likelihood attained, $L(\hat{g})$.

posterior_sampler A function that can be used to produce samples from the posterior. For all prior families other than the horseshoe, the sampler takes a single parameter nsamp, the number of posterior samples to return per observation. Since ebnm_horseshoe returns an MCMC sampler, it additionally takes parameter burn, the number of burn-in samples to discard.

S3 methods coef, confint, fitted, logLik, nobs, plot, predict, print, quantile, residuals, simulate, summary, and vcov have been implemented for ebnm objects. For details, see the respective help pages, linked below under **See Also**.

See Also

ebnm

Examples

```
group <- c(rep("small_sd", 100), rep("large_sd", 100))
theta <- c(rnorm(100, sd = 1), rnorm(100, sd = 10))
s <- 1
x <- theta + rnorm(200, 0, s)

ebnm.group.res <- ebnm_group(x, s, group)

# Use different prior families for each group:
ebnm.group.res <- ebnm_group(
  x, s, group,
  prior_family = list(small_sd = "normal", large_sd = "normal_scale_mixture")
)

# Different modes and scales can be set similarly:
ebnm.group.res <- ebnm_group(
  x, s, group,
  mode = list(small_sd = 0, large_sd = "estimate"),
  scale = list(small_sd = 1, large_sd = "estimate")
)
```

---

ebnm_horseshoe *Solve the EBNM problem using horseshoe priors*

Description

Solves the empirical Bayes normal means (EBNM) problem using the family of horseshoe distributions. Identical to function ebnm with argument prior_family = "horseshoe". For details about the model, see ebnm.

Usage

```
ebnm_horseshoe(
  x,
  s = 1,
  scale = "estimate",
  g_init = NULL,
  fix_g = FALSE,
  output = ebnm_output_default(),
  control = NULL
)
```

Arguments

x	A vector of observations. Missing observations (NAs) are not allowed.
s	A *scalar* specifying the standard error of the observations (observations must be homoskedastic).
scale	A scalar corresponding to $s\tau$ in the usual parametrization of the horseshoe distribution, or "estimate" if this parameter is to be estimated from the data.
g_init	The prior distribution g. Usually this is left unspecified (NULL) and estimated from the data. However, it can be used in conjuction with fix_g = TRUE to fix the prior (useful, for example, to do computations with the "true" g in simulations). If g_init is specified but fix_g = FALSE, g_init specifies the initial value of g used during optimization. When supplied, g_init should be an object of class horseshoe or an ebnm object in which the fitted prior is an object of class horseshoe.
fix_g	If TRUE, fix the prior g at g_init instead of estimating it.
output	A character vector indicating which values are to be returned. Function ebnm_output_default() provides the default return values, while ebnm_output_all() lists all possible return values. See **Value** below.
control	A list of control parameters to be passed to function optimize.

Value

An ebnm object. Depending on the argument to output, the object is a list containing elements:

data A data frame containing the observations x and standard errors s.

posterior A data frame of summary results (posterior means, standard deviations, second moments, and local false sign rates).

fitted_g The fitted prior $\hat{g}$.

log_likelihood The optimal log likelihood attained, $L(\hat{g})$.

posterior_sampler A function that can be used to produce samples from the posterior. The function takes parameters nsamp, the number of posterior samples to return per observation, and burn, the number of burn-in samples to discard (an MCMC sampler is used).

S3 methods coef, confint, fitted, logLik, nobs, plot, predict, print, quantile, residuals, simulate, summary, and vcov have been implemented for ebnm objects. For details, see the respective help pages, linked below under **See Also**.

See Also

See ebnm for examples of usage and model details.

Available S3 methods include coef.ebnm, confint.ebnm, fitted.ebnm, logLik.ebnm, nobs.ebnm, plot.ebnm, predict.ebnm, print.ebnm, print.summary.ebnm, quantile.ebnm, residuals.ebnm, simulate.ebnm, summary.ebnm, and vcov.ebnm.

---

ebnm_normal *Solve the EBNM problem using normal priors*

Description

Solves the empirical Bayes normal means (EBNM) problem using the family of normal distributions. Identical to function ebnm with argument prior_family = "normal". For details about the model, see ebnm.

Usage

```
ebnm_normal(
  x,
  s = 1,
  mode = 0,
  scale = "estimate",
  g_init = NULL,
  fix_g = FALSE,
  output = ebnm_output_default(),
  optmethod = NULL,
  control = NULL
)
```

Arguments

x	A vector of observations. Missing observations (NAs) are not allowed.
s	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed.
mode	A scalar specifying the mode of the prior g or "estimate" if the mode is to be estimated from the data.
scale	A scalar specifying the standard deviation of the normal prior or "estimate" if the standard deviation is to be estimated from the data.
g_init	The prior distribution g. Usually this is left unspecified (NULL) and estimated from the data. However, it can be used in conjuction with fix_g = TRUE to fix the prior (useful, for example, to do computations with the "true" g in simulations). If g_init is specified but fix_g = FALSE, g_init specifies the initial value of g used during optimization. When supplied, g_init should be an object of class normalmix or an ebnm object in which the fitted prior is an object of class normalmix.

fix_g	If TRUE, fix the prior g at g_init instead of estimating it.
output	A character vector indicating which values are to be returned. Function ebnm_output_default() provides the default return values, while ebnm_output_all() lists all possible return values. See **Value** below.
optmethod	A string specifying which optimization function is to be used. Options include "nlm" (which calls nlm), "lbfgsb" (which calls optim with method = "L-BFGS-B"), and "trust" (which calls into the trust package). Other options are "nohess_nlm", "nograd_nlm", and "nograd_lbfgsb", which use numerical approximations rather than exact expressions for the Hessian; both of the "nograd" functions use numerical approximations for the gradient as well. The default option is "nohess_nlm".
control	A list of control parameters to be passed to the optimization function specified by parameter optmethod.

Value

An ebnm object. Depending on the argument to output, the object is a list containing elements:

data A data frame containing the observations x and standard errors s.

posterior A data frame of summary results (posterior means, standard deviations, second moments, and local false sign rates).

fitted_g The fitted prior $\hat{g}$.

log_likelihood The optimal log likelihood attained, $L(\hat{g})$.

posterior_sampler A function that can be used to produce samples from the posterior. The sampler takes a single parameter nsamp, the number of posterior samples to return per observation.

S3 methods coef, confint, fitted, logLik, nobs, plot, predict, print, quantile, residuals, simulate, summary, and vcov have been implemented for ebnm objects. For details, see the respective help pages, linked below under **See Also**.

See Also

See ebnm for examples of usage and model details.

Available S3 methods include coef.ebnm, confint.ebnm, fitted.ebnm, logLik.ebnm, nobs.ebnm, plot.ebnm, predict.ebnm, print.ebnm, print.summary.ebnm, quantile.ebnm, residuals.ebnm, simulate.ebnm, summary.ebnm, and vcov.ebnm.

---

ebnm_normal_scale_mixture

Solve the EBNM problem using scale mixtures of normals

Description

Solves the empirical Bayes normal means (EBNM) problem using the family of scale mixtures of normals. Identical to function ebnm with argument prior_family = "normal_scale_mixture". For details about the model, see ebnm.

