Package 'CIEE'

January 20, 2025

Type Package

Title Estimating and Testing Direct Effects in Directed Acyclic Graphs using Estimating Equations

Version 0.1.1

Description In many studies across different disciplines, detailed measures of the variables of interest are available. If assumptions can be made regarding the direction of effects between the assessed variables, this has to be considered in the analysis. The functions in this package implement the novel approach CIEE (causal inference using estimating equations; Konigorski et al., 2018, <DOI:10.1002/gepi.22107>) for estimating and testing the direct effect of an exposure variable on a primary outcome, while adjusting for indirect effects of the exposure on the primary outcome through a secondary intermediate outcome and potential factors influencing the secondary outcome. The underlying directed acyclic graph (DAG) of this considered model is described in the vignette. CIEE can be applied to studies in many different fields, and it is implemented here for the analysis of a continuous primary outcome and a time-to-event primary outcome subject to censoring. CIEE uses estimating equations to obtain estimates of the direct effect and robust sandwich standard error estimates. Then, a large-sample Wald-type test statistic is computed for testing the absence of the direct effect. Additionally, standard multiple regression, regression of residuals, and the structural equation modeling approach are implemented for comparison.

License GPL-2

Encoding UTF-8

LazyData TRUE

Imports stats, survival

Suggests lavaan, knitr, rmarkdown

VignetteBuilder knitr

RoxygenNote 6.0.1

NeedsCompilation no

Author Stefan Konigorski [aut, cre], Yildiz E. Yilmaz [ctb]

Maintainer Stefan Konigorski <stefan.konigorski@gmail.com>

Repository CRAN

Date/Publication 2018-03-19 16:00:23 UTC

Contents

bootstrap_se . . . 2
ciee . . . 3
est_funct_expr . . . 5
generate_data . . . 6
get_estimates . . . 8
naive_se . . . 9
sandwich_se . . . 10
score_and_hessian_matrix_functions . . . 12
sem_appl . . . 13
summary.ciee . . . 14
traditional_regression_functions . . . 15

Index **17**

`bootstrap_se` *Bootstrap standard error estimates*

Description

Function to obtain bootstrap standard error estimates for the parameter estimates of the `get_estimates` function, under the generalized linear model (GLM) or accelerated failure time (AFT) setting for the analysis of a normally-distributed or censored time-to-event primary outcome.

Usage

```
bootstrap_se(setting = "GLM", BS_rep = 1000, Y = NULL, X = NULL,
  K = NULL, L = NULL, C = NULL)
```

Arguments

`setting`	String with value `"GLM"` or `"AFT"` indicating whether standard error estimates are obtained for a normally-distributed (`"GLM"`) or censored time-to-event (`"AFT"`) primary outcome `Y`.
`BS_rep`	Integer indicating the number of bootstrap samples that are drawn.
`Y`	Numeric input vector for the primary outcome.
`X`	Numeric input vector for the exposure variable.
`K`	Numeric input vector for the intermediate outcome.
`L`	Numeric input vector for the observed confounding factor.
`C`	Numeric input vector for the censoring indicator under the AFT setting (must be coded 0 = censored, 1 = uncensored).

Details

Under the GLM setting for the analysis of a normally-distributed primary outcome Y, bootstrap standard error estimates are obtained for the estimates of the parameters $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \sigma_1^2, \alpha_4, \alpha_{XY}, \sigma_2^2$ in the models

$$Y = \alpha_0 + \alpha_1 \cdot K + \alpha_2 \cdot X + \alpha_3 \cdot L + \epsilon_1, \epsilon_1 \sim N(0, \sigma_1^2)$$

$$Y^* = Y - \overline{Y} - \alpha_1 \cdot (K - \overline{K})$$

$$Y^* = \alpha_0 + \alpha_{XY} \cdot X + \epsilon_2, \epsilon_2 \sim N(0, \sigma_2^2),$$

accounting for the additional variability from the 2-stage approach.

Under the AFT setting for the analysis of a censored time-to-event primary outcome, bootstrap standard error estimates are similarly obtained of the parameter estimates of $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \sigma_1, \alpha_4, \alpha_{XY}, \sigma_2^2$

Value

Returns a vector with the bootstrap standard error estimates of the parameter estimates.

