Package 'prome'

May 9, 2026

Title Patient-Reported Outcome Data Analysis with Stan

Type Package

Version 4.0.2.5

Description Estimation for blinding bias in randomized controlled trials with a latent continuous outcome, a binary response depending on treatment and the latent outcome, and a noisy surrogate subject to possibly response-dependent measurement error. Implements EM estimators in R backed by compiled C routines for models with and without the restriction delta0 = 0, and Bayesian Stan wrappers for the same two models. Functions were added for latent outcome models with differential measurement error.

License Unlimited

Encoding UTF-8

Depends R (>= 4.1.0), BI

Imports stats, utils, rstan, bridgesampling

LinkingTo Rcpp, RcppEigen, StanHeaders

Suggests posterior

RoxygenNote 7.3.3

NeedsCompilation yes

Author Bin Wang [aut, cre]

Maintainer Bin Wang <bwang831@gmail.com>

Repository CRAN

Date/Publication 2026-04-26 19:50:02 UTC

Contents

`blinding.test` *Unblinding bias correction*

Description

Unblinding bias correction

Usage

```
blinding.test(W, T, G, method = "adjust", impute = TRUE)
```

Arguments

`W`	Numeric surrogate outcome.
`T`	Binary treatment indicator.
`G`	Binary response indicator.
`method`	fitting/test method.
`impute`	missing data or DNK response imputation.

Value

An object of class 'rctme_fit'.

Examples

```
## Not run:
if (requireNamespace("rstan", quietly = TRUE)) {
sigma = 1.2
sig.theta = 1.0
beta0 = 0
beta1 = 1
beta2 = 2
ntreat = nsham = 100
Tind  = c(rep(1, ntreat), rep(0,nsham))  #treatment group indicator
u1v   = rep(u1,ntreat)
u2v   = rep(u2,nsham)
uv    = c(u1v,u2v)
tauv  = uv - rep(u2, ntreat+nsham)
r = rnorm(ntreat + nsham, mean = 0, sd = sigma)
x = uv + r #actual endpoint outcome
q = 1/(1 + exp(-(beta0 + beta1*Tind + beta2*(tauv+r))))
bernGen = function(qq){rbinom(1,1,qq)}
I = sapply(q,bernGen)
rsham = rnorm(ntreat + nsham, mean = 0, sd = sig.theta)
w = x + (theta + rsham)*I
lm0 = lm(w~Tind)
tau1 = lm0$coef[2]; tau1
u12 = tapply(w,Tind,FUN=mean,na.rm=TRUE)
```

```
lm1 = lm(w~Tind+I)
tau2 = lm1$coef[2]; tau2
mydata <- data.frame(y=w,group=Tind,guess=I)
out1 <- blinding.test(W = mydata$y, T = mydata$group, G = mydata$guess,
                      method='BI',impute=TRUE)
out1 # Blinding Index test

out2 <- blinding.test(W = mydata$y, T = mydata$group, G = mydata$guess,
                      method='classic',impute=TRUE)
out2 # classic estimate (t-test like)

out3 <- blinding.test(W = mydata$y, T = mydata$group, G = mydata$guess,
                      method='adjusted',impute=TRUE)
out3 # ANOVA or linear regression using G as a covariate

out4 <- blinding.test(W = mydata$y, T = mydata$group, G = mydata$guess,
                      method='EM',impute=TRUE)
out4 # MLE using EM-algorithm

out5 <- blinding.test(W = mydata$y, T = mydata$group, G = mydata$guess,
                      method='Bayes',impute=TRUE)
out5 # MLE using Bayes MCMC
}

## End(Not run)
```

Index