Package 'lmtp'

September 7, 2022

Title Non-Parametric Causal Effects of Feasible Interventions Based on Modified Treatment Policies

Version 1.3.1

Description Non-parametric estimators for casual effects based on longitudinal modified treatment policies as described in Diaz, Williams, Hoffman, and Schenck <doi:10.1080/01621459.2021.1955691>, traditional point treatment, and traditional longitudinal effects. Continuous, binary, and categorical treatments are allowed as well are censored outcomes. The treatment mechanism is estimated via a density ratio classification procedure irrespective of treatment variable type. For both continuous and binary outcomes, additive treatment effects can be calculated and relative risks and odds ratios may be calculated for binary outcomes.

Depends R (>= 2.10)

License AGPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.2.1

Imports stats, nnls, cli, R6, generics, origami, future (>= 1.17.0), progressr, data.table (>= 1.13.0), checkmate (>= 2.1.0), SuperLearner

URL https://github.com/nt-williams/lmtp

BugReports https://github.com/nt-williams/lmtp/issues

Suggests testthat (>= 2.1.0), covr, rmarkdown, knitr, ranger, twang

NeedsCompilation no

Author Nicholas Williams [aut, cre, cph] (<https://orcid.org/0000-0002-1378-4831>), Iván Díaz [aut, cph] (<https://orcid.org/0000-0001-9056-2047>)

Maintainer Nicholas Williams <ntwilliams.personal@gmail.com>

Repository CRAN

Date/Publication 2022-09-07 21:32:54 UTC

R topics documented:

`create_node_list` *Create a node list specification*

Description

Creates a node list specification that is used by the provided estimators. `create_node_list()` is not explicitly called by the analyst, rather it is provided so the analyst can confirm how estimators will use variables before actually performing the estimation procedure.

Usage

```
create_node_list(trt, tau, time_vary = NULL, baseline = NULL, k = Inf)
```

Arguments

`trt`	A vector of column names of treatment variables.
`tau`	The number of time points of observation, excluding the final outcome.
`time_vary`	A list of length tau with the column names for new time_vary to be introduced at each time point. The list should be ordered following the time ordering of the model.
`baseline`	An optional vector of columns names for baseline covariates to be included for adjustment at every timepoint.
`k`	An integer specifying how previous time points should be used for estimation at the given time point. Default is `Inf`, all time points.

Value

A list of lists. Each sub-list is the same length of the `time_vary` parameter with the variables to be used for estimation at that given time point for either the treatment mechanism or outcome regression.

Examples

```
a <- c("A_1", "A_2", "A_3", "A_4")
bs <- c("W_1", "W_2")
time_vary <- list(c("L_1"), c("L_2"), c("L_3"), c("L_4"))

# assuming no Markov property
create_node_list(a, 4, time_vary, bs, k = Inf)

# assuming a Markov property
create_node_list(a, 4, time_vary, bs, k = 0)
```

---

event_locf *Time To Event Last Outcome Carried Forward*

Description

A helper function to prepare survival data for use with LMTP estimators by imputing outcome nodes using last outcome carried forward when an observation experiences the event before the end-of-follow-up.

Usage

```
event_locf(data, outcomes)
```

Arguments

`data`	The dataset to modify.
`outcomes`	A vector of outcome nodes ordered by time.

Value

A modified dataset with future outcome nodes set to 1 if an observation experienced an event at any previous time point.

Examples

```
event_locf(sim_point_surv, paste0("Y.", 1:6))
```

`lmtp_contrast` *Perform Contrasts of LMTP Fits*

Description

Estimates contrasts of multiple LMTP fits compared to either a known reference value or a reference LMTP fit.

Usage

```
lmtp_contrast(..., ref, type = c("additive", "rr", "or"))
```

Arguments

`...`	One or more objects of class lmtp.
`ref`	A reference value or another lmtp fit to compare all other fits against.
`type`	The contrasts of interest. Options are "additive" (the default), "rr", and "or".

Value

A list of class `lmtp_contrast` containing the following components:

`type`	The type of contrast performed.
`null`	The null hypothesis.
`vals`	A dataframe containing the contrasts estimates, standard errors, and confidence intervals.
`eifs`	Un-centered estimated influence functions for contrasts estimated.

Examples

```
a <- c("A1", "A2")
nodes <- list(c("L1"), c("L2"))
cens <- c("C1", "C2")
y <- "Y"

# mean population outcome
psi_null <- lmtp_tmle(sim_cens, a, y, time_vary = nodes, cens = cens, shift = NULL, folds = 1)

# treatment rule, everyone is increased by 0.5
d <- function(data, x) data[[x]] + 0.5
psi_rule1 <- lmtp_tmle(sim_cens, a, y, time_vary = nodes, cens = cens,
                       shift = d, folds = 1, mtp = TRUE)

# treatment rule, everyone is decreased by 0.5
d <- function(data, x) data[[x]] - 0.5
psi_rule2 <- lmtp_tmle(sim_cens, a, y, time_vary = nodes, cens = cens,
```

```
                        shift = d, folds = 1, mtp = TRUE)

# Example 1.1
# Additive effect of rule 1 compared to a known constant
lmtp_contrast(psi_rule1, ref = 0.9)

# Example 1.2
# Additive effect of rule 1 compared to the population mean outcome
lmtp_contrast(psi_rule1, ref = psi_null)

# Example 1.3
# Additive effects of rule 1 and 2 compared to the population mean outcome
lmtp_contrast(psi_rule1, psi_rule2, ref = psi_null)

# Example 1.4
# Relative risk of rule 1 compared to observed exposure
lmtp_contrast(psi_rule1, ref = psi_null, type = "rr")

# Example 1.5
# Odds of rule 1 compared to observed exposure
lmtp_contrast(psi_rule1, ref = psi_null, type = "or")
```

`lmtp_ipw` *LMTP IPW Estimator*

Description

Inverse probability of treatment weighting estimator for the effects of traditional causal effects and modified treatment policies for both point treatment and longitudinal data with binary, continuous, or time-to-event outcomes. Supports binary, categorical, and continuous exposures.

