Package 'lmw'

February 8, 2024

Type Package

Title Linear Model Weights

Version 0.0.2

Description Computes the implied weights of linear regression models for estimating average causal effects and provides diagnostics based on these weights. These diagnostics rely on the analyses in Chattopadhyay and Zubizarreta (2023) <doi:10.1093/biomet/asac058> where several regression estimators are represented as weighting estimators, in connection to inverse probability weighting. 'lmw' provides tools to diagnose representativeness, balance, extrapolation, and influence for these models, clarifying the target population of inference. Tools are also available to simplify estimating treatment effects for specific target populations of interest.

License GPL (>= 2)

Encoding UTF-8

URL https://github.com/ngreifer/lmw

BugReports https://github.com/ngreifer/lmw/issues

Depends R (>= 3.5.0)

Imports chk (>= 0.9.1), sandwich (>= 3.0-2), backports (>= 1.4.1)

Suggests MatchIt (>= 4.3.2), WeightIt (>= 0.14.2), marginaleffects (>= 0.17.0), PSweight (>= 1.1.8), estimatr, lmtest, ivreg, mlogit, testthat (>= 3.0.0)

RoxygenNote 7.3.0

LazyData true

Config/testthat/edition 3

NeedsCompilation no

Author Ambarish Chattopadhyay [aut] (<https://orcid.org/0000-0002-1502-0974>), Noah Greifer [aut, cre] (<https://orcid.org/0000-0003-3067-7154>), Jose Zubizarreta [aut] (<https://orcid.org/0000-0002-0322-147X>)

Maintainer Noah Greifer <ngreifer@iq.harvard.edu>

Repository CRAN

Date/Publication 2024-02-08 16:30:03 UTC

R topics documented:

---

influence.lmw	*Regression Diagnostics for* lmw *and* lmw_est *objects*

Description

influence() produces influence measures for lmw objects that can be used as regression diagnostics to identify influential cases. These functions produce similar outputs to lm.influence() but also include the sample influence curve (SIC) values, which combine information about the hat values, residuals, and implied regression weights.

Usage

```
## S3 method for class 'lmw'
influence(model, outcome, data = NULL, ...)

## S3 method for class 'lmw_est'
influence(model, ...)
```

Arguments

model	an lmw or lmw_est object; the output of a call to lmw() or lmw_est().
outcome	the name of the outcome variable. Can be supplied as a string containing the name of the outcome variable or as the outcome variable itself. If not supplied, the outcome variable in the formula supplied to lmw(), if any, will be used.
data	an optional data frame containing the outcome variable named in outcome.
...	ignored.

Details

influence() computes the hat values, (weighted) residuals, and sample influence curve (SIC) values for each unit, which can be used as regression diagnostics to assess influence. The weighted residuals are weighted by the sampling weights (if supplied), not the implied regression weights. The SIC values are computed as SIC = (N−1) * w * r / (1 − h), where N is the sample size, w are the units' implied regression weights, r are the (weighted) residuals, and h are the hat values. SIC values are scaled to have a maximum of 1. Higher values indicate greater relative influence.

Value

A list with the following components:

hat	a vector containing the diagonal of the hat matrix.
wt.res	a vector of (weighted) residuals.
sic	a vector containing the scaled SIC values.

Note

influence.lmw() uses non-standard evaluation to interpret its outcome argument. For programmers who wish to use influence.lmw() inside other functions, an effective way to pass the name of an arbitrary outcome (e.g., y passed as a string) is to use do.call(), for example:

```
fun <- function(m, y, d) {
do.call("influence", list(m, y, d)) }
```

When using influence.lmw() inside lapply() or purrr::map to loop over outcomes, this syntax must be used as well.

See Also

plot.lmw() for plotting the SIC values; lm.influence() for influence measures for lm objects, which do not include SIC values; hatvalues() for hat values for lm objects (note that lmw_est objects also have a hatvalues() method).

Examples

```
data("lalonde")

# URI regression for ATT
lmw.out1 <- lmw(~ treat + age + education + race + married +
                    nodegree + re74 + re75,
               data = lalonde, estimand = "ATT",
               method = "URI", treat = "treat")

# Influence for re78 outcome
infl <- influence(lmw.out1, outcome = "re78")
str(infl)

# Can also be used after lmw_est():
```

```
lmw.est1 <- lmw_est(lmw.out1, outcome = "re78")
all.equal(infl,
          influence(lmw.est1))
```

---

lalonde	*Data from National Supported Work Demonstration and PSID, as an-alyzed by Dehejia and Wahba (1999).*

Description

This is a subsample of the data from the treated group in the National Supported Work Demonstration (NSW) and the comparison sample from the Population Survey of Income Dynamics (PSID). This data was previously analyzed extensively by Lalonde (1986) and Dehejia and Wahba (1999). The original dataset is available at https://users.nber.org/~rdehejia/nswdata2.html.