Examples

```
dat <- generate_data(setting = "GLM", n = 100)

# For illustration use here only 100 bootstrap samples, recommended is using 1000
bootstrap_se(setting = "GLM", BS_rep = 100, Y = dat$Y, X = dat$X,
             K = dat$K, L = dat$L)
```

`ciee` *CIEE: Causal inference based on estimating equations*

Description

Functions to perform CIEE under the GLM or AFT setting: `ciee` obtains point and standard error estimates of all parameter estimates, and p-values for testing the absence of effects; `ciee_loop` performs `ciee` in separate analyses of multiple exposure variables with the same outcome measures and factors ond only returns point estimates, standard error estimates and p-values for the exposure variables. Both functions can also compute estimates and p-values from the two traditional regression methods and from the structural equation modeling method.

Usage

```
ciee(setting = "GLM", estimates = c("ee", "mult_reg", "res_reg", "sem"),
  ee_se = c("sandwich"), BS_rep = NULL, Y = NULL, X = NULL, K = NULL,
  L = NULL, C = NULL)

ciee_loop(setting = "GLM", estimates = c("ee", "mult_reg", "res_reg",
  "sem"), ee_se = c("sandwich"), BS_rep = NULL, Y = NULL, X = NULL,
  K = NULL, L = NULL, C = NULL)
```

Arguments

setting	String with value "GLM" or "AFT" indicating whether a normally-distributed ("GLM") or censored time-to-event ("AFT") primary outcome Y is analyzed.
estimates	String vector with possible values "ee", "mult_reg", "res_reg", "sem" indicating which methods are computed. "ee" computes CIEE, "mult_reg" the traditional multiple regression method, "res_reg" the traditional regression of residuals method, and "sem" the structural equation modeling approach. Multiple methods can be specified.
ee_se	String with possible values "sandwich", "bootstrap", "naive" indicating how the standard error estimates of the parameter estiamtes are computed for CIEE approach. "sandwich" computes the robust sandwich estimates (default, recommended), "bootstrap" the bootstrap estimates and "naive" the naive unadjusted standard error estimates (not recommended, only for comparison). One method has to be specified.
BS_rep	Integer indicating the number of bootstrap samples that are drawn (recommended 1000) if bootstrap standard errors are computed.
Y	Numeric input vector for the primary outcome.
X	Numeric input vector for the exposure variable if the ciee function is used; or numeric input dataframe containing the exposure variables as columns if the ciee_loop function is used.
K	Numeric input vector for the intermediate outcome.
L	Numeric input vector for the observed confounding factor.
C	Numeric input vector for the censoring indicator under the AFT setting (must be coded 0 = censored, 1 = uncensored).

Details

For the computation of CIEE, point estimates of the parameters are obtained using the get_estimates function. Robust sandwich (recommended), bootstrap, or naive standard error estimates of the parameter estimates are obtained using the sandwich_se, bootstrap_se or naive_se function. Large-sample Wald-type tests are performed for testing the absence of effects, using either the robust sandwich or bootstrap standard errors.

Regarding the traditional regression methods, the multiple regression or regression of residual approaches can be computed using the mult_reg and res_reg functions. Finally, the structural equation modeling approachcan be performed using the sem_appl function.

Value

Object of class ciee, for which the summary function summary.ciee is implemented. ciee returns a list containing the point and standard error estimates of all parameters as well as p-values from hypothesis tests of the absence of effects, for each specified approach. ciee_loop returns a list containing the point and standard error estimates only of the exposure variables as well as p-values from hypothesis tests of the absence of effects, for each specified approach.

