Package: AFTMicroMed (via r-universe)

November 24, 2026

Title Microbiome Mediation Analysis with Accelerated Failure Time Model

Version 0.0.2

Description A causal mediation framework for microbiome data with survival outcomes. An accelerated failure time model is used to estimate mediation effects for individual microbial mediators while adjusting for covariates. Microbiome mediators are modeled using a two-part marginal framework that accounts for excess zeros and non-zero relative abundances through a zero-inflated beta distribution.

License MIT + file LICENSE

Encoding UTF-8

Imports doParallel, foreach, gamlss, gamlss.dist, MASS, numDeriv, parallel, stats, survival, utils

Config/roxygen2/version 8.1.0

Depends R (>= 3.5)

LazyData true

RoxygenNote 7.3.2

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author Weijia Fu [cre, aut] (ORCID: <https://orcid.org/0009-0002-5955-3534>), Samuel Osarfo [aut], Ethan Park [aut], Alicia Yang [ctb], Sangyoon Yi [aut], Zhigang Li [ctb], Seungjun Ahn [aut] (ORCID: <https://orcid.org/0000-0002-4816-8924>)

Maintainer Weijia Fu <weijia.fu@mountsinai.org>

Repository https://cran.r-universe.dev

Date/Publication 2026-11-23 14:40:11 UTC

RemoteUrl https://github.com/cran/AFTMicroMed

RemoteRef HEAD

RemoteSha e433f0f6a81af162c106b93a37ca1f67282c0dda

Contents

AFTMicroMed-package 2
example_data 3
fit_AFTMicroMed 3
fit_model_em 5
monitor_progress 6
setup_parallel 6

Index **8**

`AFTMicroMed-package` *AFTMicroMed*

Description

Description of the AFTMicroMed package.

Author(s)

Maintainer: Weijia Fu <weijia.fu@mountsinai.org> (ORCID)

Authors:

- Samuel Osarfo <sosarfo@okstate.edu>
- Ethan Park <ejp147@scarletmail.rutgers.edu>
- Sangyoon Yi <sayi@okstate.edu>
- Seungjun Ahn <seungjun.ahn@mountsinai.org> (ORCID)

Other contributors:

- Alicia Yang <alicia.yang@icahn.mssm.edu> [contributor]
- Zhigang Li <zhigang.li@ufl.edu> [contributor]

example_data *Example Dataset*

Description

A example dataset used in the AFTMicroMed package.

Usage

```
data(example_data)
```

Format

```
example_data
```

Examples

```
data(example_data)
```

fit_AFTMicroMed *Main Function*

Description

Main function with full pipeline across all specified mediators in parallel.

Usage

```
fit_AFTMicroMed(
  mediator.tab,
  X,
  Time,
  evnt_status,
  CovZ,
  med_indices = NULL,
  keep_raw = FALSE,
  debug = FALSE,
  x1 = 0,
  x2 = 1,
  control = list(),
  init = NULL,
  log = TRUE
)
```

Arguments

mediator.tab	raw count matrix (n x p); rows are subjects, columns are mediators
X	exposure vector (length n)
Time	survival time vector (length n), ORIGINAL scale (not log-transformed - the log is applied internally)
evnt_status	event indicator: 1 = observed, 0 = censored
CovZ	covariate matrix (n x q) or NULL
med_indices	integer vector of mediator column indices to fit. If NULL (default), all mediators in mediator.tab are fitted and the user is notified.
keep_raw	if TRUE return raw list of fit objects instead of the summarised NIE table
debug	passed to summarize helpers
x1, x2	exposure contrast for NIE computation (default x1=0, x2=1)
control	list of EM control settings passed to fit_model_em
init	optional named list of per-mediator starting values, keyed by mediator index as character, e.g. list("5" = list(phi = 10), "25" = list(...))
log	if TRUE (default), write a progress log file and print per-mediator status messages. Set to FALSE to run silently