Usage

```
lalonde
```

Format

A data frame with 2675 observations (185 treated, 2490 control). There are 9 variables measured for each individual. In addition, two constructed variables are included: `treat_multi`, which splits the original control group into two, and `Ins`, which is a constructed instrumental variable.

- "treat" is the treatment assignment (1=treated, 0=control).
- "age" is age in years.
- "education" is education in number of years of schooling.
- "race" is the individual's race/ethnicity, (Black, Hispanic, or White).
- "married" is an indicator for married (1=married, 0=not married).
- "nodegree" is an indicator for whether the individual lacks a high school degree (i.e., has fewer than 12 years of schooling; 1=no degree, 0=degree).
- "re74" is income in 1974, in U.S. dollars.
- "re75" is income in 1975, in U.S. dollars.
- "re78" is income in 1978, in U.S. dollars.
- "treat_multi" is a constructed version of "treat" that splits the control group into to levels (1=treated, 2=control group A, 3=control group B).
- "Ins" is a binary instrumental variable.

"treat" is the treatment variable, "re78" is the outcome, and the others are pre-treatment covariates. Note that in the original data, "race" is instead coded as two dummy variables, "black" and "hispan".

Details

The data corresponds to the NSW treated sample and the PSID control sample with 1974 earnings included. This specific dataset is different from the one in the **MatchIt** and **cobalt** packages, which is a subset of this dataset.

References

Lalonde, R. (1986). Evaluating the econometric evaluations of training programs with experimental data. American Economic Review 76: 604-620.

Dehejia, R.H. and Wahba, S. (1999). Causal Effects in Nonexperimental Studies: Re-Evaluating the Evaluation of Training Programs. Journal of the American Statistical Association 94: 1053-1062.

---

lmw   *Compute linear regression-implied weights*

Description

Computes the weights implied by a linear outcome regression model that would estimate a weighted difference in outcome means equal to the covariate-adjusted treatment effect resulting from the supplied regression model.

Usage

```
lmw(
  formula,
  data = NULL,
  estimand = "ATE",
  method = "URI",
  treat = NULL,
  base.weights = NULL,
  s.weights = NULL,
  dr.method = "WLS",
  obj = NULL,
  fixef = NULL,
  target = NULL,
  target.weights = NULL,
  contrast = NULL,
  focal = NULL
)
```

Arguments

formula   a one-sided formula with the treatment and covariates on the right-hand side corresponding to the outcome regression model to be fit. The outcome variable is not involved in computing the weights and does not need to be specified. See Details for how this formula is interpreted in light of other options.

data	a data frame containing the variables named in `formula` and `treat`.
estimand	the estimand of interest, which determines how covariates are centered. Should be one of `"ATE"` for the average treatment effect, `"ATT"` for the average treatment effect in the treated, `"ATC"` for the average treatment effect in the control, or `"CATE"` for the conditional average treatment effect. When `estimand = "CATE"`, an argument to `target` must be supplied. This argument also affects what [`summary.lmw()`](#) considers to be the target population. Default is `"ATE"` unless `obj` is specified, in which case it takes its value from the supplied object.
method	the method used to estimate the weights; either `"URI"` (the default) for uni-regression imputation weights, where a single model is fit to the whole dataset, or `"MRI"` for multi-regression imputation, where the model is fit separately in the treatment groups. This affects the interpretation of `formula`. See Details.
treat	the name of the treatment variable in `data`. If unspecified, the first variable present in `formula` will be taken as the treatment variable with a message. See Details.
base.weights	a vector of base weights. See Details. If omitted and `obj` is specified, the weights from the supplied object will be used. Can be supplied as a numeric vector, a string containing the name of the variable in `data` containing the base weights, or the unquoted name of the variable in `data` containing the base weights.
s.weights	a vector of sampling weights. See Details. If omitted and `obj` is specified, the sampling weights from the supplied object will be used. Can be supplied as a numeric vector, a string containing the name of the variable in `data` containing the sampling weights, or the unquoted name of the variable in `data` containing the sampling weights.
dr.method	the method used to incorporate the `base.weights` into a doubly-robust estimator. Can be one of `"WLS"` for weighted least squares (`"IPWRA"` is an allowable alias) or `"AIPW"` for augmented inverse probability weighting. Ignored when `base.weights` is NULL.
obj	a `matchit` or `weightit` object corresponding to the matched or weighted sample in which the implied outcome regression would take place. See Details.
fixef	optional; a string or one-sided formula containing the name of the fixed effects variable in `data`. See Details. Cannot be used with `dr.method = "AIPW"`.
target	a list or data frame containing the target values for each covariate included in `formula`. Ignored with a warning when `estimand` is not `"CATE"`. See Details.
target.weights	a vector of sampling weights to be applied to `target` when supplied as a data frame. Ignored with a warning when `estimand` is not `"CATE"`. See Details.
contrast	for multi-category treatments with `method = "URI"`, a vector containing the names or indices of the two treatment levels to be contrasted (since in this case the weights depend on the specific contrast). See Details.
focal	the level of the treatment variable to be considered "focal" (i.e., the "treated" level when `estimand = "ATT"` or the control level when `estimand = "ATC"`). Ignored when `estimand` is `"ATE"` or `"CATE"`. Otherwise, if unspecified, the second value of `contrast` will be considered focal when `estimand = "ATT"` and the first value of `contrast` will be considered focal when `estimand = "ATC"`. For binary treatments, this generally does not need to be supplied. See Details.