Usage

```
ebnm_normal_scale_mixture(
  x,
  s = 1,
  mode = 0,
  scale = "estimate",
  g_init = NULL,
  fix_g = FALSE,
  output = ebnm_output_default(),
  optmethod = NULL,
  control = NULL,
  ...
)
```

Arguments

x	A vector of observations. Missing observations (NAs) are not allowed.
s	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed.
mode	A scalar specifying the mode of the prior g or "estimate" if the mode is to be estimated from the data.
scale	The nonparametric family of scale mixtures of normals is approximated via a finite mixture of normal distributions $$\pi_1 N(\mu, \sigma_1^2) + \ldots + \pi_K N(\mu, \sigma_K^2),$$ where parameters π_k are estimated and the grid of standard deviations $(\sigma_1, \ldots, \sigma_K)$ is fixed in advance. By making the grid sufficiently dense, one can obtain an arbitrarily good approximation. The grid can be specified by the user via parameter scale, in which case the argument should be the vector of standard deviations $(\sigma_1, \ldots, \sigma_K)$; alternatively, if scale = "estimate", then ebnm sets the grid via function ebnm_scale_normalmix. Note that ebnm sets the grid differently from function ash. To use the ash grid, set scale = "estimate" and pass in gridmult as an additional parameter. See ash for defaults and details.
g_init	The prior distribution g. Usually this is left unspecified (NULL) and estimated from the data. However, it can be used in conjuction with fix_g = TRUE to fix the prior (useful, for example, to do computations with the "true" g in simulations). If g_init is specified but fix_g = FALSE, g_init specifies the initial value of g used during optimization. This has the side effect of fixing the mode and scale parameters. When supplied, g_init should be an object of class normalmix or an ebnm object in which the fitted prior is an object of class normalmix.
fix_g	If TRUE, fix the prior g at g_init instead of estimating it.
output	A character vector indicating which values are to be returned. Function ebnm_output_default() provides the default return values, while ebnm_output_all() lists all possible return values. See **Value** below.

optmethod — A string specifying which optimization function is to be used. Options are provided by package ashr. The default method uses the mix-SQP algorithm implemented in the mixsqp package. See the ash function documentation for other options.

control — A list of control parameters to be passed to the optimization function specified by parameter optmethod.

... — When parameter gridmult is set, an ash-style grid will be used instead of the default ebnm grid (see parameter scale above). Other additional parameters are ignored.

Value

An ebnm object. Depending on the argument to output, the object is a list containing elements:

data A data frame containing the observations x and standard errors s.

posterior A data frame of summary results (posterior means, standard deviations, second moments, and local false sign rates).

fitted_g The fitted prior $\hat{g}$.

log_likelihood The optimal log likelihood attained, $L(\hat{g})$.

posterior_sampler A function that can be used to produce samples from the posterior. The sampler takes a single parameter nsamp, the number of posterior samples to return per observation.

S3 methods coef, confint, fitted, logLik, nobs, plot, predict, print, quantile, residuals, simulate, summary, and vcov have been implemented for ebnm objects. For details, see the respective help pages, linked below under **See Also**.

See Also

See ebnm for examples of usage and model details.

Available S3 methods include coef.ebnm, confint.ebnm, fitted.ebnm, logLik.ebnm, nobs.ebnm, plot.ebnm, predict.ebnm, print.ebnm, print.summary.ebnm, quantile.ebnm, residuals.ebnm, simulate.ebnm, summary.ebnm, and vcov.ebnm.

---

ebnm_npmle — *Solve the EBNM problem using the family of all distributions*

Description

Solves the empirical Bayes normal means (EBNM) problem using the family of all distributions. Identical to function ebnm with argument prior_family = "npmle". For details about the model, see ebnm.

Usage

```
ebnm_npmle(
  x,
  s = 1,
  scale = "estimate",
  g_init = NULL,
  fix_g = FALSE,
  output = ebnm_output_default(),
  optmethod = NULL,
  control = NULL
)
```

Arguments

x	A vector of observations. Missing observations (NAs) are not allowed.
s	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed.
scale	The nonparametric family of all distributions is approximated via a finite mixture of point masses $$\pi_1\delta_{\mu_1} + \ldots + \pi_K\delta_{\mu_K},$$ where parameters π_k are estimated and the point masses are evenly spaced over (μ_1, μ_K). By taking a sufficiently dense grid of point masses, one can obtain an arbitrarily good approximation. The distance between successive point masses can be specified by the user via parameter scale, in which case the argument should be a scalar specifying the distance $d = \mu_2 - \mu_1 = \cdots = \mu_K - \mu_{K-1}$; alternatively, if scale = "estimate", then ebnm sets the grid via function ebnm_scale_npmle.
g_init	The prior distribution g. Usually this is left unspecified (NULL) and estimated from the data. However, it can be used in conjuction with fix_g = TRUE to fix the prior (useful, for example, to do computations with the "true" g in simulations). If g_init is specified but fix_g = FALSE, g_init specifies the initial value of g used during optimization. This has the side effect of fixing the scale parameter. When supplied, g_init should be an object of class normalmix or an ebnm object in which the fitted prior is an object of class normalmix.
fix_g	If TRUE, fix the prior g at g_init instead of estimating it.
output	A character vector indicating which values are to be returned. Function ebnm_output_default() provides the default return values, while ebnm_output_all() lists all possible return values. See **Value** below.
optmethod	A string specifying which optimization function is to be used. Options are provided by package ashr. The default method uses the mix-SQP algorithm implemented in the mixsqp package. See the ash function documentation for other options. It is also possible to specify optmethod = "REBayes", which uses function GLmix in the REBayes package to estimate the NPMLE rather than ashr. Note that REBayes requires installation of the commercial interior-point solver MOSEK; for details, see KWDual (the core optimization routine for the REBayes package).

`control` A list of control parameters to be passed to the optimization function specified by parameter `optmethod`.

Value

An `ebnm` object. Depending on the argument to `output`, the object is a list containing elements:

`data` A data frame containing the observations `x` and standard errors `s`.

`posterior` A data frame of summary results (posterior means, standard deviations, second moments, and local false sign rates).

`fitted_g` The fitted prior $\hat{g}$.

`log_likelihood` The optimal log likelihood attained, $L(\hat{g})$.

`posterior_sampler` A function that can be used to produce samples from the posterior. The sampler takes a single parameter `nsamp`, the number of posterior samples to return per observation.

S3 methods `coef`, `confint`, `fitted`, `logLik`, `nobs`, `plot`, `predict`, `print`, `quantile`, `residuals`, `simulate`, `summary`, and `vcov` have been implemented for `ebnm` objects. For details, see the respective help pages, linked below under **See Also**.

See Also

See `ebnm` for examples of usage and model details.