Examples

```
# Generate data under the GLM setting with default values
maf <- 0.2
n <- 100
dat <- generate_data(n = n, maf = maf)
datX <- data.frame(X = dat$X)
names(datX)[1] <- "X1"
# Add 9 more exposure variables names X2, ..., X10 to X
for (i in 2:10){
  X <- stats::rbinom(n, size = 2, prob = maf)
  datX$X <- X
  names(datX)[i] <- paste("X", i, sep="")
}

# Perform analysis of one exposure variable using all four methods
ciee(Y = dat$Y, X = datX$X1, K = dat$K, L = dat$L)

# Perform analysis of all exposure variables only for CIEE
ciee_loop(estimates = "ee", Y = dat$Y, X = datX, K = dat$K, L = dat$L)
```

`est_funct_expr` *Estimating functions.*

Description

Function to compute `logL1` and `logL2` under the GLM and AFT setting for the analysis of a normally-distributed and of a censored time-to-event primary outcome. `logL1` and `logL2` are functions which underlie the estimating functions of CIEE for the derivation of point estimates and standard error estimates. `est_funct_expr` computes their expression, which is then further used in the functions `deriv_obj`, `ciee` and `ciee_loop`.

Usage

```
est_funct_expr(setting = "GLM")
```

Arguments

`setting`	String with value `"GLM"` or `"AFT"` indicating whether the expression of `logL1` and `logL2` is computed under the GLM or AFT setting.

Details

Under the GLM setting for the analysis of a normally-distributed primary outcome `Y`, the goal is to obtain estimates for the pararameters $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \sigma_1^2, \alpha_4, \alpha_{XY}, \sigma_2^2$ under the model

$$Y = \alpha_0 + \alpha_1 \cdot K + \alpha_2 \cdot X + \alpha_3 \cdot L + \epsilon_1, \epsilon_1 \sim N(0, \sigma_1^2)$$

$$Y^* = Y - \overline{Y} - \alpha_1 \cdot (K - \overline{K})$$

$$Y^* = \alpha_0 + \alpha_{XY} \cdot X + \epsilon_2, \epsilon_2 \sim N(0, \sigma_2^2)$$

`logL1` underlies the estimating functions for the derivation of the first 5 parameters $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \sigma_1^2$ and `logL2` underlies the estimating functions for the derivation of the last 3 parameters $\alpha_4, \alpha_{XY}, \sigma_2^2$.

Under the AFT setting for the analysis of a censored time-to-event primary outcome `Y`, the goal is to obtain estimates of the parameters $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \sigma_1, \alpha_4, \alpha_{XY}, \sigma_2^2$. Here, `logL1` similarly underlies the estimating functions for the derivation of the first 5 parameters and `logL2` underlies the estimating functions for the derivation of the last 3 parameters.

`logL1`, `logL2` equal the log-likelihood functions (`logL2` given that α_1 is known). For more details and the underlying model, see the vignette.

Value

Returns a list containing the expression of the functions `logL1` and `logL2`.

Examples

```
est_funct_expr(setting = "GLM")
est_funct_expr(setting = "AFT")
```

`generate_data` *Data generation function*

Description

Function to generate data with `n` observations of a primary outcome `Y`, secondary outcome `K`, exposure `X`, and measured as well as unmeasured confounders `L` and `U`, where the primary outcome is a quantitative normally-distributed variable (`setting = "GLM"`) or censored time-to-event outcome under an accelerated failure time (AFT) model (`setting = "AFT"`). Under the AFT setting, the observed time-to-event variable `T=exp(Y)` as well as the censoring indicator `C` are also computed. `X` is generated as a genetic exposure variable in the form of a single nucleotide variant (SNV) in 0-1-2 additive coding with minor allele frequency `maf`. `X` can be generated independently of `U` (`X_orth_U = TRUE`) or dependent on `U` (`X_orth_U = FALSE`). For more details regarding the underlying model, see the vignette.