Details

formula is interpreted differently depending on whether method is "URI" or "MRI". When method = "URI", the formula is taken literally as the right-hand side of the outcome model formula. The only difference is that the covariates will be centered based on the argument to estimand (see below). When method = "MRI", all references to the treatment are removed (i.e., covariate interactions with treatment become covariate main effects if not already present), and the new formula is taken as the right-hand side of the model formula fit within each treatment group. This is equivalent to allowing all covariates to have both main effects and interactions with treatment after centering the covariates based on the argument to estimand. Allowing the treatment to interact with all covariates with method = "URI" is equivalent to specifying method = "MRI", and, for binary treatments, the returned weights will be the same when fixef = NULL.

When any treatment-by-covariate interactions are present in formula or when method = "MRI", covariates are centered at specific values to ensure the resulting weights correspond to the desired estimand as supplied to the estimand argument. For the ATE, the covariates are centered at their means in the full sample. For the ATT and ATC, the covariates are centered at their means in the treatment or control group (i.e., the focal group), respectively. For the CATE, the covariates are centered according to the argument supplied to target (see below). Note that when covariate-by-covariate interactions are present, they will be centered after computing the interaction rather than the interaction being computed on the centered covariates unless estimand = "CATE", in which case the covariates will be centered at the values specified in target prior to involvement in interactions.

Estimating a CATE:

When estimand = "CATE", target can be supplied either as a single target profile (i.e., a list or a data frame with one row) or as a target dataset, potentially with its own sampling weights, which are supplied to target.weights. The variables included in target must correspond to all the named *covariates* in formula; for example, if formula = ~ X1 + log(X1) + X2 + X1:X2, values in target must be given for X1 and X2, but not log(X1) or X1:X2. To choose a target profile value for a factor corresponding to a proportion (e.g., a target value of .5 for a variable like sex indicating a target population with a 50-50 sex split), the factor variable must be split into a numeric variable beforehand, e.g., using model.matrix() or cobalt::splitfactor(). target values cannot be given to variables specified using $, [[]], or [] (e.g., data$X1), so an error will be thrown if they are used in formula. When a target dataset is supplied, covariates will be centered at their means in the (target.weights-weighted) target dataset.

Base weights and sampling weights:

Base weights (base.weights) and sampling weights (s.weights) are similar in that they both involve combining weights with an outcome regression model. However, they differ in a few ways. Sampling weights are primarily used to adjust the target population; when the outcome model is fit, it is fit using weighted least squares, and when target balance is assessed, it is assessed using the sampling weighted population as the target population. Centering of covariates in the outcome model is done using the sampling weighted covariate means. Base weights are primarily used to offer a second level of balancing beyond the implied regression weights; they can be incorporated into the effect estimate either using weighted least squares or using the augmented inverse probability weighting (AIPW) estimator. Base weights do not change the target population, so when target balance is assessed, it is assessed using the unweighted population as the target population.

Some forms of weights both change the target population and provide an extra layer of balancing, like propensity score weights that target estimands other than the ATT, ATC, or ATE (e.g., overlap weights), or matching weights where the target population is defined by the matching

(e.g., matching with a caliper, cardinality matching, or coarsened exact matching). Because these weights change the target population, they should be supplied to `s.weights` to ensure covariates are appropriately centered. When there are no treatment-by-covariate interactions and `method = "URI"`, whether weights are supplied to `base.weights` or `s.weights` will not matter for the estimation of the weights but will affect the target population in balance assessment.

When both `base.weights` and `s.weights` are supplied, e.g., when the base weights are the result of a propensity score model fit with sampling weights, it is assumed the base weights do not incorporate the sampling weights; that is, it is assumed that to estimate a treatment effect *without* regression adjustment, the base weights and the sampling weights would have to be multiplied together. This is true, for example, for the weights in a `matchit` or `weightit` object (see below) but not for weights in the output of `MatchIt::match.data()` unless called with `include.s.weights = FALSE` or weights resulting from `CBPS::CBPS()`.