Usage

```
lmtp_ipw(
  data,
  trt,
  outcome,
  baseline = NULL,
  time_vary = NULL,
  cens = NULL,
  shift = NULL,
  shifted = NULL,
  mtp = FALSE,
  k = Inf,
  id = NULL,
  outcome_type = c("binomial", "continuous", "survival"),
  learners = "SL.glm",
  folds = 10,
```

```
  weights = NULL,
  .bound = 1e-05,
  .trim = 0.999,
  .learners_folds = 10,
  .return_full_fits = FALSE,
  ...
)
```

Arguments

data	[data.frame] A data.frame in wide format containing all necessary variables for the estimation problem. Must not be a data.table.
trt	[character] A vector containing the column names of treatment variables ordered by time.
outcome	[character] The column name of the outcome variable. In the case of time-to-event analysis, a vector containing the columns names of intermediate outcome variables and the final outcome variable ordered by time. Only numeric values are allowed. If the outcome type is binary, data should be coded as 0 and 1.
baseline	[character] An optional vector containing the column names of baseline covariates to be included for adjustment at every time point.
time_vary	[list] A list the same length as the number of time points of observation with the column names for new time-varying covariates introduced at each time point. The list should be ordered following the time ordering of the model.
cens	[character] An optional vector of column names of censoring indicators the same length as the number of time points of observation. If missingness in the outcome is present or if time-to-event outcome, must be provided.
shift	[closure] A two argument function that specifies how treatment variables should be shifted. See examples for how to specify shift functions for continuous, binary, and categorical exposures.
shifted	[data.frame] An optional data frame, the same as in data, but modified according to the treatment policy of interest. If specified, shift is ignored.
mtp	[logical(1)] Is the intervention of interest a modified treatment policy? Default is FALSE. If treatment variables are continuous this should be TRUE.
k	[integer(1)] An integer specifying how previous time points should be used for estimation at the given time point. Default is Inf, all time points.
id	[character(1)] An optional column name containing cluster level identifiers.

`outcome_type` [character(1)]
Outcome variable type (i.e., continuous, binomial, survival).

`learners` [character]
A vector of SuperLearner algorithms for estimation of the exposure mechanism. Default is "SL.glm", a main effects GLM. **Only include candidate learners capable of binary classification**.

`folds` [integer(1)]
The number of folds to be used for cross-fitting.

`weights` [numeric(nrow(data))]
An optional vector containing sampling weights.

`.bound` [numeric(1)]
Determines that maximum and minimum values (scaled) predictions will be bounded by. The default is 1e-5, bounding predictions by 1e-5 and 0.9999.

`.trim` [numeric(1)]
Determines the amount the density ratios should be trimmed. The default is 0.999, trimming the density ratios greater than the 0.999 percentile to the 0.999 percentile. A value of 1 indicates no trimming.

`.learners_folds` [integer(1)]
The number of cross-validation folds for learners.

`.return_full_fits` [logical(1)]
Return full SuperLearner fits? Default is FALSE, return only SuperLearner weights.

`...` Extra arguments. Exists for backwards compatibility.

Details

Should `mtp = TRUE`**?:**
A modified treatment policy (MTP) is an intervention that depends on the natural value of the exposure (the value that the treatment would have taken under no intervention). This differs from other causal effects, such as the average treatment effect (ATE), where an exposure would be increased (or decreased) deterministically. **If your intervention of interest adds, subtracts, or multiplies the observed treatment values by some amount, use** `mtp = TRUE`.

Value

A list of class `lmtp` containing the following components:

`estimator` The estimator used, in this case "IPW".

`theta` The estimated population LMTP effect.

`standard_error` NA

`low` NA

`high` NA

`shift` The shift function specifying the treatment policy of interest.

`density_ratios` An n x Tau matrix of the estimated density ratios.

`fits_r` A list the same length as `folds`, containing the fits at each time-point for each fold of density ratio estimation.

Examples

```
set.seed(56)
n <- 1000
W <- rnorm(n, 10, 5)
A <- 23 + 5*W + rnorm(n)
Y <- 7.2*A + 3*W + rnorm(n)
ex1_dat <- data.frame(W, A, Y)

# Example 1.1
# Point treatment, continuous exposure, continuous outcome, no loss-to-follow-up
# Interested in the effect of a modified treatment policy where A is decreased by 15
# units only among observations whose observed A was above 80.
# The true value under this intervention is about 513.
policy <- function(data, x) {
  (data[[x]] > 80)*(data[[x]] - 15) + (data[[x]] <= 80)*data[[x]]
}

lmtp_ipw(ex1_dat, "A", "Y", "W", mtp = TRUE, shift = policy,
         outcome_type = "continuous", folds = 2)

# Example 2.1
# Longitudinal setting, time-varying continuous exposure bounded by 0,
# time-varying covariates, and a binary outcome with no loss-to-follow-up.
# Interested in the effect of a treatment policy where exposure decreases by
# one unit at every time point if an observations observed exposure is greater
# than or equal to 2. The true value under this intervention is about 0.305.
head(sim_t4)

A <- c("A_1", "A_2", "A_3", "A_4")
L <- list(c("L_1"), c("L_2"), c("L_3"), c("L_4"))

policy <- function(data, trt) {
  a <- data[[trt]]
  (a - 1) * (a - 1 >= 1) + a * (a - 1 < 1)
}

# BONUS: progressr progress bars!
progressr::handlers(global = TRUE)

lmtp_ipw(sim_t4, A, "Y", time_vary = L,
         shift = policy, folds = 2, mtp = TRUE)

# Example 2.2
# The previous example assumed that the outcome (as well as the treatment variables)
# were directly affected by all other nodes in the past. In certain situations,
# domain specific knowledge may suggest otherwise.
# This can be controlled using the k argument.
lmtp_ipw(sim_t4, A, "Y", time_vary = L, mtp = TRUE,
         shift = policy, k = 0, folds = 2)

# Example 2.3
```

```
# Using the same data as examples 2.1 and 2.2.
# Now estimating the effect of a dynamic modified treatment policy.
# creating a dynamic mtp that applies the shift function
# but also depends on history and the current time
policy <- function(data, trt) {
  mtp <- function(data, trt) {
    (data[[trt]] - 1) * (data[[trt]] - 1 >= 1) + data[[trt]] * (data[[trt]] - 1 < 1)
  }

  # if its the first time point, follow the same mtp as before
  if (trt == "A_1") return(mtp(data, trt))

  # otherwise check if the time varying covariate equals 1
  ifelse(
    data[[sub("A", "L", trt)]] == 1,
    mtp(data, trt), # if yes continue with the policy
    data[[trt]]     # otherwise do nothing
  )
}

lmtp_ipw(sim_t4, A, "Y", time_vary = L,
         k = 0, mtp = TRUE, shift = policy, folds = 2)

# Example 2.4
# Using the same data as examples 2.1, 2.2, and 2.3, but now treating the exposure
# as an ordered categorical variable. To account for the exposure being a
# factor we just need to modify the shift function (and the original data)
# so as to respect this.
tmp <- sim_t4
for (i in A) {
  tmp[[i]] <- factor(tmp[[i]], levels = 0:5, ordered = TRUE)
}

policy <- function(data, trt) {
  out <- list()
  a <- data[[trt]]
  for (i in 1:length(a)) {
    if (as.character(a[i]) %in% c("0", "1")) {
      out[[i]] <- as.character(a[i])
    } else {
      out[[i]] <- as.numeric(as.character(a[i])) - 1
    }
  }
  factor(unlist(out), levels = 0:5, ordered = TRUE)
}

lmtp_ipw(tmp, A, "Y", time_vary = L, shift = policy,
         k = 0, folds = 2, mtp = TRUE)

# Example 3.1
# Longitudinal setting, time-varying binary treatment, time-varying covariates
# and baseline covariates with no loss-to-follow-up. Interested in a "traditional"
# causal effect where treatment is set to 1 at all time points for all observations.
```

```
if (require("twang")) {
  data("iptwExWide", package = "twang")

  A <- paste0("tx", 1:3)
  W <- c("gender", "age")
  L <- list(c("use0"), c("use1"), c("use2"))

  lmtp_ipw(iptwExWide, A, "outcome", baseline = W, time_vary = L,
           shift = static_binary_on, outcome_type = "continuous",
           mtp = FALSE, folds = 2)
}

# Example 4.1
# Longitudinal setting, time-varying continuous treatment, time-varying covariates,
# binary outcome with right censoring. Interested in the mean population outcome under
# the observed exposures in a hypothetical population with no loss-to-follow-up.
head(sim_cens)

A <- c("A1", "A2")
L <- list(c("L1"), c("L2"))
C <- c("C1", "C2")
Y <- "Y"

lmtp_ipw(sim_cens, A, Y, time_vary = L, cens = C, shift = NULL, folds = 2)

# Example 5.1
# Time-to-event analysis with a binary time-invariant exposure. Interested in
# the effect of treatment being given to all observations on the cumulative
# incidence of the outcome.
# For a survival problem, the outcome argument now takes a vector of outcomes
# if an observation experiences the event prior to the end of follow-up, all future
# outcome nodes should be set to 1 (i.e., last observation carried forward).
A <- "trt"
Y <- paste0("Y.", 1:6)
C <- paste0("C.", 0:5)
W <- c("W1", "W2")

lmtp_ipw(sim_point_surv, A, Y, W, cens = C, folds = 2,
          shift = static_binary_on, outcome_type = "survival")
```