Usage

```
generate_data(setting = "GLM", n = 1000, maf = 0.2, cens = 0.3,
  a = NULL, b = NULL, aXK = 0.2, aXY = 0.1, aXL = 0, aKY = 0.3,
  aLK = 0, aLY = 0, aUY = 0, aUL = 0, mu_X = NULL, sd_X = NULL,
  X_orth_U = TRUE, mu_U = 0, sd_U = 1, mu_K = 0, sd_K = 1, mu_L = 0,
  sd_L = 1, mu_Y = 0, sd_Y = 1)
```

Arguments

setting	String with value "GLM" or "AFT" indicating whether the primary outcome is generated as a normally-distributed quantitative outcome ("GLM") or censored time-to-event outcome ("AFT").
n	Numeric. Sample size.
maf	Numeric. Minor allele frequency of the genetic exposure variable.
cens	Numeric. Desired percentage of censored individuals and has to be specified under the AFT setting. Note that the actual censoring rate is generated through specification of the parameters a and b, and cens is mostly used as a check whether the desired censoring rate is obtained through a and b (otherwise, a warning is issued).
a	Integer for generating the desired censoring rate under the AFT setting. Has to be specified under the AFT setting.
b	Integer for generating the desired censoring rate under the AFT setting. Has to be specified under the AFT setting.
aXK	Numeric. Size of the effect of X on K.
aXY	Numeric. Size of the effect of X on Y.
aXL	Numeric. Size of the effect of X on L.
aKY	Numeric. Size of the effect of K on Y.
aLK	Numeric. Size of the effect of L on K.
aLY	Numeric. Size of the effect of L on Y.
aUY	Numeric. Size of the effect of U on Y.
aUL	Numeric. Size of the effect of U on L.
mu_X	Numeric. Expected value of X.
sd_X	Numeric. Standard deviation of X.
X_orth_U	Logical. Indicator whether X should be generated independently of U (X_orth_U = TRUE) or dependent on U (X_orth_U = FALSE).
mu_U	Numeric. Expected value of U.
sd_U	Numeric. Standard deviation of U.
mu_K	Numeric. Expected value of K.
sd_K	Numeric. Standard deviation of K.
mu_L	Numeric. Expected value of L.
sd_L	Numeric. Standard deviation of L.
mu_Y	Numeric. Expected value of Y.
sd_Y	Numeric. Standard deviation of Y.

Value

A dataframe containing n observations of the variables Y, K, X, L, U. Under the AFT setting, T=exp(Y) and the censoring indicator C (0 = censored, 1 = uncensored) are also computed.

Examples

```
# Generate data under the GLM setting with default values
dat_GLM <- generate_data()
head(dat_GLM)

# Generate data under the AFT setting with default values
dat_AFT <- generate_data(setting = "AFT", a = 0.2, b = 4.75)
head(dat_AFT)
```

`get_estimates` *CIEE parameter point estimates*

Description

Function to perform CIEE to obtain point estimates under the GLM or AFT setting for the analysis of a normally-distributed or censored time-to-event primary outcome.

Usage

```
get_estimates(setting = "GLM", Y = NULL, X = NULL, K = NULL, L = NULL,
  C = NULL)
```

Arguments

`setting`	String with value `"GLM"` or `"AFT"` indicating whether CIEE point estimates are obtained for a normally-distributed (`"GLM"`) or censored time-to-event (`"AFT"`) primary outcome `Y`.
`Y`	Numeric input vector for the primary outcome.
`X`	Numeric input vector for the exposure variable.
`K`	Numeric input vector for the intermediate outcome.
`L`	Numeric input vector for the observed confounding factor.
`C`	Numeric input vector for the censoring indicator under the AFT setting (must be coded 0 = censored, 1 = uncensored).

Details

Under the GLM setting for the analysis of a normally-distributed primary outcome Y, estimates of the parameters $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \sigma_1^2, \alpha_4, \alpha_{XY}, \sigma_2^2$ are obtained by constructing estimating equations for the models

$$Y = \alpha_0 + \alpha_1 \cdot K + \alpha_2 \cdot X + \alpha_3 \cdot L + \epsilon_1, \epsilon_1 \sim N(0, \sigma_1^2)$$

$$Y^* = Y - \overline{Y} - \alpha_1 \cdot (K - \overline{K})$$

$$Y^* = \alpha_0 + \alpha_{XY} \cdot X + \epsilon_2, \epsilon_2 \sim N(0, \sigma_2^2).$$

Under the AFT setting for the analysis of a censored time-to-event primary outcome, estimates of the parameters $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \sigma_1, \alpha_4, \alpha_{XY}, \sigma_2^2$ are obtained by constructing similar estimating

equations based on a censored regression model and adding an additional computation to estimate the true underlying survival times. In addition to the parameter estimates, the mean of the estimated true survival times is computed and returned in the output. For more details and the underlying model, see the vignette.

