Package ‘bin2norm’

November 17, 2025

Title Hierarchical Probit Estimation for Dichotomized Data

Version 0.1.0

Description Provides likelihood-based and hierarchical estimation methods for thresholded (binomial-probit) data. Supports fixed-mean and random-mean models with maximum likelihood estimation (MLE), generalized linear mixed model (GLMM), and Bayesian Markov chain Monte Carlo (MCMC) implementations. For methodological background, see Albert and Chib (1993) <doi:10.1080/01621459.1993.10476321> and McCulloch (1994) <doi:10.2307/2297959>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Imports lme4, rstan, stats, statmod

NeedsCompilation no

Author Zhaoze Liu [aut, cre]

Maintainer Zhaoze Liu <786633848@qq.com>

Repository CRAN

Date/Publication 2025-11-17 21:10:13 UTC

Contents

bin2norm	*bin2norm: A user-friendly interface to estimate normal distribution parameters from dichotomized data*

Description

This function handles two data-collection settings for estimating normal parameters from threshold-based (dichotomized) data:

- **Single-threshold per study:** Each of I studies reports one threshold c_i, a sample size n_i, and the observed proportion p_i^{obs} of samples above that threshold. We assume one normal distribution $\mathcal{N}(\mu, \sigma^2)$ across all studies. Methods include `"MLE"` and `"probit"`.
- **Multiple-thresholds per study:** Each study i reports K_i thresholds $\{c_{ij}\}$, each with an observed proportion p_{ij}^{obs}. We assume the study-specific mean $\mu_i \sim \mathcal{N}(\mu_0, \tau^2)$ and within-study variance σ^2. Because each study has multiple cutpoints, one can estimate μ_0, σ, τ. Methods include `"MLE_integration"`, `"GLMM"`, or `"Bayesian"` (MCMC).

Usage

```
bin2norm(
  scenario = c("single_threshold", "multiple_thresholds"),
  method = NULL,
  n_i = NULL,
  c_i = NULL,
  p_i_obs = NULL,
  data_list = NULL,
  ...
)
```

Arguments

`scenario` character string, either `"single_threshold"` or `"multiple_thresholds"`.

`method` character string indicating which estimation method to use.

- For `scenario = "single_threshold"`, valid `method` are `"MLE"` or `"probit"`.
- For `scenario = "multiple_thresholds"`, valid `method` are `"MLE_integration"`, `"GLMM"`, or `"Bayesian"`.

`n_i, c_i, p_i_obs`

used **only if** `scenario="single_threshold"`. Numeric vectors of the same length. n_i is study sample size, c_i is threshold, $p_{i,\mathrm{obs}}$ is observed proportion above threshold.

`data_list` used **only if** `scenario="multiple_thresholds"`, a list with:

- `n_i`: numeric vector (length I) of sample sizes
- `c_ij`: list of length I, where `c_ij[[i]]` is a numeric vector of thresholds in study i

- `p_ij_obs`: list of length I, where `p_ij_obs[[i]]` is a numeric vector of observed proportions above each threshold

`...` additional arguments passed to lower-level functions (e.g. `use_wols_init`, `gh_points`, `iter`, `chains`, etc.).

Value

A list of estimated parameters, depending on the data-collection setting (`scenario`) and the chosen method. Typically includes:

- `mu` or `mu0`
- `sigma`
- `tau` (only for multiple-threshold methods)

Examples

```
# Single-threshold example
n_i <- c(100, 120, 80)
c_i <- c(1.2, 1.0, 1.5)
p_i_obs <- c(0.30, 0.25, 0.40)
bin2norm(scenario="single_threshold", method="MLE", n_i=n_i, c_i=c_i, p_i_obs=p_i_obs)

# Multiple-thresholds example
data_list <- list(
  n_i = c(100, 120),
  c_ij = list(c(1.0,1.2), c(0.8,1.5,2.0)),
  p_ij_obs = list(c(0.20,0.30), c(0.15,0.40,0.55))
)

# MLE with numeric integration
bin2norm(scenario="multiple_thresholds", method="MLE_integration",
         data_list=data_list, gh_points=5)

# GLMM approximation
# library(lme4)
bin2norm(scenario="multiple_thresholds", method="GLMM",
         data_list=data_list, use_lme4=TRUE)

# Bayesian MCMC approach
# library(rstan)
bin2norm(scenario="multiple_thresholds", method="Bayesian",
         data_list=data_list, iter=1000, chains=2)
```

`estimate_initial_values_from_data` *Get initial values from data*

Description

Get initial values from data

Usage

```
estimate_initial_values_from_data(data_list)
```

Arguments

`data_list`	your inputs

Value

a named list of initial values

`estimate_multiThresh_GLMM` *GLMM (Multiple Thresholds per Study, Probit Link, Random Intercepts)*

Description

Creates a single data frame stacking all thresholds from all studies, then calls `lme4::glmer(..., family=binomial(link='probit'))` to fit a random-intercept model:

$$k_{ij} \sim \mathrm{Binomial}\big(n_i, \Phi(\alpha_i + \beta\, c_{ij})\big),$$

with $\alpha_i \sim \mathcal{N}(0, \sigma_\alpha^2)$.