Available S3 methods include `coef.ebnm`, `confint.ebnm`, `fitted.ebnm`, `logLik.ebnm`, `nobs.ebnm`, `plot.ebnm`, `predict.ebnm`, `print.ebnm`, `print.summary.ebnm`, `quantile.ebnm`, `residuals.ebnm`, `simulate.ebnm`, `summary.ebnm`, and `vcov.ebnm`.

---

ebnm_point_exponential

Solve the EBNM problem using point-exponential priors

Description

Solves the empirical Bayes normal means (EBNM) problem using the family of point-exponential priors (the family of mixtures where one component is a point mass at μ and the other is a (nonnegative) exponential distribution with mode μ). Identical to function `ebnm` with argument `prior_family = "point_exponential"`. For details about the model, see `ebnm`.

Usage

```
ebnm_point_exponential(
  x,
  s = 1,
  mode = 0,
  scale = "estimate",
  g_init = NULL,
  fix_g = FALSE,
```

```
  output = ebnm_output_default(),
  optmethod = NULL,
  control = NULL
)
```

Arguments

x	A vector of observations. Missing observations (NAs) are not allowed.
s	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed.
mode	A scalar specifying the mode of the prior g or "estimate" if the mode is to be estimated from the data.
scale	A scalar specifying the scale parameter of the exponential component or "estimate" if the scale is to be estimated from the data. The mean of the exponential component is the same as the value of scale.
g_init	The prior distribution g. Usually this is left unspecified (NULL) and estimated from the data. However, it can be used in conjuction with fix_g = TRUE to fix the prior (useful, for example, to do computations with the "true" g in simulations). If g_init is specified but fix_g = FALSE, g_init specifies the initial value of g used during optimization. When supplied, g_init should be an object of class gammamix or an ebnm object in which the fitted prior is an object of class gammamix.
fix_g	If TRUE, fix the prior g at g_init instead of estimating it.
output	A character vector indicating which values are to be returned. Function ebnm_output_default() provides the default return values, while ebnm_output_all() lists all possible return values. See **Value** below.
optmethod	A string specifying which optimization function is to be used. Options include "nlm" (which calls nlm), "lbfgsb" (which calls optim with method = "L-BFGS-B"), and "trust" (which calls into the trust package). Other options are "nohess_nlm", "nograd_nlm", and "nograd_lbfgsb", which use numerical approximations rather than exact expressions for the Hessian; both of the "nograd" functions use numerical approximations for the gradient as well. The default option is "nohess_nlm".
control	A list of control parameters to be passed to the optimization function specified by parameter optmethod.

Value

An ebnm object. Depending on the argument to output, the object is a list containing elements:

data A data frame containing the observations x and standard errors s.

posterior A data frame of summary results (posterior means, standard deviations, second moments, and local false sign rates).

fitted_g The fitted prior $\hat{g}$.

log_likelihood The optimal log likelihood attained, $L(\hat{g})$.

posterior_sampler A function that can be used to produce samples from the posterior. The sampler takes a single parameter nsamp, the number of posterior samples to return per observation.

S3 methods coef, confint, fitted, logLik, nobs, plot, predict, print, quantile, residuals, simulate, summary, and vcov have been implemented for ebnm objects. For details, see the respective help pages, linked below under **See Also**.

See Also

See ebnm for examples of usage and model details.

Available S3 methods include coef.ebnm, confint.ebnm, fitted.ebnm, logLik.ebnm, nobs.ebnm, plot.ebnm, predict.ebnm, print.ebnm, print.summary.ebnm, quantile.ebnm, residuals.ebnm, simulate.ebnm, summary.ebnm, and vcov.ebnm.

---

ebnm_point_laplace *Solve the EBNM problem using point-Laplace priors*

Description

Solves the empirical Bayes normal means (EBNM) problem using the family of point-Laplace priors (the family of mixtures where one component is a point mass at μ and the other is a double-exponential distribution centered at μ). Identical to function ebnm with argument prior_family = "point_laplace". For details about the model, see ebnm.

Usage

```
ebnm_point_laplace(
  x,
  s = 1,
  mode = 0,
  scale = "estimate",
  g_init = NULL,
  fix_g = FALSE,
  output = ebnm_output_default(),
  optmethod = NULL,
  control = NULL
)
```

Arguments

x	A vector of observations. Missing observations (NAs) are not allowed.
s	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed.
mode	A scalar specifying the mode of the prior g or "estimate" if the mode is to be estimated from the data.
scale	A scalar specifying the scale parameter of the Laplace component or "estimate" if the scale is to be estimated from the data.

g_init — The prior distribution g. Usually this is left unspecified (NULL) and estimated from the data. However, it can be used in conjuction with fix_g = TRUE to fix the prior (useful, for example, to do computations with the "true" g in simulations). If g_init is specified but fix_g = FALSE, g_init specifies the initial value of g used during optimization. When supplied, g_init should be an object of class laplacemix or an ebnm object in which the fitted prior is an object of class laplacemix.

fix_g — If TRUE, fix the prior g at g_init instead of estimating it.

output — A character vector indicating which values are to be returned. Function ebnm_output_default() provides the default return values, while ebnm_output_all() lists all possible return values. See **Value** below.

optmethod — A string specifying which optimization function is to be used. Options include "nlm" (which calls nlm), "lbfgsb" (which calls optim with method = "L-BFGS-B"), and "trust" (which calls into the trust package). Other options are "nohess_nlm", "nograd_nlm", and "nograd_lbfgsb", which use numerical approximations rather than exact expressions for the Hessian; both of the "nograd" functions use numerical approximations for the gradient as well. The default option is "nohess_nlm".

control — A list of control parameters to be passed to the optimization function specified by parameter optmethod.

Value

An ebnm object. Depending on the argument to output, the object is a list containing elements:

data A data frame containing the observations x and standard errors s.

posterior A data frame of summary results (posterior means, standard deviations, second moments, and local false sign rates).

fitted_g The fitted prior $\hat{g}$.

log_likelihood The optimal log likelihood attained, $L(\hat{g})$.

posterior_sampler A function that can be used to produce samples from the posterior. The sampler takes a single parameter nsamp, the number of posterior samples to return per observation.

S3 methods coef, confint, fitted, logLik, nobs, plot, predict, print, quantile, residuals, simulate, summary, and vcov have been implemented for ebnm objects. For details, see the respective help pages, linked below under **See Also**.

See Also

See ebnm for examples of usage and model details.

Available S3 methods include coef.ebnm, confint.ebnm, fitted.ebnm, logLik.ebnm, nobs.ebnm, plot.ebnm, predict.ebnm, print.ebnm, print.summary.ebnm, quantile.ebnm, residuals.ebnm, simulate.ebnm, summary.ebnm, and vcov.ebnm.

ebnm_point_mass *Solve the EBNM problem using a point mass prior*

Description

Solves the empirical Bayes normal means (EBNM) problem using the family of point masses δ_μ. Posteriors are simply point masses at μ. Identical to function ebnm with argument prior_family = "point_mass". For details about the model, see ebnm.