Regression after using MatchIt or WeightIt:

Regression weights can be computed in a matched or weighted sample by supplying a `matchit` or `weightit` object (from **MatchIt** or **WeightIt**, respectively) to the `obj` argument of `lmw()`. The estimand, focal group (if any), base weights, and sampling weights (if any) will be taken from the supplied object and used in the calculation of the implied regression weights, unless these have been supplied separately to `lmw()`. The `weights` component of the supplied object containing the matching or balancing weights will be passed to `base.weights` and the `s.weights` component will be passed to `s.weights`. Arguments supplied to `lmw()` will take precedence over the corresponding components in the `obj` object.

Multi-category treatments:

There are a few differences when the treatment has multiple (i.e., more than 2) categories. If `estimand` is `"ATT"` or `"ATC"`, an argument should be supplied to `focal` identifying which group is the treated or control (i.e., "focal") group, respectively.

The key difference, though, is when `method = "URI"`, because in this case the contrast between each pair of treatment groups has its own weights and its own implied target population. Because `lmw()` only produces one set of weights, an argument must be supplied to `contrast` identifying which groups are to be used as the contrast for computing the weights. In addition, to compute the treatment effect corresponding to the chosen contrast as a weighted difference in outcome means, the difference must be taken between the weighted mean of the non-reference group and the weighted mean of *all other groups combined*, rather than simply the weighted mean of the reference group.

The implication of this is that contrast statistics computed in the weighted sample involve all units, even those not in the contrasted groups, whereas statistics computed in the unweighted sample only involve units in the contrasted groups. See `summary.lmw()` for more information on assessing balance using the regression weights for multi-category treatments. Given these complications, it is generally best to use `method = "MRI"` with multi-category treatments.

Fixed effects:

A fixed effects variable can be supplied to the `fixef` argument. This is equivalent to adding the fixed effects variable as a predictor that does not interact with the treatment or any other covariate. The difference is that computation is much faster when the fixed effect has many levels because demeaning is used rather than including the fixed effect variable as a collection of dummy variables. When using URI, the weights will be the same regardless of whether the fixed effect variable is included as a covariate or supplied to `fixef`; when using MRI, results will differ

because the fixed effect variable does not interact with treatment. The fixed effects variable will not appear in the summary.lmw() output (but can be added using addlvariables argument) or in the model output of lmw_est() or summary.lmw_est(). Because it does not interact with the treatment, the distribution of the fixed effect variable may not correspond to the target population, so caution should be used if it is expected the treatment effect varies across levels of this variable (in which case it should be included as a predictor). Currently only one fixed effect variable is allowed.

Value

An lmw object, which contains the following components:

treat	the treatment variable, given as a factor.
weights	the computed implied regression weights.
covs	a data frame containing the covariates included the model formula.
estimand	the requested estimand.
method	the method used to estimate the weights ("URI" or "MRI").
base.weights	the weights supplied to base.weights.
s.weights	the weights supplied to s.weights.
dr.method	when base.weights are supplied, the method for computing the doubly-robust weights.
call	the original call to lmw().
fixef	the fixed effects variable if supplied to fixef.
formula	the model formula.
target	the supplied target profile or dataset when estimand = "CATE", after some initial processing. The "target.weights" attribute contains the target.weights if supplied.
contrast	the contrasted treatment groups.
focal	the focal treatment level when estimand is "ATT" or "ATC".

References

Chattopadhyay, A., & Zubizarreta, J. R. (2023). On the implied weights of linear regression for causal inference. *Biometrika*, 110(3), 615–629. doi:10.1093/biomet/asac058

See Also

summary.lmw() for summarizing balance and representativeness; plot.lmw() for plotting features of the weights; lmw_est() for estimating treatment effects from lmw objects; influence.lmw() for influence measures; lm() for fitting standard regression models.