For both settings, the point estimates based on estimating equations equal least squares (and maximum likelihood) estimates, and are obtained using the `lm` and `survreg` functions for computational purposes.

Value

Returns a list with point estimates of the parameters. Under the AFT setting, the mean of the estimated true survival times is also computed and returned.

Examples

```
dat_GLM <- generate_data(setting = "GLM")
get_estimates(setting = "GLM", Y = dat_GLM$Y, X = dat_GLM$X, K = dat_GLM$K,
              L = dat_GLM$L)

dat_AFT <- generate_data(setting = "AFT", a = 0.2, b = 4.75)
get_estimates(setting = "AFT", Y = dat_AFT$Y, X = dat_AFT$X, K = dat_AFT$K,
              L = dat_AFT$L, C = dat_AFT$C)
```

---

naive_se *Naive standard error estimates*

Description

Function to obtain naive standard error estimates for the parameter estimates of the `get_estimates` function, under the GLM or AFT setting for the analysis of a normally-distributed or censored time-to-event primary outcome.

Usage

```
naive_se(setting = "GLM", Y = NULL, X = NULL, K = NULL, L = NULL,
  C = NULL)
```

Arguments

`setting`	String with value `"GLM"` or `"AFT"` indicating whether standard error estimates are obtained for a normally-distributed (`"GLM"`) or censored time-to-event (`"AFT"`) primary outcome `Y`.
`Y`	Numeric input vector for the primary outcome.
`X`	Numeric input vector for the exposure variable.
`K`	Numeric input vector for the intermediate outcome.
`L`	Numeric input vector for the observed confounding factor.
`C`	Numeric input vector for the censoring indicator under the AFT setting (must be coded 0 = censored, 1 = uncensored).

Details

Under the GLM setting for the analysis of a normally-distributed primary outcome Y, naive standard error estimates are obtained for the estimates of the parameters $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_{XY}$ in the models

$$Y = \alpha_0 + \alpha_1 \cdot K + \alpha_2 \cdot X + \alpha_3 \cdot L + \epsilon_1, \epsilon_1 \sim N(0, \sigma_1^2)$$

$$Y^* = Y - \overline{Y} - \alpha_1 \cdot (K - \overline{K})$$

$$Y^* = \alpha_0 + \alpha_{XY} \cdot X + \epsilon_2, \epsilon_2 \sim N(0, \sigma_2^2),$$

using the `lm` function, without accounting for the additional variability due to the 2-stage approach.

Under the AFT setting for the analysis of a censored time-to-event primary outcome, bootstrap standard error estimates are similarly obtained of the parameter estimates of $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_{XY}$ from the output of the `survreg` and `lm` functions.

Value

Returns a vector with the naive standard error estimates of the parameter estimates.

Examples

```
dat <- generate_data(setting = "GLM")
naive_se(setting = "GLM", Y = dat$Y, X = dat$X, K = dat$K, L = dat$L)
```

---

sandwich_se *Sandwich standard error estimates*

Description

Function to obtain consistent and robust sandwich standard error estimates based on estimating equations, for the parameter estimates of the `get_estimates` function, under the GLM or AFT setting for the analysis of a normally-distributed or censored time-to-event primary outcome.

Usage

```
sandwich_se(setting = "GLM", scores = NULL, hessian = NULL)
```

Arguments

`setting`	String with value `"GLM"` or `"AFT"` indicating whether standard error estimates are obtained for a normally-distributed (`"GLM"`) or censored time-to-event (`"AFT"`) primary outcome `Y`.
`scores`	Score matrix of the parameters, which can be obtained using the `scores` function.
`hessian`	Hessian matrix of the parameters, which can be obtained using the `hessian` function.