Usage

```
ebnm_point_mass(
  x,
  s = 1,
  mode = 0,
  g_init = NULL,
  fix_g = FALSE,
  output = ebnm_output_default(),
  control = NULL
)
```

Arguments

x	A vector of observations. Missing observations (NAs) are not allowed.
s	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed.
mode	A scalar specifying the location of the point mass or "estimate" if the location is to be estimated from the data.
g_init	The prior distribution g. Usually this is left unspecified (NULL) and estimated from the data. However, it can be used in conjuction with fix_g = TRUE to fix the prior (useful, for example, to do computations with the "true" g in simulations). If g_init is specified but fix_g = FALSE, g_init specifies the initial value of g used during optimization. When supplied, g_init should be an object of class normalmix or an ebnm object in which the fitted prior is an object of class normalmix.
fix_g	If TRUE, fix the prior g at g_init instead of estimating it.
output	A character vector indicating which values are to be returned. Function ebnm_output_default() provides the default return values, while ebnm_output_all() lists all possible return values. See **Value** below.
control	A list of control parameters to be passed to function optimize.

Value

An ebnm object. Depending on the argument to output, the object is a list containing elements:

data A data frame containing the observations x and standard errors s.

`posterior` A data frame of summary results (posterior means, standard deviations, second moments, and local false sign rates).

`fitted_g` The fitted prior $\hat{g}$.

`log_likelihood` The optimal log likelihood attained, $L(\hat{g})$.

`posterior_sampler` A function that can be used to produce samples from the posterior. The sampler takes a single parameter `nsamp`, the number of posterior samples to return per observation.

S3 methods `coef`, `confint`, `fitted`, `logLik`, `nobs`, `plot`, `predict`, `print`, `quantile`, `residuals`, `simulate`, `summary`, and `vcov` have been implemented for `ebnm` objects. For details, see the respective help pages, linked below under **See Also**.

See Also

See `ebnm` for examples of usage and model details.

Available S3 methods include `coef.ebnm`, `confint.ebnm`, `fitted.ebnm`, `logLik.ebnm`, `nobs.ebnm`, `plot.ebnm`, `predict.ebnm`, `print.ebnm`, `print.summary.ebnm`, `quantile.ebnm`, `residuals.ebnm`, `simulate.ebnm`, `summary.ebnm`, and `vcov.ebnm`.

---

ebnm_point_normal *Solve the EBNM problem using point-normal priors*

Description

Solves the empirical Bayes normal means (EBNM) problem using the family of point-normal priors (the family of mixtures where one component is a point mass at μ and the other is a normal distribution centered at μ). Identical to function `ebnm` with argument `prior_family = "point_normal"`. For details about the model, see `ebnm`.

Usage

```
ebnm_point_normal(
  x,
  s = 1,
  mode = 0,
  scale = "estimate",
  g_init = NULL,
  fix_g = FALSE,
  output = ebnm_output_default(),
  optmethod = NULL,
  control = NULL
)
```

Arguments

x	A vector of observations. Missing observations (NAs) are not allowed.
s	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed.
mode	A scalar specifying the mode of the prior g or "estimate" if the mode is to be estimated from the data.
scale	A scalar specifying the standard deviation of the normal component or "estimate" if the standard deviation is to be estimated from the data.
g_init	The prior distribution g. Usually this is left unspecified (NULL) and estimated from the data. However, it can be used in conjuction with fix_g = TRUE to fix the prior (useful, for example, to do computations with the "true" g in simulations). If g_init is specified but fix_g = FALSE, g_init specifies the initial value of g used during optimization. When supplied, g_init should be an object of class normalmix or an ebnm object in which the fitted prior is an object of class normalmix.
fix_g	If TRUE, fix the prior g at g_init instead of estimating it.
output	A character vector indicating which values are to be returned. Function ebnm_output_default() provides the default return values, while ebnm_output_all() lists all possible return values. See **Value** below.
optmethod	A string specifying which optimization function is to be used. Options include "nlm" (which calls nlm), "lbfgsb" (which calls optim with method = "L-BFGS-B"), and "trust" (which calls into the trust package). Other options are "nohess_nlm", "nograd_nlm", and "nograd_lbfgsb", which use numerical approximations rather than exact expressions for the Hessian; both of the "nograd" functions use numerical approximations for the gradient as well. The default option is "nohess_nlm".
control	A list of control parameters to be passed to the optimization function specified by parameter optmethod.

Value

An ebnm object. Depending on the argument to output, the object is a list containing elements:

data A data frame containing the observations x and standard errors s.

posterior A data frame of summary results (posterior means, standard deviations, second moments, and local false sign rates).

fitted_g The fitted prior $\hat{g}$.

log_likelihood The optimal log likelihood attained, $L(\hat{g})$.

posterior_sampler A function that can be used to produce samples from the posterior. The sampler takes a single parameter nsamp, the number of posterior samples to return per observation.

S3 methods coef, confint, fitted, logLik, nobs, plot, predict, print, quantile, residuals, simulate, summary, and vcov have been implemented for ebnm objects. For details, see the respective help pages, linked below under **See Also**.

See Also

See ebnm for examples of usage and model details.

Available S3 methods include coef.ebnm, confint.ebnm, fitted.ebnm, logLik.ebnm, nobs.ebnm, plot.ebnm, predict.ebnm, print.ebnm, print.summary.ebnm, quantile.ebnm, residuals.ebnm, simulate.ebnm, summary.ebnm, and vcov.ebnm.

---

ebnm_scale_normalmix *Set scale parameter for scale mixtures of normals*

Description

The default method for setting the scale parameter for function ebnm_normal_scale_mixture.

Usage

```
ebnm_scale_normalmix(
  x,
  s,
  mode = 0,
  min_K = 3,
  max_K = 300,
  KLdiv_target = 1/length(x)
)
```

Arguments

x	A vector of observations. Missing observations (NAs) are not allowed.
s	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed.
mode	A scalar specifying the mode of the prior g.
min_K	The minimum number of components K to include in the finite mixture of normal distributions used to approximate the nonparametric family of scale mixtures of normals.
max_K	The maximum number of components K to include in the approximating mixture of normal distributions.
KLdiv_target	The desired bound κ on the KL-divergence from the solution obtained using the approximating mixture to the exact solution. More precisely, the scale parameter is set such that given the exact MLE $$\hat{g} := \text{argmax}_{g \in G} L(g),$$ where G is the full nonparametric family, and given the MLE for the approximating family $\tilde{G}$ $$\tilde{g} := \text{argmax}_{g \in \tilde{G}} L(g),$$

we have that

$$\mathrm{KL}(\hat{g} * N(0, s^2) \mid \tilde{g} * N(0, s^2)) \leq \kappa,$$

where $* N(0, s^2)$ denotes convolution with the normal error distribution (the derivation of the bound assumes homoskedastic observations). For details, see **References** below.

References

Jason Willwerscheid (2021). *Empirical Bayes Matrix Factorization: Methods and Applications*. University of Chicago, PhD dissertation.

`ebnm_scale_npmle` *Set scale parameter for NPMLE and deconvolveR prior family*

Description

The default method for setting the `scale` parameter for functions `ebnm_npmle` and `ebnm_deconvolver`.