Examples

```
data("lalonde")

# URI regression for ATT
lmw.out1 <- lmw(~ treat + age + education + race + married +
                  nodegree + re74 + re75, data = lalonde,
              estimand = "ATT", method = "URI",
              treat = "treat")
lmw.out1
summary(lmw.out1)

# MRI regression for ATT
lmw.out2 <- lmw(~ treat + age + education + race + married +
                  nodegree + re74 + re75, data = lalonde,
              estimand = "ATT", method = "MRI",
              treat = "treat")
lmw.out2
summary(lmw.out2)


# MRI regression for ATT after propensity score matching
m.out <- MatchIt::matchit(treat ~ age + education + race +
                            married + nodegree + re74 + re75,
                        data = lalonde, method = "nearest",
                        estimand = "ATT")
lmw.out3 <- lmw(~ treat + age + education + race + married +
                  nodegree + re74 + re75, data = lalonde,
              method = "MRI", treat = "treat", obj = m.out)
lmw.out3
summary(lmw.out3)

# MRI regression for CATE with given target profile
target.prof <- list(age = 25, education = 11, race = "black",
                    married = 0, nodegree = 1, re74 = 0,
                    re75 = 0)
lmw.out4 <- lmw(~ treat + age + education + race + married +
                  nodegree + re74 + re75, data = lalonde,
              estimand = "CATE", method = "MRI",
              treat = "treat", target = target.prof)
lmw.out4
summary(lmw.out4)

# MRI regression for CATE with given target dataset (in
# this case, will give the same as with estimand = "ATT")
target.data <- subset(lalonde, treat == 1)
lmw.out4 <- lmw(~ treat + age + education + race + married +
                  nodegree + re74 + re75, data = lalonde,
              estimand = "CATE", method = "MRI",
              treat = "treat", target = target.data)
lmw.out4
summary(lmw.out4)
```

```
# URI regression with fixed effects for 'race'
lmw.out5 <- lmw(~ treat + age + education + married +
                  nodegree + re74 + re75, data = lalonde,
               method = "URI", treat = "treat",
               fixef = ~race)
lmw.out5

# Produces the same weights as when included as a covariate
all.equal(lmw.out1$weights, lmw.out5$weights)

# MRI for a multi-category treatment, ATT with 1 as the focal
# group
lmw.out6 <- lmw(~ treat_multi + age + education + race + married +
                  nodegree + re74 + re75, data = lalonde,
               estimand = "ATT", method = "MRI",
               treat = "treat_multi", focal = "1")
lmw.out6
summary(lmw.out6)

# URI for a multi-category treatment; need to specify
# contrast because only two groups can be compared at
# a time
lmw.out7 <- lmw(~ treat_multi + age + education + race + married +
                  nodegree + re74 + re75, data = lalonde,
               estimand = "ATE", method = "URI",
               treat = "treat_multi", contrast = c("2", "3"))
lmw.out7
summary(lmw.out7)
```

---

lmw_est   *Estimate a treatment effect from a linear model*

Description

lmw_est() fits the outcome regression corresponding to the model used to compute the weights in the supplied lmw object and returns the model coefficients and their covariance matrix. Use [summary.lmw_est()](#) to compute and view the treatment effect and potential outcome mean estimates and their standard errors.

Usage

```
lmw_est(x, ...)

## S3 method for class 'lmw'
lmw_est(x, outcome, data = NULL, robust = TRUE, cluster = NULL, ...)

## S3 method for class 'lmw_aipw'
lmw_est(x, outcome, data = NULL, robust = TRUE, cluster = NULL, ...)
```

```
## S3 method for class 'lmw_iv'
lmw_est(x, outcome, data = NULL, robust = TRUE, cluster = NULL, ...)
```

Arguments

x	an lmw or lmw_iv object; the output of a call to lmw() or lmw_iv().
...	other arguments passed to sandwich::vcovHC() or sandwich::vcovCL().
outcome	the name of the outcome variable. Can be supplied as a string containing the name of the outcome variable or as the outcome variable itself. If not supplied, the outcome variable in the formula supplied to lmw() or lmw_iv(), if any, will be used.
data	an optional data frame containing the outcome variable named in outcome and the cluster variable(s) when cluster is supplied as a formula.
robust	whether to compute the robust covariance matrix for the model coefficients. Allowable values include those allowed for the type argument of sandwich::vcovHC() or sandwich::vcovCL() when cluster is specified. Can also be specified as TRUE (the default), which means "HC3" or "HC1" when cluster is specified, or FALSE, which means "const" (i.e., the standard non-robust covariance). When cluster is specified, robust will be set to TRUE if FALSE. When AIPW is used, robust is ignored; the HC0 robust covariance matrix is used.
cluster	the clustering variable(s) for computing a cluster-robust covariance matrix. See sandwich::vcovCL(). If supplied as a formula, the clustering variables must be present in the original dataset used to compute the weights or data. When AIPW is used, cluster is ignored.

Details

lmw_est() uses lm.fit() or lm.wfit() to fit the outcome regression model (and first stage model for lmw_iv objects) and returns the output of these functions augmented with other components related to the estimation of the weights. Unlike with lm.[w]fit(), the covariance matrix of the parameter estimates is also included in the output.