Value

a data frame with the following information: mediator index, value of beta coefficients, standard error, Z score and p-value of NIE_1, NIE_2, NIE

Examples

```
data(example_data)
CovZ <- example_data$CovZ
X <- example_data$X
Time  <- example_data$Time
Status <- example_data$status
mediator.tab <- example_data$mediator.tab
cl <- setup_parallel(n_cores = 2)
results <- fit_AFTMicroMed(
mediator.tab = mediator.tab,
X            = X,
Time         = Time,
evnt_status  = Status,
CovZ         = CovZ,
med_indices  = c(1:10),
keep_raw     = FALSE,
control      = list(verbose = FALSE),
log          = FALSE
)
parallel::stopCluster(cl)
print(results)
```

fit_model_em *fit_model_em*

Description

Fits the model for a single mediator using the EM algorithm with blockwise BFGS M-steps.

Usage

```
fit_model_em(dat_obj, target_idx, init = NULL, control = list())
```

Arguments

dat_obj	list with Y (log Time), M_obs, S_depth, X, evnt_status, and Z (NULL if no covariates). NOTE: Time must be on the original (not log)
target_idx	column index of the mediator to fit
init	optional named list of starting values; overrides the defaults from survreg/gamlss. For use in `fit_AFTMicroMed()`, supply as a named list keyed by mediator index, e.g. `list("5" = list(phi = 10), "25" = list(...))`
control	list of EM control settings: • max_iter (default 100) • reltol (default 1e-3) • abstol (default 1e-3) • maxit_omega1 (default 10) • maxit_omega2 (default 10) • maxit_outcome (default 10) • verbose (default FALSE)

Value

a list with the following information: mediator index, results of EM algorithm, and results of diagnosis

Examples

```
data(example_data)
mediator_counts <- as.matrix(example_data$mediator.tab)
S_depth <- rowSums(mediator_counts)
M_obs <- sweep(mediator_counts, 1, S_depth, FUN = "/")
dat_obj <- list(
  Y          = log(example_data$Time),
  M_obs      = M_obs,
  S_depth    = S_depth,
  X          = example_data$X,
  evnt_status = example_data$status,
  Z          = example_data$CovZ
```

```
)

fit <- fit_model_em(
  dat_obj,
  target_idx = 3,
  control = list(max_iter = 20, verbose = FALSE)
)

print(fit$em_res)
```

`monitor_progress` *monitor_progress*

Description

Prints live progress from the log file written by `fit_AFTMicroMed()`. Call this from a second R console, terminal, or shell while the analysis is running. Alternatively, from any shell: `tail -f <log_file_path>`

Usage

```
monitor_progress(log_file = NULL, interval_secs = 5)
```

Arguments

`log_file`	path returned by `fit_AFTMicroMed()`.
`interval_secs`	polling interval in seconds (default 5)

`setup_parallel` *setup_parallel*

Description

Registers a platform-independent parallel backend using PSOCK clusters, which work on Windows, Mac, and Linux, and under any HPC scheduler. The returned cluster is used by `fit_AFTMicroMed()` to fit mediators in parallel.

Usage

```
setup_parallel(n_cores)
```

Arguments

`n_cores` number of workers to start. You must supply this, as it depends on your environment:

- Desktop: total cores - 1 (leave one for the OS)
- SLURM: value of –cpus-per-task in your job script
- PBS: value of -l ncpus or ppn
- LSF: value of -n in your bsub command
- SGE: value of -pe smp

Value

A cluster object of class `cluster`. Pass it to `fit_AFTMicroMed()` and call `parallel::stopCluster()` on it when the analysis is done.

Examples

```
cl = setup_parallel(2)
parallel::stopCluster(cl)
```

Index

∗ **Example**
example_data, 3
∗ **dataset**
example_data, 3

AFTMicroMed *(*AFTMicroMed-package*)*, 2
AFTMicroMed-package, 2

example_data, 3

fit_AFTMicroMed, 3
fit_model_em, 5

monitor_progress, 6

setup_parallel, 6