Details

Under the GLM setting for the analysis of a normally-distributed primary outcome Y, robust sandwich standard error estimates are obtained for the estimates of the parameters $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \sigma_1^2, \alpha_4, \alpha_{XY}, \sigma_2^2$ in the model

$$Y = \alpha_0 + \alpha_1 \cdot K + \alpha_2 \cdot X + \alpha_3 \cdot L + \epsilon_1, \epsilon_1 \sim N(0, \sigma_1^2)$$

$$Y^* = Y - \overline{Y} - \alpha_1 \cdot (K - \overline{K})$$

$$Y^* = \alpha_0 + \alpha_{XY} \cdot X + \epsilon_2, \epsilon_2 \sim N(0, \sigma_2^2)$$

by using the score and hessian matrices of the parameters.

Under the AFT setting for the analysis of a censored time-to-event primary outcome, robust sandwich standard error estimates are similarly obtained of the parameter estimates of $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \sigma_1, \alpha_4, \alpha_{XY}, \sigma_2^2$. For more details and the underlying model, see the vignette.

Value

Returns a vector with the CIEE sandwich standard error estimates of the parameter estimates.

Examples

```
# Generate data including Y, K, L, X under the GLM setting
dat <- generate_data(setting = "GLM")

# Obtain estimating functions expressions
estfunct <- est_funct_expr(setting = "GLM")

# Obtain point estimates of the parameters
estimates <- get_estimates(setting = "GLM", Y = dat$Y, X = dat$X,
                           K = dat$K, L = dat$L)

# Obtain matrices with all first and second derivatives
derivobj <- deriv_obj(setting = "GLM", logL1 = estfunct$logL1,
                      logL2 = estfunct$logL2, Y = dat$Y, X = dat$X,
                      K = dat$K, L = dat$L, estimates = estimates)

# Obtain score and hessian matrices
results_scores <- scores(derivobj)
results_hessian <- hessian(derivobj)

# Obtain sandwich standard error estimates of the parameters
sandwich_se(scores = results_scores, hessian = results_hessian)
```

score_and_hessian_matrix_functions
Score and hessian matrix based on the estimating functions.

Description

Functions to compute the score and hessian matrices of the parameters based on the estimating functions, under the GLM and AFT setting for the analysis of a normally-distributed or censored time-to-event primary outcome. The score and hessian matrices are further used in the functions `sandwich_se`, `ciee` and `ciee_loop` to obtain robust sandwich error estimates of the parameter estimates of $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \sigma_1^2, \alpha_4, \alpha_{XY}, \sigma_2^2$ under the GLM setting and $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \sigma_1, \alpha_4, \alpha_{XY}, \sigma_2^2$ under the AFT setting.

Usage

```
deriv_obj(setting = "GLM", logL1 = NULL, logL2 = NULL, Y = NULL,
  X = NULL, K = NULL, L = NULL, C = NULL, estimates = NULL)

scores(derivobj = NULL)

hessian(derivobj = NULL)
```

Arguments

`setting`	String with value `"GLM"` or `"AFT"` indicating whether the matrices are computed under the GLM or AFT setting.
`logL1`	Expression of the function `logL1` generated by the `est_funct_expr` function.
`logL2`	Expression of the function `logL2` generated by the `est_funct_expr` function.
`Y`	Numeric input vector for the primary outcome.
`X`	Numeric input vector for the exposure variable.
`K`	Numeric input vector for the intermediate outcome.
`L`	Numeric input vector for the observed confounding factor.
`C`	Numeric input vector for the censoring indicator under the AFT setting (must be coded 0 = censored, 1 = uncensored).
`estimates`	Numeric input vector with point estimates of the parameters $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \sigma_1^2, \alpha_4, \alpha_{XY}, \sigma_2^2$ under the GLM setting and of $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \sigma_1, \alpha_4, \alpha_{XY}, \sigma_2^2$ under the AFT setting. Under the AFT setting, `estimates` must also contain the mean of the estimated true survival times `"y_adj_bar"`.
`derivobj`	Output of the `deriv_obj` function used as input in the `scores` and `hessian` functions.