Usage

```
ebnm_scale_npmle(
  x,
  s,
  min_K = 3,
  max_K = 300,
  KLdiv_target = 1/length(x),
  pointmass = TRUE
)
```

Arguments

`x`	A vector of observations. Missing observations (`NA`s) are not allowed.
`s`	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed.
`min_K`	The minimum number of components K to include in the mixture of point masses used to approximate the nonparametric family of all distributions.
`max_K`	The maximum number of components K to include in the approximating mixture of point masses.
`KLdiv_target`	The desired bound κ on the KL-divergence from the solution obtained using the approximating mixture to the exact solution. More precisely, the scale parameter is set such that given the exact MLE $\hat{g} := \mathrm{argmax}_{g \in G} L(g)$, where G is the full nonparametric family, and given the MLE for the approximating family $\tilde{G}$ $\tilde{g} := \mathrm{argmax}_{g \in \tilde{G}} L(g)$,

we have that

$$\mathrm{KL}(\hat{g} * N(0, s^2) \mid \tilde{g} * N(0, s^2)) \leq \kappa,$$

where $* N(0, s^2)$ denotes convolution with the normal error distribution (the derivation of the bound assumes homoskedastic observations). For details, see **References** below.

`pointmass` When the range of the data is so large that `max_K` point masses cannot provide a good approximation to the family of all distributions, then `ebnm` will instead use a mixture of normal distributions, with the standard deviation of each component equal to `scale`/2. Setting `pointmass = FALSE` gives the default `scale` for this mixture of normal distributions.

To be exact, `ebnm` uses a mixture of normal distributions rather than a mixture of point masses when

$$\frac{\max(x) - \min(x)}{\min(s)} > 3 \max_K;$$

for a rationale, see **References** below. Note however that `ebnm` only uses a mixture of normal distributions when `scale = "estimate"`; if parameter `scale` is set manually, then a mixture of point masses will be used in all cases. To use a mixture of normal distributions with the scale set manually, an object created by the constructor function `normalmix` must be provided as argument to parameter `g_init` in function `ebnm_npmle` or function `ebnm_deconvolver`.

References

Jason Willwerscheid (2021). *Empirical Bayes Matrix Factorization: Methods and Applications*. University of Chicago, PhD dissertation.

`ebnm_scale_unimix` *Set scale parameter for nonparametric unimodal prior families*

Description

The default method for setting the `scale` parameter for functions `ebnm_unimodal`, `ebnm_unimodal_symmetric`, `ebnm_unimodal_nonnegative`, and `ebnm_unimodal_nonpositive`.

Usage

```
ebnm_scale_unimix(
  x,
  s,
  mode = 0,
  min_K = 3,
  max_K = 300,
  KLdiv_target = 1/length(x)
)
```

Arguments

x	A vector of observations. Missing observations (NAs) are not allowed.
s	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed.
mode	A scalar specifying the mode of the prior g.
min_K	The minimum number of components K to include in the finite mixture of uniform distributions used to approximate the nonparametric family of unimodal distributions.
max_K	The maximum number of components K to include in the approximating mixture of uniform distributions.
KLdiv_target	The desired bound κ on the KL-divergence from the solution obtained using the approximating mixture to the exact solution. More precisely, the scale parameter is set such that given the exact MLE $$\hat{g} := \text{argmax}_{g \in G} L(g),$$ where G is the full nonparametric family, and given the MLE for the approximating family $\tilde{G}$ $$\tilde{g} := \text{argmax}_{g \in \tilde{G}} L(g),$$ we have that $$\text{KL}(\hat{g} * N(0, s^2) \mid \tilde{g} * N(0, s^2)) \leq \kappa,$$ where $* N(0, s^2)$ denotes convolution with the normal error distribution (the derivation of the bound assumes homoskedastic observations). For details, see **References** below.

References

Jason Willwerscheid (2021). *Empirical Bayes Matrix Factorization: Methods and Applications*. University of Chicago, PhD dissertation.

---

ebnm_unimodal *Solve the EBNM problem using unimodal distributions*

Description

Solves the empirical Bayes normal means (EBNM) problem using the family of all unimodal distributions. Identical to function ebnm with argument prior_family = "unimodal". For details about the model, see ebnm.

Usage

```
ebnm_unimodal(
  x,
  s = 1,
  mode = 0,
  scale = "estimate",
  g_init = NULL,
  fix_g = FALSE,
  output = ebnm_output_default(),
  optmethod = NULL,
  control = NULL,
  ...
)
```

Arguments

x	A vector of observations. Missing observations (NAs) are not allowed.
s	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed.
mode	A scalar specifying the mode of the prior g or "estimate" if the mode is to be estimated from the data.
scale	The nonparametric family of unimodal distributions is approximated via a finite mixture of uniform distributions $\pi_1^l \text{Unif}(\mu-a_1,\mu)+\pi_1^u \text{Unif}(\mu,\mu+a_1)+\ldots+\pi_K^l \text{Unif}(\mu-a_K,\mu)+\pi_K^u \text{Unif}(\mu,\mu+a_K),$ where parameters π_k^l and π_k^u are estimated and the grid of lengths $(a_1,\ldots,a_K)$ is fixed in advance. By making the grid sufficiently dense, one can obtain an arbitrarily good approximation. The grid can be specified by the user via parameter scale, in which case the argument should be the vector of lengths $(a_1,\ldots,a_K)$; alternatively, if scale = "estimate", then ebnm sets the grid via function ebnm_scale_unimix. Note that ebnm sets the grid differently from function ash. To use the ash grid, set scale = "estimate" and pass in gridmult as an additional parameter. See ash for defaults and details.
g_init	The prior distribution g. Usually this is left unspecified (NULL) and estimated from the data. However, it can be used in conjuction with fix_g = TRUE to fix the prior (useful, for example, to do computations with the "true" g in simulations). If g_init is specified but fix_g = FALSE, g_init specifies the initial value of g used during optimization. This has the side effect of fixing the mode and scale parameters. When supplied, g_init should be an object of class unimix or an ebnm object in which the fitted prior is an object of class unimix.
fix_g	If TRUE, fix the prior g at g_init instead of estimating it.
output	A character vector indicating which values are to be returned. Function ebnm_output_default() provides the default return values, while ebnm_output_all() lists all possible return values. See **Value** below.

optmethod	A string specifying which optimization function is to be used. Options are provided by package ashr. The default method uses the mix-SQP algorithm implemented in the mixsqp package. See the ash function documentation for other options.
control	A list of control parameters to be passed to the optimization function specified by parameter optmethod.
...	Additional parameters to be passed to function ash in package ashr.

Value

An ebnm object. Depending on the argument to output, the object is a list containing elements:

data A data frame containing the observations x and standard errors s.

posterior A data frame of summary results (posterior means, standard deviations, second moments, and local false sign rates).

fitted_g The fitted prior $\hat{g}$.

log_likelihood The optimal log likelihood attained, $L(\hat{g})$.

posterior_sampler A function that can be used to produce samples from the posterior. The sampler takes a single parameter nsamp, the number of posterior samples to return per observation.