For lmw objects, the model fit is that supplied to the formula input to lmw() except that it is fit in a dataset appropriately centered to ensure the estimand corresponds with the one requested. When method = "MRI" in the call to lmw(), the model is fit as an interaction between the treatment and all the (centered) terms in the model formula. The results will be similar to those from using lm() on this model and supplied data except that the covariates are centered beforehand. The product of the sampling weights and base weights supplied to lmw(), if any, will be supplied to lm.wfit() to fit the model using weighted least squares.

For lmw_aipw objects, the model is fit as above except that base weights are not included in the model fitting and are instead used to compute additional augmentation terms that are added to the estimated potential outcome means from the outcome regression. The variance-covariance matrix is computed using M-estimation; this corresponds to the HC0 robust covariance matrix for the model parameters with the base weights treated as fixed, which yields conservative standard errors for the ATE. Inference is only approximate for the ATT and ATC.

For lmw_iv objects, the first stage model is constructed by removing the treatment from the supplied model formula, adding the instrumental variable as a main effect, and using the treatment variable

as the outcome. For the second stage (reduced form) model, the fitted values of the treatment from the first stage model are used in place of the treatment in the outcome model. The results are similar to those from using `ivreg::ivreg()`, and the coefficients estimates will be the same except for the intercept due to the centering of covariates.

Although some coefficients in the model may be interpretable as treatment effect estimates, `summary.lmw_est()` should be used to view and extract the treatment effect and potential outcome mean estimates, standard errors, and other model statistics. The output of `lmw_est()` should rarely be used except to be supplied to `summary()`.

Value

An `lmw_est` object with the following components:

`coefficients, residuals, fitted.values, effects, weights, rank, df.residual, qr`
for lmw objects, the output of the `lm.fit()` or `lm.wfit()` call used to fit the outcome model. For `lmw_iv` objects, the output of the `lm.fit()` or `lm.wfit()` call used to fit the the second stage model, with `residuals` corresponding to the residuals computed when substituting the true treatment variable in place of the fitted treatment values in the model.

`model.matrix` the model matrix (supplied to the x argument of `lm.fit`).

`vcov`   the estimated covariance matrix of the parameter estimates as produced by `sandwich::vcovHC()` or `sandwich::vcovCL()`.

`lmw.weights`   the implied regression weights computed by `lmw_est()`.

`call`   the call to `lmw_est()`.

`estimand`   the requested estimand.

`focal`   the focal treatment level when `estimand` is "ATT" or "ATC".

`method`   the method used to estimate the weights ("URI" or "MRI").

`robust`   the type standard error used.

`outcome`   the name of the outcome variable.

`treat_levels` the levels of the treatment.

When AIPW is used, the object will be of class `lmw_est_aipw`, which inherits from `lmw_est`, and contains the additional components:

`coef_aipw`   the model-predicted potential outcome means (mu) and the augmentation terms (aug).

`vcov_aipw`   the covariance matrix of the quantities in `coef_aipw`.

When weights are included in the estimation (i.e., `base.weights` or `s.weights` supplied to `lmw()` or `lmw_iv()`), any units will weights equal to zero will be removed from the data prior to model fitting.

Methods exist for `lmw_est` objects for `model.matrix()`, `vcov()`, `hatvalues()`, `sandwich::bread()`, and `sandwich::estfun()`, all of which are used internally to compute the parameter estimate covariance matrix. The first two simply extract the corresponding component from the `lmw_est` object and the last three imitate the corresponding methods for `lm` objects (or `ivreg` objects for `lmw_iv`

inputs). Other regression-related functions, such as coef(), residuals(), and fitted(), use the default methods and should work correctly with lmw_est objects.

Note that when fixed effects are supplied through the fixef argument to lmw() or lmw_iv(), standard error estimates computed using functions outside **lmw** may not be accurate due to issues relating to degrees of freedom. In particular, this affects conventional and HC1-robust standard errors. Otherwise, sandwich::vcovHC() can be used to compute standard errors (setting type = "const" for conventional standard errors), though sandwich::vcovCL() may not work as expected and should not be used. To calculate cluster-robust standard errors, supply an argument to cluster in lmw_est().

Note

lmw_est() uses non-standard evaluation to interpret its outcome argument. For programmers who wish to use lmw_est() inside other functions, an effective way to pass the name of an arbitrary outcome (e.g., y passed as a string) is to use do.call(), for example:

```
fun <- function(model, outcome, data) {
do.call("lmw_est", list(model, outcome, data)) }
```

When using lmw_est() inside lapply() or purrr::map to loop over outcomes, this syntax must be used as well.