`lmtp_sdr` *LMTP Sequential Doubly Robust Estimator*

Description

Sequentially doubly robust estimator for the effects of traditional causal effects and modified treatment policies for both point treatment and longitudinal data with binary, continuous, or time-to-event outcomes. Supports binary, categorical, and continuous exposures.

Usage

```
lmtp_sdr(
  data,
  trt,
  outcome,
  baseline = NULL,
  time_vary = NULL,
  cens = NULL,
  shift = NULL,
  shifted = NULL,
  k = Inf,
  mtp = FALSE,
  outcome_type = c("binomial", "continuous", "survival"),
  id = NULL,
  bounds = NULL,
  learners_outcome = "SL.glm",
  learners_trt = "SL.glm",
  folds = 10,
  weights = NULL,
  .bound = 1e-05,
  .trim = 0.999,
  .learners_outcome_folds = 10,
  .learners_trt_folds = 10,
  .return_full_fits = FALSE,
  ...
)
```

Arguments

`data`	[`data.frame`] A `data.frame` in wide format containing all necessary variables for the estimation problem. Must not be a `data.table`.
`trt`	[`character`] A vector containing the column names of treatment variables ordered by time.
`outcome`	[`character`] The column name of the outcome variable. In the case of time-to-event analysis, a vector containing the columns names of intermediate outcome variables and the final outcome variable ordered by time. Only numeric values are allowed. If the outcome type is binary, data should be coded as 0 and 1.
`baseline`	[`character`] An optional vector containing the column names of baseline covariates to be included for adjustment at every time point.
`time_vary`	[`list`] A list the same length as the number of time points of observation with the column names for new time-varying covariates introduced at each time point. The list should be ordered following the time ordering of the model.

`cens` [character]
An optional vector of column names of censoring indicators the same length as the number of time points of observation. If missingness in the outcome is present or if time-to-event outcome, must be provided.

`shift` [closure]
A two argument function that specifies how treatment variables should be shifted. See examples for how to specify shift functions for continuous, binary, and categorical exposures.

`shifted` [data.frame]
An optional data frame, the same as in `data`, but modified according to the treatment policy of interest. If specified, `shift` is ignored.

`k` [integer(1)]
An integer specifying how previous time points should be used for estimation at the given time point. Default is `Inf`, all time points.

`mtp` [logical(1)]
Is the intervention of interest a modified treatment policy? Default is `FALSE`. If treatment variables are continuous this should be `TRUE`.

`outcome_type` [character(1)]
Outcome variable type (i.e., continuous, binomial, survival).

`id` [character(1)]
An optional column name containing cluster level identifiers.

`bounds` [numeric(2)]
An optional, ordered vector of the bounds for a continuous outcomes. If `NULL`, the bounds will be taken as the minimum and maximum of the observed data. Should be left as `NULL` if the outcome type is binary.

`learners_outcome` [character]
A vector of `SuperLearner` algorithms for estimation of the outcome regression. Default is `"SL.glm"`, a main effects GLM.

`learners_trt` [character]
A vector of `SuperLearner` algorithms for estimation of the exposure mechanism. Default is `"SL.glm"`, a main effects GLM. **Only include candidate learners capable of binary classification**.

`folds` [integer(1)]
The number of folds to be used for cross-fitting.

`weights` [numeric(nrow(data))]
An optional vector containing sampling weights.

`.bound` [numeric(1)]
Determines that maximum and minimum values (scaled) predictions will be bounded by. The default is 1e-5, bounding predictions by 1e-5 and 0.9999.

`.trim` [numeric(1)]
Determines the amount the density ratios should be trimmed. The default is 0.999, trimming the density ratios greater than the 0.999 percentile to the 0.999 percentile. A value of 1 indicates no trimming.

`.learners_outcome_folds`
: [integer(1)]
The number of cross-validation folds for `learners_outcome`.

`.learners_trt_folds`
: [integer(1)]
The number of cross-validation folds for `learners_trt`.

`.return_full_fits`
: [logical(1)]
Return full SuperLearner fits? Default is `FALSE`, return only SuperLearner weights.

`...`
: Extra arguments. Exists for backwards compatibility.