S3 methods coef, confint, fitted, logLik, nobs, plot, predict, print, quantile, residuals, simulate, summary, and vcov have been implemented for ebnm objects. For details, see the respective help pages, linked below under **See Also**.

See Also

See ebnm for examples of usage and model details.

Available S3 methods include coef.ebnm, confint.ebnm, fitted.ebnm, logLik.ebnm, nobs.ebnm, plot.ebnm, predict.ebnm, print.ebnm, print.summary.ebnm, quantile.ebnm, residuals.ebnm, simulate.ebnm, summary.ebnm, and vcov.ebnm.

---

ebnm_unimodal_nonnegative

Solve the EBNM problem using unimodal nonnegative distributions

---

Description

Solves the empirical Bayes normal means (EBNM) problem using the family of unimodal distributions with support constrained to be greater than the mode. Identical to function ebnm with argument prior_family = "unimodal_nonnegative". For details about the model, see ebnm.

Usage

```
ebnm_unimodal_nonnegative(
  x,
  s = 1,
  mode = 0,
  scale = "estimate",
  g_init = NULL,
  fix_g = FALSE,
  output = ebnm_output_default(),
  optmethod = NULL,
  control = NULL,
  ...
)
```

Arguments

x	A vector of observations. Missing observations (NAs) are not allowed.
s	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed.
mode	A scalar specifying the mode of the prior g or "estimate" if the mode is to be estimated from the data.
scale	The nonparametric family of nonnegative unimodal distributions is approximated via a finite mixture of uniform distributions $$\pi_1 \text{Unif}(\mu, \mu + a_1) + \ldots + \pi_K \text{Unif}(\mu, \mu + a_K),$$ where parameters π_k are estimated and the grid of lengths $(a_1, \ldots, a_K)$ is fixed in advance. By making the grid sufficiently dense, one can obtain an arbitrarily good approximation. The grid can be specified by the user via parameter scale, in which case the argument should be the vector of lengths $(a_1, \ldots, a_K)$; alternatively, if scale = "estimate", then ebnm sets the grid via function ebnm_scale_unimix. Note that ebnm sets the grid differently from function ash. To use the ash grid, set scale = "estimate" and pass in gridmult as an additional parameter. See ash for defaults and details.
g_init	The prior distribution g. Usually this is left unspecified (NULL) and estimated from the data. However, it can be used in conjuction with fix_g = TRUE to fix the prior (useful, for example, to do computations with the "true" g in simulations). If g_init is specified but fix_g = FALSE, g_init specifies the initial value of g used during optimization. This has the side effect of fixing the mode and scale parameters. When supplied, g_init should be an object of class unimix or an ebnm object in which the fitted prior is an object of class unimix.
fix_g	If TRUE, fix the prior g at g_init instead of estimating it.
output	A character vector indicating which values are to be returned. Function ebnm_output_default() provides the default return values, while ebnm_output_all() lists all possible return values. See **Value** below.

optmethod	A string specifying which optimization function is to be used. Options are provided by package ashr. The default method uses the mix-SQP algorithm implemented in the mixsqp package. See the ash function documentation for other options.
control	A list of control parameters to be passed to the optimization function specified by parameter optmethod.
...	Additional parameters to be passed to function ash in package ashr.

Value

An ebnm object. Depending on the argument to output, the object is a list containing elements:

data A data frame containing the observations x and standard errors s.

posterior A data frame of summary results (posterior means, standard deviations, second moments, and local false sign rates).

fitted_g The fitted prior $\hat{g}$.

log_likelihood The optimal log likelihood attained, $L(\hat{g})$.

posterior_sampler A function that can be used to produce samples from the posterior. The sampler takes a single parameter nsamp, the number of posterior samples to return per observation.

S3 methods coef, confint, fitted, logLik, nobs, plot, predict, print, quantile, residuals, simulate, summary, and vcov have been implemented for ebnm objects. For details, see the respective help pages, linked below under **See Also**.

See Also

See ebnm for examples of usage and model details.

Available S3 methods include coef.ebnm, confint.ebnm, fitted.ebnm, logLik.ebnm, nobs.ebnm, plot.ebnm, predict.ebnm, print.ebnm, print.summary.ebnm, quantile.ebnm, residuals.ebnm, simulate.ebnm, summary.ebnm, and vcov.ebnm.

---

ebnm_unimodal_nonpositive

Solve the EBNM problem using unimodal nonpositive distributions

---

Description

Solves the empirical Bayes normal means (EBNM) problem using the family of unimodal distributions with support constrained to be less than the mode. Identical to function ebnm with argument prior_family = "unimodal_nonpositive". For details about the model, see ebnm.

Usage

```
ebnm_unimodal_nonpositive(
  x,
  s = 1,
  mode = 0,
  scale = "estimate",
  g_init = NULL,
  fix_g = FALSE,
  output = ebnm_output_default(),
  optmethod = NULL,
  control = NULL,
  ...
)
```

Arguments

x	A vector of observations. Missing observations (NAs) are not allowed.
s	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed.
mode	A scalar specifying the mode of the prior g or "estimate" if the mode is to be estimated from the data.
scale	The nonparametric family of nonnpositive unimodal distributions is approximated via a finite mixture of uniform distributions $$\pi_1 \text{Unif}(\mu - a_1, \mu) + \ldots + \pi_K \text{Unif}(\mu - a_K, \mu),$$ where parameters π_k are estimated and the grid of lengths $(a_1, \ldots, a_K)$ is fixed in advance. By making the grid sufficiently dense, one can obtain an arbitrarily good approximation. The grid can be specified by the user via parameter scale, in which case the argument should be the vector of lengths $(a_1, \ldots, a_K)$; alternatively, if scale = "estimate", then ebnm sets the grid via function ebnm_scale_unimix. Note that ebnm sets the grid differently from function ash. To use the ash grid, set scale = "estimate" and pass in gridmult as an additional parameter. See ash for defaults and details.
g_init	The prior distribution g. Usually this is left unspecified (NULL) and estimated from the data. However, it can be used in conjuction with fix_g = TRUE to fix the prior (useful, for example, to do computations with the "true" g in simulations). If g_init is specified but fix_g = FALSE, g_init specifies the initial value of g used during optimization. This has the side effect of fixing the mode and scale parameters. When supplied, g_init should be an object of class unimix or an ebnm object in which the fitted prior is an object of class unimix.
fix_g	If TRUE, fix the prior g at g_init instead of estimating it.
output	A character vector indicating which values are to be returned. Function ebnm_output_default() provides the default return values, while ebnm_output_all() lists all possible return values. See **Value** below.

optmethod	A string specifying which optimization function is to be used. Options are provided by package ashr. The default method uses the mix-SQP algorithm implemented in the mixsqp package. See the ash function documentation for other options.
control	A list of control parameters to be passed to the optimization function specified by parameter optmethod.
...	Additional parameters to be passed to function ash in package ashr.