Details

For the computation of the score and hessian matrices, first, the help function `deriv_obj` is used. In a first step, the expression of all first and second derivatives of the parameters is computed using the expressions of `logL1` and `logL2` from the `est_funct_expr` as input. Then, the numerical values of all first and second derivatives are obtained for the observed data `Y`, `X`, `K`, `L` (and `C` under the AFT setting) and point estimates (`estimates`) of the parameters, for all observed individuals.

Second, the functions `scores` and `hessian` are used to extract the relevant score and hessian matrices with respect to `logL1` and `logL2` from the output of `deriv_obj` and piece them together. For further details, see the vignette.

Value

The `deriv_obj` function returns a list with objects `logL1_deriv`, `logL2_deriv` which contain the score and hessian matrices based on `logL1`, `logL2`, respectively.

The `scores` function returns the $(n \times 8)$ score matrix.

The `hessian` function returns the $(n \times 8 \times 8)$ hessian matrix.

Examples

```
# Generate data including Y, K, L, X under the GLM setting
dat <- generate_data(setting = "GLM")

# Obtain estimating functions' expressions
estfunct <- est_funct_expr(setting = "GLM")

# Obtain point estimates of the parameters
estimates <- get_estimates(setting = "GLM", Y = dat$Y, X = dat$X,
                           K = dat$K, L = dat$L)

# Obtain matrices with all first and second derivatives
derivobj <- deriv_obj(setting = "GLM", logL1 = estfunct$logL1,
                      logL2 = estfunct$logL2, Y = dat$Y, X = dat$X,
                      K = dat$K, L = dat$L, estimates = estimates)
names(derivobj)
head(derivobj$logL1_deriv$gradient)

# Obtain score and hessian matrices
scores(derivobj)
hessian(derivobj)
```

Description

Function which uses the sem function in the lavaan package to fit the model

$$L = \alpha_0 + \alpha_1 \cdot X + \epsilon_1, \epsilon_1 \sim N(0, \sigma_1^2)$$

$$K = \alpha_2 + \alpha_3 \cdot X + \alpha_4 \cdot L + \epsilon_2, \epsilon_2 \sim N(0, \sigma_2^2)$$

$$Y = \alpha_5 + \alpha_6 \cdot K + \alpha_{XY} \cdot X + \epsilon_3, \epsilon_3 \sim N(0, \sigma_3^2)$$

in order to obtain point and standard error estimates of the parameters $\alpha_1, \alpha_3, \alpha_4, \alpha_6, \alpha_{XY}$ for the GLM setting. See the vignette for more details.

Usage

```
sem_appl(Y = NULL, X = NULL, K = NULL, L = NULL)
```

Arguments

Y	Numeric input vector for the primary outcome.
X	Numeric input vector for the exposure variable.
K	Numeric input vector for the intermediate outcome.
L	Numeric input vector for the observed confounding factor.

Value

Returns a list with point estimates of the parameters (point_estimates), standard error estimates (SE_estimates) and p-values from large-sample Wald-type tests (pvalues).

Examples

```
dat <- generate_data(setting = "GLM")
sem_appl(Y = dat$Y, X = dat$X, K = dat$K, L = dat$L)
```

---

summary.ciee *Summary function.*

Description

Summary function for the ciee and ciee_loop functions.

Usage

```
## S3 method for class 'ciee'
summary(object = NULL, ...)
```

Arguments

object	ciee object (output of the ciee or ciee_loop function).
...	Additional arguments affecting the summary produced.

Value

Formatted data frames of the results of all computed methods.

Examples

```
maf <- 0.2
n <- 1000
dat <- generate_data(n = n, maf = maf)
datX <- data.frame(X = dat$X)
names(datX)[1] <- "X1"
for (i in 2:10){
  X <- stats::rbinom(n, size = 2, prob = maf)
  datX$X <- X
  names(datX)[i] <- paste("X", i, sep="")
}

results1 <- ciee(Y = dat$Y, X = datX$X1, K = dat$K, L = dat$L)
summary(results1)

results2 <- ciee_loop(Y = dat$Y, X = datX, K = dat$K, L = dat$L)
summary(results2)
```

---

traditional_regression_functions

Traditional regression approaches.