Details

Should `mtp = TRUE`?:

A modified treatment policy (MTP) is an intervention that depends on the natural value of the exposure (the value that the treatment would have taken under no intervention). This differs from other causal effects, such as the average treatment effect (ATE), where an exposure would be increased (or decreased) deterministically. **If your intervention of interest adds, subtracts, or multiplies the observed treatment values by some amount, use `mtp = TRUE`.**

Value

A list of class `lmtp` containing the following components:

`estimator`	The estimator used, in this case "SDR".
`theta`	The estimated population LMTP effect.
`standard_error`	The estimated standard error of the LMTP effect.
`low`	Lower bound of the 95% confidence interval of the LMTP effect.
`high`	Upper bound of the 95% confidence interval of the LMTP effect.
`eif`	The estimated, un-centered, influence function of the estimate.
`shift`	The shift function specifying the treatment policy of interest.
`outcome_reg`	An n x Tau + 1 matrix of outcome regression predictions. The mean of the first column is used for calculating theta.
`density_ratios`	An n x Tau matrix of the estimated, non-cumulative, density ratios.
`fits_m`	A list the same length as `folds`, containing the fits at each time-point for each fold for the outcome regression.
`fits_r`	A list the same length as `folds`, containing the fits at each time-point for each fold of density ratio estimation.
`outcome_type`	The outcome variable type.

Examples

```
set.seed(56)
n <- 1000
W <- rnorm(n, 10, 5)
```

```
A <- 23 + 5*W + rnorm(n)
Y <- 7.2*A + 3*W + rnorm(n)
ex1_dat <- data.frame(W, A, Y)

# Example 1.1
# Point treatment, continuous exposure, continuous outcome, no loss-to-follow-up
# Interested in the effect of a modified treatment policy where A is decreased by 15
# units only among observations whose observed A was above 80.
# The true value under this intervention is about 513.
policy <- function(data, x) {
  (data[[x]] > 80)*(data[[x]] - 15) + (data[[x]] <= 80)*data[[x]]
}

lmtp_sdr(ex1_dat, "A", "Y", "W", shift = policy,
         outcome_type = "continuous", folds = 2, mtp = TRUE)

# Example 2.1
# Longitudinal setting, time-varying continuous exposure bounded by 0,
# time-varying covariates, and a binary outcome with no loss-to-follow-up.
# Interested in the effect of a treatment policy where exposure decreases by
# one unit at every time point if an observations observed exposure is greater
# than or equal to 2. The true value under this intervention is about 0.305.
head(sim_t4)

A <- c("A_1", "A_2", "A_3", "A_4")
L <- list(c("L_1"), c("L_2"), c("L_3"), c("L_4"))

policy <- function(data, trt) {
  a <- data[[trt]]
  (a - 1) * (a - 1 >= 1) + a * (a - 1 < 1)
}

# BONUS: progressr progress bars!
progressr::handlers(global = TRUE)

lmtp_sdr(sim_t4, A, "Y", time_vary = L, shift = policy,
         folds = 2, mtp = TRUE)

# Example 2.2
# The previous example assumed that the outcome (as well as the treatment variables)
# were directly affected by all other nodes in the past. In certain situations,
# domain specific knowledge may suggest otherwise.
# This can be controlled using the k argument.
lmtp_sdr(sim_t4, A, "Y", time_vary = L, shift = policy,
         k = 0, folds = 2, mtp = TRUE)

# Example 2.3
# Using the same data as examples 2.1 and 2.2.
# Now estimating the effect of a dynamic modified treatment policy.
# creating a dynamic mtp that applies the shift function
# but also depends on history and the current time
policy <- function(data, trt) {
  mtp <- function(data, trt) {
```

```
    (data[[trt]] - 1) * (data[[trt]] - 1 >= 1) + data[[trt]] * (data[[trt]] - 1 < 1)
  }

  # if its the first time point, follow the same mtp as before
  if (trt == "A_1") return(mtp(data, trt))

  # otherwise check if the time varying covariate equals 1
  ifelse(
    data[[sub("A", "L", trt)]] == 1,
    mtp(data, trt), # if yes continue with the policy
    data[[trt]]     # otherwise do nothing
  )
}

lmtp_sdr(sim_t4, A, "Y", time_vary = L, mtp = TRUE,
         k = 0, shift = policy, folds = 2)

# Example 2.4
# Using the same data as examples 2.1, 2.2, and 2.3, but now treating the exposure
# as an ordered categorical variable. To account for the exposure being a
# factor we just need to modify the shift function (and the original data)
# so as to respect this.
tmp <- sim_t4
for (i in A) {
  tmp[[i]] <- factor(tmp[[i]], levels = 0:5, ordered = TRUE)
}

policy <- function(data, trt) {
  out <- list()
  a <- data[[trt]]
  for (i in 1:length(a)) {
    if (as.character(a[i]) %in% c("0", "1")) {
      out[[i]] <- as.character(a[i])
    } else {
      out[[i]] <- as.numeric(as.character(a[i])) - 1
    }
  }
  factor(unlist(out), levels = 0:5, ordered = TRUE)
}

lmtp_sdr(tmp, A, "Y", time_vary = L, shift = policy,
         k = 0, folds = 2, mtp = TRUE)

# Example 3.1
# Longitudinal setting, time-varying binary treatment, time-varying covariates
# and baseline covariates with no loss-to-follow-up. Interested in a "traditional"
# causal effect where treatment is set to 1 at all time points for all observations.
if (require("twang")) {
  data("iptwExWide", package = "twang")

  A <- paste0("tx", 1:3)
  W <- c("gender", "age")
  L <- list(c("use0"), c("use1"), c("use2"))
```

```
  lmtp_sdr(
    iptwExWide, A, "outcome", baseline = W, time_vary = L,
    shift = static_binary_on, outcome_type = "continuous",
    mtp = FALSE, folds = 2
  )
}

# Example 4.1
# Longitudinal setting, time-varying continuous treatment, time-varying covariates,
# binary outcome with right censoring. Interested in the mean population outcome under
# the observed exposures in a hypothetical population with no loss-to-follow-up.
head(sim_cens)

A <- c("A1", "A2")
L <- list(c("L1"), c("L2"))
C <- c("C1", "C2")
Y <- "Y"

lmtp_sdr(sim_cens, A, Y, time_vary = L, cens = C, shift = NULL, folds = 2)

# Example 5.1
# Time-to-event analysis with a binary time-invariant exposure. Interested in
# the effect of treatment being given to all observations on the cumulative
# incidence of the outcome.
# For a survival problem, the outcome argument now takes a vector of outcomes
# if an observation experiences the event prior to the end of follow-up, all future
# outcome nodes should be set to 1 (i.e., last observation carried forward).
A <- "trt"
Y <- paste0("Y.", 1:6)
C <- paste0("C.", 0:5)
W <- c("W1", "W2")

lmtp_sdr(sim_point_surv, A, Y, W, cens = C, folds = 2,
         shift = static_binary_on, outcome_type = "survival")
```

`lmtp_sub` *LMTP Substitution Estimator*

Description

G-computation estimator for the effects of traditional causal effects and modified treatment policies for both point treatment and longitudinal data with binary, continuous, or time-to-event outcomes. Supports binary, categorical, and continuous exposures.