Value

An ebnm object. Depending on the argument to output, the object is a list containing elements:

data A data frame containing the observations x and standard errors s.

posterior A data frame of summary results (posterior means, standard deviations, second moments, and local false sign rates).

fitted_g The fitted prior $\hat{g}$.

log_likelihood The optimal log likelihood attained, $L(\hat{g})$.

posterior_sampler A function that can be used to produce samples from the posterior. The sampler takes a single parameter nsamp, the number of posterior samples to return per observation.

S3 methods coef, confint, fitted, logLik, nobs, plot, predict, print, quantile, residuals, simulate, summary, and vcov have been implemented for ebnm objects. For details, see the respective help pages, linked below under **See Also**.

See Also

See ebnm for examples of usage and model details.

Available S3 methods include coef.ebnm, confint.ebnm, fitted.ebnm, logLik.ebnm, nobs.ebnm, plot.ebnm, predict.ebnm, print.ebnm, print.summary.ebnm, quantile.ebnm, residuals.ebnm, simulate.ebnm, summary.ebnm, and vcov.ebnm.

---

ebnm_unimodal_symmetric

Solve the EBNM problem using symmetric unimodal distributions

---

Description

Solves the empirical Bayes normal means (EBNM) problem using the family of symmetric unimodal distributions. Identical to function ebnm with argument prior_family = "unimodal_symmetric". For details about the model, see ebnm.

Usage

```
ebnm_unimodal_symmetric(
  x,
  s = 1,
  mode = 0,
  scale = "estimate",
  g_init = NULL,
  fix_g = FALSE,
  output = ebnm_output_default(),
  optmethod = NULL,
  control = NULL,
  ...
)
```

Arguments

x	A vector of observations. Missing observations (NAs) are not allowed.
s	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed.
mode	A scalar specifying the mode of the prior g or "estimate" if the mode is to be estimated from the data.
scale	The nonparametric family of symmetric unimodal distributions is approximated via a finite mixture of uniform distributions $$\pi_1 \text{Unif}(\mu - a_1, \mu + a_1) + \ldots + \pi_K \text{Unif}(\mu - a_K, \mu + a_K),$$ where parameters π_k are estimated and the grid of (half-)lengths $(a_1, \ldots, a_K)$ is fixed in advance. By making the grid sufficiently dense, one can obtain an arbitrarily good approximation. The grid can be specified by the user via parameter scale, in which case the argument should be the vector $(a_1, \ldots, a_K)$; alternatively, if scale = "estimate", then ebnm sets the grid via function ebnm_scale_unimix. Note that ebnm sets the grid differently from function ash. To use the ash grid, set scale = "estimate" and pass in gridmult as an additional parameter. See ash for defaults and details.
g_init	The prior distribution g. Usually this is left unspecified (NULL) and estimated from the data. However, it can be used in conjuction with fix_g = TRUE to fix the prior (useful, for example, to do computations with the "true" g in simulations). If g_init is specified but fix_g = FALSE, g_init specifies the initial value of g used during optimization. This has the side effect of fixing the mode and scale parameters. When supplied, g_init should be an object of class unimix or an ebnm object in which the fitted prior is an object of class unimix.
fix_g	If TRUE, fix the prior g at g_init instead of estimating it.
output	A character vector indicating which values are to be returned. Function ebnm_output_default() provides the default return values, while ebnm_output_all() lists all possible return values. See **Value** below.

optmethod	A string specifying which optimization function is to be used. Options are provided by package ashr. The default method uses the mix-SQP algorithm implemented in the mixsqp package. See the ash function documentation for other options.
control	A list of control parameters to be passed to the optimization function specified by parameter optmethod.
...	Additional parameters to be passed to function ash in package ashr.

Value

An ebnm object. Depending on the argument to output, the object is a list containing elements:

data A data frame containing the observations x and standard errors s.

posterior A data frame of summary results (posterior means, standard deviations, second moments, and local false sign rates).

fitted_g The fitted prior $\hat{g}$.

log_likelihood The optimal log likelihood attained, $L(\hat{g})$.

posterior_sampler A function that can be used to produce samples from the posterior. The sampler takes a single parameter nsamp, the number of posterior samples to return per observation.

S3 methods coef, confint, fitted, logLik, nobs, plot, predict, print, quantile, residuals, simulate, summary, and vcov have been implemented for ebnm objects. For details, see the respective help pages, linked below under **See Also**.

See Also

See ebnm for examples of usage and model details.

Available S3 methods include coef.ebnm, confint.ebnm, fitted.ebnm, logLik.ebnm, nobs.ebnm, plot.ebnm, predict.ebnm, print.ebnm, print.summary.ebnm, quantile.ebnm, residuals.ebnm, simulate.ebnm, summary.ebnm, and vcov.ebnm.

---

fitted.ebnm *Extract posterior estimates from a fitted EBNM model*

Description

The fitted method for class ebnm. Returns a data frame that includes posterior means, standard deviations, and local false sign rates (when available).

Usage

```
## S3 method for class 'ebnm'
fitted(object, ...)
```

Arguments

object	The fitted ebnm object.
...	Not used. Included for consistency as an S3 method.

gammamix *Constructor for gammamix class*

Description

Creates a finite mixture of gamma distributions.

Usage

```
gammamix(pi, shape, scale, shift = rep(0, length(pi)))
```

Arguments

pi	A vector of mixture proportions.
shape	A vector of shape parameters.
scale	A vector of scale parameters.
shift	A vector of shift parameters.

Value

An object of class gammamix (a list with elements pi, shape, scale, and shift, described above).

horseshoe *Constructor for horseshoe class*

Description

Creates a horseshoe prior (see Carvalho, Polson, and Scott (2010)). The horseshoe is usually parametrized as $\theta_i \sim N(0, s^2\tau^2\lambda_i^2)$, $\lambda_i \sim \text{Cauchy}^+(0, 1)$, with s^2 the variance of the error distribution. We use a single parameter scale, which corresponds to $s\tau$ and thus does not depend on the error distribution.

Usage

```
horseshoe(scale)
```

Arguments

scale	The scale parameter (must be a scalar).

Value

An object of class horseshoe (a list with a single element scale, described above).

laplacemix *Constructor for laplacemix class*

Description

Creates a finite mixture of Laplace distributions.

Usage

```
laplacemix(pi, mean, scale)
```

Arguments

pi	A vector of mixture proportions.
mean	A vector of means.
scale	A vector of scale parameters.

Value

An object of class laplacemix (a list with elements pi, mean, and scale, described above).

logLik.ebnm *Extract the log likelihood from a fitted EBNM model*

Description

The logLik method for class ebnm.

Usage

```
## S3 method for class 'ebnm'
logLik(object, ...)
```

Arguments

object	The fitted ebnm object.
...	Not used. Included for consistency as an S3 method.

Value

An object of class logLik, which includes attributes df, the degrees of freedom — i.e., number of parameters in the model —, and nobs, the number of observations in the data.

nobs.ebnm *Get the number of observations used to fit an EBNM model*

Description

The nobs method for class ebnm.

Usage

```
## S3 method for class 'ebnm'
nobs(object, ...)
```

Arguments

object	The fitted ebnm object.
...	Not used. Included for consistency as an S3 method.