See Also

summary.lmw_est() for viewing and extracting the treatment effect and potential outcome mean estimates, standard errors, and other model statistics; lmw() or lmw_iv() for estimating the weights that correspond to the model estimated by lmw_est(); lm() and lm.fit() for fitting the corresponding model; ivreg() in the **ivreg** package for fitting 2SLS models; influence.lmw_est() for influence measures

Examples

```
data("lalonde")

# MRI regression for ATT
lmw.out1 <- lmw(~ treat + age + education + race + married +
                nodegree + re74 + re75, data = lalonde,
                estimand = "ATT", method = "MRI",
                treat = "treat")

lmw.fit1 <- lmw_est(lmw.out1, outcome = "re78")
lmw.fit1

summary(lmw.fit1)


# MRI regression for ATT after propensity score matching
m.out <- MatchIt::matchit(treat ~ age + education + race +
                          married + nodegree + re74 + re75,
                        data = lalonde, method = "nearest",
                        estimand = "ATT")
```

```
lmw.out2 <- lmw(~ treat + age + education + race + married +
                  nodegree + re74 + re75, data = lalonde,
                method = "MRI", treat = "treat", obj = m.out)

## Using a cluster-robust SE with subclass (pair membership)
## as the cluster variable
lmw.fit2 <- lmw_est(lmw.out2, outcome = "re78", cluster = ~subclass)
lmw.fit2

summary(lmw.fit2)

# AIPW for ATE with MRI regression after propensity score
# weighting
ps <- glm(treat ~ age + education + race + married + nodegree +
            re74 + re75, data = lalonde,
          family = binomial)$fitted
ipw <- ifelse(lalonde$treat == 1, 1/ps, 1/(1-ps))

lmw.out3 <- lmw(re78 ~ treat + age + education + race + married +
                  nodegree + re74 + re75, data = lalonde,
                method = "MRI", treat = "treat",
                base.weights = ipw, dr.method = "AIPW")
lmw.fit3 <- lmw_est(lmw.out3)
lmw.fit3

summary(lmw.fit3)

# MRI for multi-category treatment ATE
lmw.out3 <- lmw(~ treat_multi + age + education + race + married +
                  nodegree + re74 + re75, data = lalonde,
                estimand = "ATE", method = "MRI",
                treat = "treat_multi")
lmw.fit3 <- lmw_est(lmw.out3, outcome = "re78")
lmw.fit3

summary(lmw.fit3)
```

---

lmw_iv   *Compute instrumental variable regression-implied weights*

Description

Computes the weights implied by an instrumental variable (IV) model that would estimate a weighted difference in outcome means equal to the treatment effect resulting from the supplied model fit with two-stage least squares.

Usage

```
lmw_iv(
  formula,
```

```
  data = NULL,
  estimand = "ATE",
  method = "URI",
  treat = NULL,
  iv,
  base.weights = NULL,
  s.weights = NULL,
  obj = NULL,
  fixef = NULL,
  target = NULL,
  target.weights = NULL,
  contrast = NULL,
  focal = NULL
)
```

Arguments

formula	a one-sided formula with the treatment and covariates on the right-hand side corresponding to the second-stage (reduced form) outcome regression model to be fit. If an outcome variable is supplied on the left-hand side, it will be ignored. This model should not include an IV. See Details for how this formula is interpreted in light of other options.
data	a data frame containing the variables named in formula, treat, and iv.
estimand	the estimand of interest, which determines how covariates are centered. Should be one of "ATE" for the average treatment effect, "ATT" for the average treatment effect in the treated, "ATC" for the average treatment effect in the control, or "CATE" for the conditional average treatment effect. When estimand = "CATE", an argument to target must be supplied. This argument also affects what [summary.lmw()](#) considers to be the target population. Default is "ATE" unless obj is specified, in which case it takes its value from the supplied object.
method	the method used to estimate the weights; either "URI" (the default) for uni-regression imputation weights, where a single model is fit to the whole dataset, or "MRI" for multi-regression imputation, where the covariates fully interact with the treatment. This affects the interpretation of formula. See Details.
treat	the name of the treatment variable in data. If unspecified, the first variable present in formula will be taken as the treatment variable with a message. Currently, only binary treatments are supported. See Details.
iv	a character vector or one-sided formula containing the names of the IVs in data. These variables should not appear in formula. Multiple IVs are allowed. See Details. This argument is required.
base.weights	a vector of base weights. See Details. If omitted and obj is specified, the weights from the supplied object will be used. Can be supplied as a numeric vector, a string containing the name of the variable in data containing the base weights, or the unquoted name of the variable in data containing the base weights.
s.weights	a vector of sampling weights. See Details. If omitted and obj is specified, the sampling weights from the supplied object will be used. Can be supplied as a numeric vector, a string containing the name of the variable in data containing

	the sampling weights, or the unquoted name of the variable in `data` containing the sampling weights.
`obj`	a `matchit` or `weightit` object corresponding to the matched or weighted sample in which the implied IV regression would take place. See Details.
`fixef`	optional; a string or one-sided formula containing the name of the fixed effects variable in `data`. See Details.
`target`	a list or data frame containing the target values for each covariate included in `formula`. Ignored with a warning when `estimand` is not `"CATE"`.
`target.weights`	a vector of sampling weights to be applied to `target` when supplied as a data frame. Ignored with a warning when `estimand` is not `"CATE"`.
`contrast`	ignored.
`focal`	the level of the treatment variable to be considered "focal" (i.e., the "treated" level when `estimand = "ATT"` or the control level when `estimand = "ATC"`). Ignored when `estimand` is `"ATE"` or `"CATE"`. For binary treatments, this generally does not need to be supplied.