Usage

```
lmtp_sub(
  data,
```

```
  trt,
  outcome,
  baseline = NULL,
  time_vary = NULL,
  cens = NULL,
  shift = NULL,
  shifted = NULL,
  k = Inf,
  outcome_type = c("binomial", "continuous", "survival"),
  id = NULL,
  bounds = NULL,
  learners = "SL.glm",
  folds = 10,
  weights = NULL,
  .bound = 1e-05,
  .learners_folds = 10,
  .return_full_fits = FALSE
)
```

Arguments

data	[data.frame] A data.frame in wide format containing all necessary variables for the estimation problem. Must not be a data.table.
trt	[character] A vector containing the column names of treatment variables ordered by time.
outcome	[character] The column name of the outcome variable. In the case of time-to-event analysis, a vector containing the columns names of intermediate outcome variables and the final outcome variable ordered by time. Only numeric values are allowed. If the outcome type is binary, data should be coded as 0 and 1.
baseline	[character] An optional vector containing the column names of baseline covariates to be included for adjustment at every time point.
time_vary	[list] A list the same length as the number of time points of observation with the column names for new time-varying covariates introduced at each time point. The list should be ordered following the time ordering of the model.
cens	[character] An optional vector of column names of censoring indicators the same length as the number of time points of observation. If missingness in the outcome is present or if time-to-event outcome, must be provided.
shift	[closure] A two argument function that specifies how treatment variables should be shifted. See examples for how to specify shift functions for continuous, binary, and categorical exposures.

`shifted` [data.frame]
An optional data frame, the same as in `data`, but modified according to the treatment policy of interest. If specified, `shift` is ignored.

`k` [integer(1)]
An integer specifying how previous time points should be used for estimation at the given time point. Default is `Inf`, all time points.

`outcome_type` [character(1)]
Outcome variable type (i.e., continuous, binomial, survival).

`id` [character(1)]
An optional column name containing cluster level identifiers.

`bounds` [numeric(2)]
An optional, ordered vector of the bounds for a continuous outcomes. If `NULL`, the bounds will be taken as the minimum and maximum of the observed data. Should be left as `NULL` if the outcome type is binary.

`learners` [character]
A vector of `SuperLearner` algorithms for estimation of the outcome regression. Default is `"SL.glm"`, a main effects GLM.

`folds` [integer(1)]
The number of folds to be used for cross-fitting.

`weights` [numeric(nrow(data))]
An optional vector containing sampling weights.

`.bound` [numeric(1)]
Determines that maximum and minimum values (scaled) predictions will be bounded by. The default is 1e-5, bounding predictions by 1e-5 and 0.9999.

`.learners_folds`
[integer(1)]
The number of cross-validation folds for `learners`.

`.return_full_fits`
[logical(1)]
Return full SuperLearner fits? Default is `FALSE`, return only SuperLearner weights.

Value

A list of class `lmtp` containing the following components:

`estimator` The estimator used, in this case "substitution".

`theta` The estimated population LMTP effect.

`standard_error` NA

`low` NA

`high` NA

`shift` The shift function specifying the treatment policy of interest.

`outcome_reg` An n x Tau + 1 matrix of outcome regression predictions. The mean of the first column is used for calculating theta.

`fits_m` A list the same length as `folds`, containing the fits at each time-point for each fold for the outcome regression.

`outcome_type` The outcome variable type.