Value

The number of observations used to fit the ebnm object.

plot.ebnm *Plot an ebnm object*

Description

Given one or more fitted ebnm object(s), produces a plot of posterior means vs. observations. If desired, a plot of cumulative distribution functions of fitted prior(s) can also be produced.

Usage

```
## S3 method for class 'ebnm'
plot(
  x,
  ...,
  incl_pm = TRUE,
  incl_cdf = FALSE,
  subset = NULL,
  remove_abline = FALSE
)
```

Arguments

x	The fitted ebnm object.
...	Additional ebnm objects to be included on the same plots.
incl_pm	Plot posterior means vs. observations?
incl_cdf	Plot the cumulative distribution functions?
subset	The subset of observations to include on the plot of posterior means vs. observations. Can be a numeric vector corresponding to indices of observations to plot, or a character vector if observations are named. If subset = NULL then all observations will be plotted.
remove_abline	To better illustrate shrinkage effects, the plot of posterior means vs. observations includes the line $y = x$ by default. If remove_abline = TRUE, then this line will not be drawn.

Examples

```
theta <- c(rep(0, 100), rexp(100))
theta[1:50] <- 0
s <- 1
x <- theta + rnorm(200, 0, s)
pn.res <- ebnm_point_normal(x, s)
plot(pn.res)

pe.res <- ebnm_point_exponential(x, s)
plot(pn.res, pe.res)

# Customize plot:
library(ggplot2)
plot(pn.res, pe.res, remove_abline = TRUE) +
  theme_bw() +
  labs(x = "Simulated data")
```

---

predict.ebnm *Use the estimated prior from a fitted EBNM model to solve the EBNM problem for new data*

Description

The predict method for class ebnm.

Usage

```
## S3 method for class 'ebnm'
predict(object, newdata, s = 1, ...)
```

Arguments

object	The fitted ebnm object.
newdata	A vector of new observations. Missing observations (NAs) are not allowed.
s	A vector of standard errors (or a scalar if all are equal). Standard errors may not be exactly zero, and missing standard errors are not allowed. Two prior families have additional restrictions: when horseshoe priors are used, errors must be homoskedastic; and since function deconv in package deconvolveR takes z-scores, the "deconvolver" family requires that all standard errors be equal to 1.
...	Not used. Included for consistency as an S3 method.

Value

A data frame that includes posterior means, posterior standard deviations, and local false sign rates for the observations in newdata.

---

print.ebnm *Print an ebnm object*

Description

The print method for class ebnm.

Usage

```
## S3 method for class 'ebnm'
print(x, digits = 2, ...)
```

Arguments

x	The fitted ebnm object.
digits	Number of significant digits to use.
...	Not used. Included for consistency as an S3 method.

print.summary.ebnm *Print a summary.ebnm object*

Description

The print method for class summary.ebnm.

Usage

```
## S3 method for class 'summary.ebnm'
print(x, digits = 2, ...)
```

Arguments

x	The summary.ebnm object.
digits	Number of significant digits to use.
...	Not used. Included for consistency as an S3 method.

quantile.ebnm *Obtain posterior quantiles using a fitted EBNM model*

Description

The quantile method for class ebnm. Quantiles for posterior distributions $\theta_i \mid x_i, s_i, g$ are estimated via sampling. By default, ebnm does not return a posterior sampler; one can be added to the ebnm object using function ebnm_add_sampler.

Usage

```
## S3 method for class 'ebnm'
quantile(
  x,
  probs = seq(0, 1, 0.25),
  names = TRUE,
  type = 7,
  digits = 7,
  nsim = 1000,
  ...
)
```

Arguments

x	The fitted ebnm object.
probs	numeric vector of probabilities with values in $[0, 1]$. (Values up to '2e-14' outside that range are accepted and moved to the nearby endpoint.)
names	logical; if true, the result has a names attribute. Set to FALSE for speedup with many probs.
type	An integer between 1 and 9 selecting one of the nine quantile algorithms detailed in quantile to be used.
digits	used only when names is true: the precision to use when formatting the percentages. In R versions up to 4.0.x, this had been set to max(2, getOption("digits")), internally.
nsim	The number of samples to use to estimate quantiles.
...	Additional arguments to be passed to the posterior sampler function. Since ebnm_horseshoe returns an MCMC sampler, it takes parameter burn, the number of burn-in samples to discard. At present, no other samplers take any additional parameters.

Value

A matrix with columns giving quantiles for each posterior $\theta_i \mid x_i, s_i, g$.

---

residuals.ebnm *Calculate residuals for a fitted EBNM model*

Description

The residuals method for class ebnm. Calculates "residuals" $x_i - \hat{\theta}_i$.

Usage

```
## S3 method for class 'ebnm'
residuals(object, ...)
```

Arguments

object	The fitted ebnm object.
...	Not used. Included for consistency as an S3 method.

simulate.ebnm *Sample from the posterior of a fitted EBNM model*

Description

The simulate method for class ebnm. Extracts the posterior sampler from an object of class ebnm and returns a specified number of samples.

Usage

```
## S3 method for class 'ebnm'
simulate(object, nsim = 1, seed = NULL, ...)
```

Arguments

object	The fitted ebnm object.
nsim	The number of posterior samples to return per observation.
seed	Either NULL or an integer that will be used in a call to set.seed before simulating. If set, the value is saved as the "seed" attribute of the returned value. The default, NULL, will not change the random generator state.
...	Additional arguments to be passed to the posterior sampler function. Since ebnm_horseshoe returns an MCMC sampler, it takes parameter burn, the number of burn-in samples to discard. At present, no other samplers take any additional parameters.

Value

A matrix of posterior samples, with rows corresponding to distinct samples and columns corresponding to observations.

summary.ebnm *Summarize an ebnm object*

Description

The summary method for class ebnm.

Usage

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

Arguments

object	The fitted ebnm object.
...	Not used. Included for consistency as an S3 method.

Value

A summary.ebnm object.

---

vcov.ebnm *Extract posterior variances from a fitted EBNM model*

Description

The vcov method for class ebnm.

Usage

```
## S3 method for class 'ebnm'
vcov(object, ...)
```

Arguments

object	The fitted ebnm object.
...	Not used. Included for consistency as an S3 method.

---

wOBA *2022 MLB wOBA Data*

Description

Weighted on-base average (wOBA) for hitters from the 2022 Major League Baseball (MLB) season. Standard errors are calculated by modeling hitting outcomes as multinomially distributed and plugging in empirical proportions as the "true" outcome probabilities. To handle small sample sizes, standard errors are lower bounded by the errors that would be obtained using league-wide proportions rather than the plug-in estimates.

Format

wOBA is a data frame with 688 rows and 6 columns:

FanGraphsID The hitter's FanGraphs identifier.

Name The hitter's name.

Team The hitter's MLB team (given as a three-letter code) or NA if the hitter played for multiple teams.

PA The hitter's number of plate appearances over the 2022 season.

x The hitter's wOBA over the 2022 season.

s The standard error for the hitter's wOBA.

Source

<https://fangraphs.com>

Examples

```
data(wOBA)
summary(wOBA)
```

Index