Description

Functions to fit traditional regression approaches for a quantitative normally-distributed primary outcome (setting = "GLM") and a censoredtime-to-event primary outcome (setting = "AFT"). mult_reg fits the multiple regression approach and res_reg computes the regression of residuals approach.

Usage

```
mult_reg(setting = "GLM", Y = NULL, X = NULL, K = NULL, L = NULL,
  C = NULL)

res_reg(Y = NULL, X = NULL, K = NULL, L = NULL)
```

Arguments

setting	String with value "GLM" or "AFT" indicating whether the approaches are fitted for a normally-distributed primary outcome Y ("GLM") or a censored time-to-event primary outcome Y ("AFT"). Under the "AFT" setting, only mult_reg is available.
Y	Numeric input vector of the primary outcome.
X	Numeric input vector of the exposure variable.
K	Numeric input vector of the intermediate outcome.
L	Numeric input vector of the observed confounding factor.
C	Numeric input vector of the censoring indicator under the AFT setting (must be coded 0 = censored, 1 = uncensored).

Details

In more detail, for a quantitative normally-distributed primary outcome Y, mult_reg fits the model

$$Y = \alpha_0 + \alpha_1 \cdot K + \alpha_{XY} \cdot X + \alpha_2 \cdot L + \epsilon$$

and obtains point and standard error estimates for the parameters $\alpha_0, \alpha_1, \alpha_{XY}, \alpha_2$. res_reg obtains point and standard error estimates for the parameters $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \alpha_{XY}$ by fitting the models

$$Y = \alpha_0 + \alpha_1 \cdot K + \alpha_2 \cdot L + \epsilon_1$$

$$\widehat{\epsilon_1} = \alpha_3 + \alpha_{XY} \cdot X + \epsilon_2$$

Both functions use the lm function and also report the provided p-values from t-tests that each parameter equals 0. For the analysis of a censored time-to-event primary outcome Y, only the multiple regression approach is implemented. Here, mult_reg fits the according censored regression model to obtain coefficient and standard error estimates as well as p-values from large-sample Wald-type tests by using the survreg function. See the vignette for more details.

Value

Returns a list with point estimates of the parameters point_estimates, standard error estimates SE_estimates and p-values pvalues.

Examples

```
dat_GLM <- generate_data(setting = "GLM")
mult_reg(setting = "GLM", Y = dat_GLM$Y, X = dat_GLM$X, K = dat_GLM$K,
         L = dat_GLM$L)
res_reg(Y = dat_GLM$Y, X = dat_GLM$X, K = dat_GLM$K, L = dat_GLM$L)

dat_AFT <- generate_data(setting = "AFT", a = 0.2, b = 4.75)
mult_reg(setting = "AFT", Y = dat_AFT$Y, X = dat_AFT$X, K = dat_AFT$K,
         L = dat_AFT$L, C = dat_AFT$C)
```

Index

bootstrap_se, 2, *4*

ciee, *3*, 3, *4, 5*, *12*, *14, 15*
ciee_loop, *3–5*, *12*, *14, 15*
ciee_loop *(*ciee*)*, 3

deriv_obj, *5*, *12, 13*
deriv_obj
*(*score_and_hessian_matrix_functions*)*, 12

est_funct_expr, *5*, 5, *12, 13*

generate_data, 6
get_estimates, *2*, *4*, 8, *9, 10*

hessian, *10*, *12, 13*
hessian
*(*score_and_hessian_matrix_functions*)*, 12

lm, *9, 10*, *16*

mult_reg, *4*, *15, 16*
mult_reg
*(*traditional_regression_functions*)*, 15

naive_se, *4*, 9

res_reg, *4*, *15, 16*
res_reg
*(*traditional_regression_functions*)*, 15

sandwich_se, *4*, 10, *12*
score_and_hessian_matrix_functions, 12
scores, *10*, *12, 13*
scores
*(*score_and_hessian_matrix_functions*)*, 12
sem, *14*
sem_appl, *4*, 13
summary.ciee, *4*, 14
survreg, *9, 10*, *16*

traditional_regression_functions, 15