Examples

```
set.seed(56)
n <- 1000
W <- rnorm(n, 10, 5)
A <- 23 + 5*W + rnorm(n)
Y <- 7.2*A + 3*W + rnorm(n)
ex1_dat <- data.frame(W, A, Y)

# Example 1.1
# Point treatment, continuous exposure, continuous outcome, no loss-to-follow-up
# Interested in the effect of a modified treatment policy where A is decreased by 15
# units only among observations whose observed A was above 80.
# The true value under this intervention is about 513.
policy <- function(data, x) {
  (data[[x]] > 80)*(data[[x]] - 15) + (data[[x]] <= 80)*data[[x]]
}

lmtp_sub(ex1_dat, "A", "Y", "W", shift = policy,
         outcome_type = "continuous", folds = 2)

# Example 2.1
# Longitudinal setting, time-varying continuous exposure bounded by 0,
# time-varying covariates, and a binary outcome with no loss-to-follow-up.
# Interested in the effect of a treatment policy where exposure decreases by
# one unit at every time point if an observations observed exposure is greater
# than or equal to 2. The true value under this intervention is about 0.305.
head(sim_t4)

A <- c("A_1", "A_2", "A_3", "A_4")
L <- list(c("L_1"), c("L_2"), c("L_3"), c("L_4"))

policy <- function(data, trt) {
  a <- data[[trt]]
  (a - 1) * (a - 1 >= 1) + a * (a - 1 < 1)
}

# BONUS: progressr progress bars!
progressr::handlers(global = TRUE)

lmtp_sub(sim_t4, A, "Y", time_vary = L, shift = policy, folds = 2)

# Example 2.2
# The previous example assumed that the outcome (as well as the treatment variables)
# were directly affected by all other nodes in the past. In certain situations,
# domain specific knowledge may suggest otherwise.
# This can be controlled using the k argument.
lmtp_sub(sim_t4, A, "Y", time_vary = L, shift = policy, k = 0, folds = 2)

# Example 2.3
# Using the same data as examples 2.1 and 2.2.
# Now estimating the effect of a dynamic modified treatment policy.
```

```
# creating a dynamic mtp that applies the shift function
# but also depends on history and the current time
policy <- function(data, trt) {
  mtp <- function(data, trt) {
    (data[[trt]] - 1) * (data[[trt]] - 1 >= 1) + data[[trt]] * (data[[trt]] - 1 < 1)
  }

  # if its the first time point, follow the same mtp as before
  if (trt == "A_1") return(mtp(data, trt))

  # otherwise check if the time varying covariate equals 1
  ifelse(
    data[[sub("A", "L", trt)]] == 1,
    mtp(data, trt), # if yes continue with the policy
    data[[trt]]     # otherwise do nothing
  )
}

lmtp_sub(sim_t4, A, "Y", time_vary = L, k = 0, shift = policy, folds = 2)

# Example 2.4
# Using the same data as examples 2.1, 2.2, and 2.3, but now treating the exposure
# as an ordered categorical variable. To account for the exposure being a
# factor we just need to modify the shift function (and the original data)
# so as to respect this.
tmp <- sim_t4
for (i in A) {
  tmp[[i]] <- factor(tmp[[i]], levels = 0:5, ordered = TRUE)
}

policy <- function(data, trt) {
  out <- list()
  a <- data[[trt]]
  for (i in 1:length(a)) {
    if (as.character(a[i]) %in% c("0", "1")) {
      out[[i]] <- as.character(a[i])
    } else {
      out[[i]] <- as.numeric(as.character(a[i])) - 1
    }
  }
  factor(unlist(out), levels = 0:5, ordered = TRUE)
}

lmtp_sub(tmp, A, "Y", time_vary = L, shift = policy, k = 0, folds = 2)

# Example 3.1
# Longitudinal setting, time-varying binary treatment, time-varying covariates
# and baseline covariates with no loss-to-follow-up. Interested in a "traditional"
# causal effect where treatment is set to 1 at all time points for all observations.
if (require("twang")) {
  data("iptwExWide", package = "twang")
```

```
  A <- paste0("tx", 1:3)
  W <- c("gender", "age")
  L <- list(c("use0"), c("use1"), c("use2"))

  lmtp_sub(iptwExWide, A, "outcome", baseline = W, time_vary = L,
           shift = static_binary_on, outcome_type = "continuous")
}

# Example 4.1
# Longitudinal setting, time-varying continuous treatment, time-varying covariates,
# binary outcome with right censoring. Interested in the mean population outcome under
# the observed exposures in a hypothetical population with no loss-to-follow-up.
head(sim_cens)

A <- c("A1", "A2")
L <- list(c("L1"), c("L2"))
C <- c("C1", "C2")
Y <- "Y"

lmtp_sub(sim_cens, A, Y, time_vary = L, cens = C, shift = NULL, folds = 2)

# Example 5.1
# Time-to-event analysis with a binary time-invariant exposure. Interested in
# the effect of treatment being given to all observations on the cumulative
# incidence of the outcome.
# For a survival problem, the outcome argument now takes a vector of outcomes
# if an observation experiences the event prior to the end of follow-up, all future
# outcome nodes should be set to 1 (i.e., last observation carried forward).
A <- "trt"
Y <- paste0("Y.", 1:6)
C <- paste0("C.", 0:5)
W <- c("W1", "W2")

lmtp_sub(sim_point_surv, A, Y, W, cens = C, folds = 2,
         shift = static_binary_on, outcome_type = "survival")
```

`lmtp_tmle` *LMTP Targeted Maximum Likelihood Estimator*

Description

Targeted maximum likelihood estimator for the effects of traditional causal effects and modified treatment policies for both point treatment and longitudinal data with binary, continuous, or time-to-event outcomes. Supports binary, categorical, and continuous exposures.

Usage

```
lmtp_tmle(
  data,
```

```
  trt,
  outcome,
  baseline = NULL,
  time_vary = NULL,
  cens = NULL,
  shift = NULL,
  shifted = NULL,
  k = Inf,
  mtp = FALSE,
  outcome_type = c("binomial", "continuous", "survival"),
  id = NULL,
  bounds = NULL,
  learners_outcome = "SL.glm",
  learners_trt = "SL.glm",
  folds = 10,
  weights = NULL,
  .bound = 1e-05,
  .trim = 0.999,
  .learners_outcome_folds = 10,
  .learners_trt_folds = 10,
  .return_full_fits = FALSE,
  ...
)
```

Arguments

data	[data.frame] A data.frame in wide format containing all necessary variables for the estimation problem. Must not be a data.table.
trt	[character] A vector containing the column names of treatment variables ordered by time.
outcome	[character] The column name of the outcome variable. In the case of time-to-event analysis, a vector containing the columns names of intermediate outcome variables and the final outcome variable ordered by time. Only numeric values are allowed. If the outcome type is binary, data should be coded as 0 and 1.
baseline	[character] An optional vector containing the column names of baseline covariates to be included for adjustment at every time point.
time_vary	[list] A list the same length as the number of time points of observation with the column names for new time-varying covariates introduced at each time point. The list should be ordered following the time ordering of the model.
cens	[character] An optional vector of column names of censoring indicators the same length as the number of time points of observation. If missingness in the outcome is present or if time-to-event outcome, must be provided.

`shift` [closure]
A two argument function that specifies how treatment variables should be shifted. See examples for how to specify shift functions for continuous, binary, and categorical exposures.

`shifted` [data.frame]
An optional data frame, the same as in `data`, but modified according to the treatment policy of interest. If specified, `shift` is ignored.

`k` [integer(1)]
An integer specifying how previous time points should be used for estimation at the given time point. Default is `Inf`, all time points.

`mtp` [logical(1)]
Is the intervention of interest a modified treatment policy? Default is `FALSE`. If treatment variables are continuous this should be `TRUE`.

`outcome_type` [character(1)]
Outcome variable type (i.e., continuous, binomial, survival).

`id` [character(1)]
An optional column name containing cluster level identifiers.

`bounds` [numeric(2)]
An optional, ordered vector of the bounds for a continuous outcomes. If `NULL`, the bounds will be taken as the minimum and maximum of the observed data. Should be left as `NULL` if the outcome type is binary.

`learners_outcome` [character]
A vector of `SuperLearner` algorithms for estimation of the outcome regression. Default is `"SL.glm"`, a main effects GLM.

`learners_trt` [character]
A vector of `SuperLearner` algorithms for estimation of the exposure mechanism. Default is `"SL.glm"`, a main effects GLM. **Only include candidate learners capable of binary classification**.

`folds` [integer(1)]
The number of folds to be used for cross-fitting.

`weights` [numeric(nrow(data))]
An optional vector containing sampling weights.

`.bound` [numeric(1)]
Determines that maximum and minimum values (scaled) predictions will be bounded by. The default is 1e-5, bounding predictions by 1e-5 and 0.9999.

`.trim` [numeric(1)]
Determines the amount the density ratios should be trimmed. The default is 0.999, trimming the density ratios greater than the 0.999 percentile to the 0.999 percentile. A value of 1 indicates no trimming.

`.learners_outcome_folds` [integer(1)]
The number of cross-validation folds for `learners_outcome`.