Interpreting results: $\sigma = 1/|\,\beta\,|$, $\tau^2 = \sigma^2 \times \sigma_\alpha^2$, $\mu_0 = (\mathrm{Intercept}) \times \sigma$ (if not forced to 0).

Usage

```
estimate_multiThresh_GLMM(data_list, use_lme4 = TRUE)
```

Arguments

`data_list`	same structure: `n_i`, `c_ij`, `p_ij_obs`
`use_lme4`	logical; if `TRUE`, calls `lme4::glmer` with a probit link.

Value

A list with `mu0`, `sigma`, `tau`, `method="GLMM_probit"`.

`estimate_multiThresh_MCMC`
Bayesian MCMC (Multiple Thresholds per Study) using rstan

Description

Builds an inline Stan model for multiple thresholds per study. The user must have the `rstan` package installed. We place random effects $\mu_i = \mu_0 + \tau * mu\_raw[i]$ and use a binomial likelihood for each threshold. By default, uses simple weakly informative priors.

Usage

```
estimate_multiThresh_MCMC(data_list, iter = 2000, chains = 2)
```

Arguments

`data_list`	same structure as above: `n_i`, `c_ij`, `p_ij_obs`
`iter`	number of total iterations for each chain (default 2000)
`chains`	number of MCMC chains (default 2)

Value

a list containing `stan_fit` (the full Stan fit object), plus `mu0_est`, `sigma_est`, `tau_est` as posterior means, and `method="Bayesian_MCMC"`.

`estimate_multiThresh_MLE`
MLE with Numeric Integration (Multiple Thresholds per Study)

Description

Each study i has thresholds $\{c_{ij}\}$, each with an observed proportion p_{ij}^{obs}. We assume $\mu_i \sim \mathcal{N}(\mu_0, \tau^2)$ and $X_{ij} \sim \mathcal{N}(\mu_i, \sigma^2)$. The log-likelihood integrates out μ_i via Gauss-Hermite quadrature.

Usage

```
estimate_multiThresh_MLE(data_list, gh_points = 20)
```

Arguments

`data_list`	A list with: • `n_i`: numeric vector (length I) • `c_ij`: list of length I • `p_ij_obs`: list of length I
`gh_points`	integer; number of Gauss-Hermite points (default 12).

Value

A list with `mu0`, `sigma`, `tau`, `method="MLE_integration"`.

`estimate_singleThresh_MLE`

MLE (Single Threshold per Study)

Description

Treats the count of "above threshold" in study i as binomial with probability $1 - \Phi((c_i - \mu)/\sigma)$. This uses numerical optimization (`optim`) to maximize the binomial likelihood. Optionally uses Weighted OLS estimates as starting values to improve convergence.

Usage

```
estimate_singleThresh_MLE(n_i, c_i, p_i_obs, use_wols_init = TRUE)
```

Arguments

`n_i`	numeric vector of sample sizes
`c_i`	numeric vector of thresholds
`p_i_obs`	numeric vector of observed proportions above threshold
`use_wols_init`	logical; if `TRUE`, uses Weighted OLS estimates (`estimate_singleThresh_WOLS`) as initial values in `optim`.

Value

A list with `mu`, `sigma`, `method="MLE"`.

`estimate_singleThresh_probit`

GLM probit (Single Threshold per Study)

Description

For each group i, we assume the data follows:

$$\Pr(Y_i = 1) = \Phi\left(\frac{\mu - c_i}{\sigma}\right)$$

where c_i is a known threshold, and Φ is the standard normal CDF (the probit link). The function reconstructs individual binary outcomes based on observed probabilities, and estimates the parameters using generalized linear modeling with a probit link.

Usage

```
estimate_singleThresh_probit(n_i, c_i, p_i_obs)
```

Arguments

`n_i`	numeric vector
`c_i`	numeric vector
`p_i_obs`	numeric vector

Value

A list with `mu`, `sigma`, `method="probit"`.

`estimate_singleThresh_WOLS`
Weighted OLS (Initial value in Single Threshold per Study MLE)

Description

Implements the formula $c_i = \mu + \sigma * \Phi^{-1}(1 - p_i^{obs})$ in a weighted least-squares sense, with weights = n_i.

Usage

```
estimate_singleThresh_WOLS(n_i, c_i, p_i_obs)
```

Arguments

`n_i`	numeric vector
`c_i`	numeric vector
`p_i_obs`	numeric vector

Value

A list with `mu`, `sigma`.

`gaussHermite` *Minimal Gauss-Hermite Quadrature*

Description

Returns `(nodes, weights)` for approximating $\int f(x)e^{-x^2}dx$, ignoring any normalizing constant. This is a simple demonstration; for serious applications, more robust libraries or expansions might be used.

Usage

```
gaussHermite(n)
```

Arguments

`n`	integer number of quadrature points

Value

list with `nodes` and `weights`

Index