`.learners_trt_folds` [integer(1)]
The number of cross-validation folds for `learners_trt`.

`.return_full_fits`
: [logical(1)]
Return full SuperLearner fits? Default is FALSE, return only SuperLearner weights.

`...`
: Extra arguments. Exists for backwards compatibility.

Details

Should `mtp = TRUE`**?:**

A modified treatment policy (MTP) is an intervention that depends on the natural value of the exposure (the value that the treatment would have taken under no intervention). This differs from other causal effects, such as the average treatment effect (ATE), where an exposure would be increased (or decreased) deterministically. **If your intervention of interest adds, subtracts, or multiplies the observed treatment values by some amount, use** `mtp = TRUE`.

Value

A list of class `lmtp` containing the following components:

`estimator`	The estimator used, in this case "TMLE".
`theta`	The estimated population LMTP effect.
`standard_error`	The estimated standard error of the LMTP effect.
`low`	Lower bound of the 95% confidence interval of the LMTP effect.
`high`	Upper bound of the 95% confidence interval of the LMTP effect.
`eif`	The estimated, un-centered, influence function of the estimate.
`shift`	The shift function specifying the treatment policy of interest.
`outcome_reg`	An n x Tau + 1 matrix of outcome regression predictions. The mean of the first column is used for calculating theta.
`density_ratios`	An n x Tau matrix of the estimated, non-cumulative, density ratios.
`fits_m`	A list the same length as `folds`, containing the fits at each time-point for each fold for the outcome regression.
`fits_r`	A list the same length as `folds`, containing the fits at each time-point for each fold of density ratio estimation.
`outcome_type`	The outcome variable type.

Examples

```
set.seed(56)
n <- 1000
W <- rnorm(n, 10, 5)
A <- 23 + 5*W + rnorm(n)
Y <- 7.2*A + 3*W + rnorm(n)
ex1_dat <- data.frame(W, A, Y)

# Example 1.1
# Point treatment, continuous exposure, continuous outcome, no loss-to-follow-up
# Interested in the effect of a modified treatment policy where A is decreased by 15
```

```
# units only among observations whose observed A was above 80.
# The true value under this intervention is about 513.
policy <- function(data, x) {
  (data[[x]] > 80)*(data[[x]] - 15) + (data[[x]] <= 80)*data[[x]]
}

lmtp_tmle(ex1_dat, "A", "Y", "W", shift = policy,
          outcome_type = "continuous", folds = 2, mtp = TRUE)

# Example 2.1
# Longitudinal setting, time-varying continuous exposure bounded by 0,
# time-varying covariates, and a binary outcome with no loss-to-follow-up.
# Interested in the effect of a treatment policy where exposure decreases by
# one unit at every time point if an observations observed exposure is greater
# than or equal to 2. The true value under this intervention is about 0.305.
head(sim_t4)

A <- c("A_1", "A_2", "A_3", "A_4")
L <- list(c("L_1"), c("L_2"), c("L_3"), c("L_4"))
policy <- function(data, trt) {
  a <- data[[trt]]
  (a - 1) * (a - 1 >= 1) + a * (a - 1 < 1)
}

# BONUS: progressr progress bars!
progressr::handlers(global = TRUE)

lmtp_tmle(sim_t4, A, "Y", time_vary = L, shift = policy,
          folds = 2, mtp = TRUE)

# Example 2.2
# The previous example assumed that the outcome (as well as the treatment variables)
# were directly affected by all other nodes in the past. In certain situations,
# domain specific knowledge may suggest otherwise.
# This can be controlled using the k argument.
lmtp_tmle(sim_t4, A, "Y", time_vary = L, shift = policy,
          k = 0, folds = 2, mtp = TRUE)

# Example 2.3
# Using the same data as examples 2.1 and 2.2.
# Now estimating the effect of a dynamic modified treatment policy.
# creating a dynamic mtp that applies the shift function
# but also depends on history and the current time
policy <- function(data, trt) {
  mtp <- function(data, trt) {
    (data[[trt]] - 1) * (data[[trt]] - 1 >= 1) + data[[trt]] * (data[[trt]] - 1 < 1)
  }

  # if its the first time point, follow the same mtp as before
  if (trt == "A_1") return(mtp(data, trt))

  # otherwise check if the time varying covariate equals 1
  ifelse(
```

```
    data[[sub("A", "L", trt)]] == 1,
    mtp(data, trt), # if yes continue with the policy
    data[[trt]]     # otherwise do nothing
  )
}

lmtp_tmle(sim_t4, A, "Y", time_vary = L, mtp = TRUE,
          k = 0, shift = policy, folds = 2)

# Example 2.4
# Using the same data as examples 2.1, 2.2, and 2.3, but now treating the exposure
# as an ordered categorical variable. To account for the exposure being a
# factor we just need to modify the shift function (and the original data)
# so as to respect this.
tmp <- sim_t4
for (i in A) {
  tmp[[i]] <- factor(tmp[[i]], levels = 0:5, ordered = TRUE)
}

policy <- function(data, trt) {
  out <- list()
  a <- data[[trt]]
  for (i in 1:length(a)) {
    if (as.character(a[i]) %in% c("0", "1")) {
      out[[i]] <- as.character(a[i])
    } else {
      out[[i]] <- as.numeric(as.character(a[i])) - 1
    }
  }
  factor(unlist(out), levels = 0:5, ordered = TRUE)
}

lmtp_tmle(tmp, A, "Y", time_vary = L, shift = policy,
          k = 0, folds = 2, mtp = TRUE)

# Example 3.1
# Longitudinal setting, time-varying binary treatment, time-varying covariates
# and baseline covariates with no loss-to-follow-up. Interested in a "traditional"
# causal effect where treatment is set to 1 at all time points for all observations.
if (require("twang")) {
  data("iptwExWide", package = "twang")

  A <- paste0("tx", 1:3)
  W <- c("gender", "age")
  L <- list(c("use0"), c("use1"), c("use2"))

  lmtp_tmle(iptwExWide, A, "outcome", baseline = W, time_vary = L,
            shift = static_binary_on, outcome_type = "continuous",
            mtp = FALSE, folds = 2)
}

# Example 4.1
# Longitudinal setting, time-varying continuous treatment, time-varying covariates,
```

```
# binary outcome with right censoring. Interested in the mean population outcome under
# the observed exposures in a hypothetical population with no loss-to-follow-up.
head(sim_cens)

A <- c("A1", "A2")
L <- list(c("L1"), c("L2"))
C <- c("C1", "C2")
Y <- "Y"

lmtp_tmle(sim_cens, A, Y, time_vary = L, cens = C, shift = NULL, folds = 2)

# Example 5.1
# Time-to-event analysis with a binary time-invariant exposure. Interested in
# the effect of treatment being given to all observations on the cumulative
# incidence of the outcome.
# For a survival problem, the outcome argument now takes a vector of outcomes
# if an observation experiences the event prior to the end of follow-up, all future
# outcome nodes should be set to 1 (i.e., last observation carried forward).
A <- "trt"
Y <- paste0("Y.", 1:6)
C <- paste0("C.", 0:5)
W <- c("W1", "W2")

lmtp_tmle(sim_point_surv, A, Y, W, cens = C, folds = 2,
          shift = static_binary_on, outcome_type = "survival")
```

`sim_cens` *Simulated Longitudinal Data With Censoring*

Description

A dataset with a binary outcome, two time varying treatment nodes, two time varying covariates, and two censoring indicators.

Usage

```
sim_cens
```

Format

A data frame with 1000 rows and 10 variables:

L1 Time varying covariate time 1

A1 Treatment node at time 1, effected by L_1

C1 Censoring indicator that the observation is observed after time 1

L2 Time varying covariate at time 2, effected by L_1 and A_1

A2 Treatment node at time 2, effected by L_2 and A_1

C2 Censoring indicator that the observation is observed after time 2

Y Binary outcome at time 3, effected by L_2 and A_2

`sim_point_surv` *Simulated Point-treatment Survival Data*

Description

A dataset with a time-to-event outcome, two baseline nodes, a binary point treatment, six past-time outcome nodes, and six censoring indicators.

Usage

```
sim_point_surv
```

Format

A data frame with 2000 rows and 16 variables:

W1 Binary baseline variable.

W2 Categorical baseline variable.

trt Binary treatment variable.

C.0 Censoring indicator that the observation is observed future time points.

Y.1 Outcome node at time 1.

C.1 Censoring indicator that the observation is observed future time points.

Y.2 Outcome node at time 2.

C.2 Censoring indicator that the observation is observed future time points.

Y.3 Outcome node at time 3.

C.3 Censoring indicator that the observation is observed future time points.

Y.4 Outcome node at time 4.

C.4 Censoring indicator that the observation is observed future time points.

Y.5 Outcome node at time 5.

C.5 Censoring indicator that the observation is observed future time points.

Y.6 Final outcome node.

sim_t4 *Simulated Longitudinal Data*

Description

A dataset with a binary outcome, four time varying treatment nodes, and four time varying covariates.

Usage

```
sim_t4
```

Format

A data frame with 5000 rows and 10 variables:

ID observation ID

L_1 Time varying covariate time 1

A_1 Treatment node at time 1, effected by L_1

L_2 Time varying covariate time 1, effected by L_1 and A_1

A_2 Treatment node at time 2, effected by L_2 and A_1

L_3 Time varying covariate time 1, effected by L_2 and A_2

A_3 Treatment node at time 3, effected by L_3 and A_2

L_4 Time varying covariate time 1, effected by L_3 and A_3

A_4 Treatment node at time 3, effected by L_4 and A_3

Y Binary outcome at time 5, effected by L_4 and A_4

sim_timevary_surv *Simulated Time-varying Survival Data*

Description

A dataset with a time-to-event outcome, one baseline nodes, two time-varying covariates, a binary time-varying treatment, two outcome nodes, and two censoring indicators. Data-generating mechanism taken from Lendle, Schwab, Petersen, and van der Laan (https://www.jstatsoft.org/article/view/v081i01).

Usage

```
sim_timevary_surv
```

Format

A data frame with 500 rows and 11 variables:

L0.a Continuous baseline variable.

L0.b Time varying covariate at baseline.

L0.c Time varying covariate at baseline.

A0 Treatment variable at baseline

C0 Censoring indicator that the observation is observed future time points.

L1.a Time varying covariate at time 1.

L1.b Time varying covariate at time 1.

Y1 Outcome node at time 1.

A1 Treatment variable at time 1.

C1 Censoring indicator that the observation is observed future time points.

Y2 Final outcome node.

`static_binary_off` *Turn All Treatment Nodes Off*

Description

A pre-packaged shift function for use with provided estimators when the exposure is binary. Used to estimate the population intervention effect when all treatment variables are set to 0.

Usage

```
static_binary_off(data, trt)
```

Arguments

`data`	A dataframe containing the treatment variables.
`trt`	The name of the current treatment variable.

Value

A dataframe with all treatment nodes set to 0.

See Also

`lmtp_tmle()`, `lmtp_sdr()`, `lmtp_sub()`, `lmtp_ipw()`

Examples

```
data("iptwExWide", package = "twang")
a <- paste0("tx", 1:3)
baseline <- c("gender", "age")
tv <- list(c("use0"), c("use1"), c("use2"))
lmtp_sdr(iptwExWide, a, "outcome", baseline = baseline, time_vary = tv,
         shift = static_binary_off, outcome_type = "continuous", folds = 2)
```

---

static_binary_on *Turn All Treatment Nodes On*

Description

A pre-packaged shift function for use with provided estimators when the exposure is binary. Used to estimate the population intervention effect when all treatment variables are set to 1.

Usage

```
static_binary_on(data, trt)
```

Arguments

data	A dataframe containing the treatment variables.
trt	The name of the current treatment variable.

Value

A dataframe with all treatment nodes set to 1.

See Also

lmtp_tmle(), lmtp_sdr(), lmtp_sub(), lmtp_ipw()

Examples

```
data("iptwExWide", package = "twang")
a <- paste0("tx", 1:3)
baseline <- c("gender", "age")
tv <- list(c("use0"), c("use1"), c("use2"))
lmtp_sdr(iptwExWide, a, "outcome", baseline = baseline, time_vary = tv,
         shift = static_binary_on, outcome_type = "continuous", folds = 2)
```

`tidy.lmtp` *Tidy a(n) lmtp object*

Description

Tidy a(n) lmtp object

Usage

```
## S3 method for class 'lmtp'
tidy(x, ...)
```

Arguments

`x`	A `lmtp` object produced by a call to `lmtp_tmle()`, `lmtp_sdr()`, `lmtp_sub()`, or `lmtp_ipw()`.
`...`	Unused, included for generic consistency only.

Examples

```
a <- c("A1", "A2")
nodes <- list(c("L1"), c("L2"))
cens <- c("C1", "C2")
y <- "Y"
fit <- lmtp_tmle(sim_cens, a, y, time_vary = nodes, cens = cens, shift = NULL, folds = 2)
tidy(fit)
```

Index