Details

`lmw_iv()` computes weights that make the weighted difference in outcome means between the treatment groups equal to the two-stage least squares (2SLS) estimate of the treatment effect. `formula` corresponds to the second-stage (reduced form) model, with the treatment replaced by its fitted values resulting from the first stage model. The first stage is fit by replacing the treatment in the supplied `formula` with the IVs named in `iv` and using the treatment as the outcome. The treatment is assumed to be endogenous and the supplied instrumental variables assumed to be instruments conditional on the other covariates, which are assumed to to be exogenous.

When any treatment-by-covariate interactions are present in `formula` or when `method = "MRI"`, covariates are centered at specific values to ensure the resulting weights correspond to the desired estimand as supplied to the `estimand` argument. For the ATE, the covariates are centered at their means in the full sample. For the ATT and ATC, the covariates are centered at their means in the treatment or control group (i.e., the `focal` group), respectively. For the CATE, the covariates are centered according to the argument supplied to `target` (see below). Note that when covariate-by-covariate interactions are present, they will be centered after computing the interaction rather than the interaction being computed on the centered covariates unless `estimand = "CATE"`, in which case the covariates will be centered at the values specified in `target` prior to involvement in interactions. Note that the resulting effect estimate does not actually correspond to the estimand supplied unless all effect heterogeneity is due to the included covariates.

When treatment-by-covariate interactions are included in `formula`, additional instruments will be formed as the product of the supplied IVs and the interacting covariates. When `method = "MRI"`, instruments will be formed as the product of the supplied IVs and each of the covariates. All treatment-by-covariate interactions are considered endogenous.

Base weights and sampling weights:

Base weights (`base.weights`) and sampling weights (`s.weights`) are similar in that they both involve combining weights with an outcome regression model. However, they differ in a few ways. Sampling weights are primarily used to adjust the target population; when the outcome model is fit, it is fit using weighted least squares, and when target balance is assessed, it is assessed

using the sampling weighted population as the target population. Centering of covariates in the outcome model is done using the sampling weighted covariate means. Base weights are primarily used to offer a second level of balancing beyond the implied regression weights, i.e., to fit the 2SLS models in the base-weighted sample. Base weights do not change the target population, so when target balance is assessed, it is assessed using the unweighted population as the target population.

Some forms of weights both change the target population and provide an extra layer of balancing, like propensity score weights that target estimands other than the ATT, ATC, or ATE (e.g., overlap weights), or matching weights where the target population is defined by the matching (e.g., matching with a caliper, cardinality matching, or coarsened exact matching). Because these weights change the target population, they should be supplied to s.weights to ensure covariates are appropriately centered. In lmw_iv(), whether weights are supplied to base.weights or s.weights will not matter for the estimation of the weights but will affect the target population in balance assessment.

When both base.weights and s.weights are supplied, e.g., when the base weights are the result of a propensity score model fit with sampling weights, it is assumed the base weights do not incorporate the sampling weights; that is, it is assumed that to estimate a treatment effect *without* regression adjustment, the base weights and the sampling weights would have to be multiplied together. This is true, for example, for the weights in a matchit or weightit object (see below) but not for weights in the output of MatchIt::match.data() unless called with include.s.weights = FALSE or weights resulting from CBPS::CBPS().

2SLS after using MatchIt or WeightIt: Instrumental variable regression weights can be computed in a matched or weighted sample by supplying a matchit or weightit object (from **MatchIt** or **WeightIt**, respectively) to the obj argument of lmw(). The estimand, base weights, and sampling weights (if any) will be taken from the supplied object and used in the calculation of the implied regression weights, unless these have been supplied separately to lmw_iv(). The weights component of the supplied object containing the matching or balancing weights will be passed to base.weights and the s.weights component will be passed to s.weights. Arguments supplied to lmw_iv() will take precedence over the corresponding components in the obj object.

Multi-category treatments: Multi-category treatments are not currently supported for lmw_iv().

Fixed effects: A fixed effects variable can be supplied to the fixef argument. This is equivalent to adding the fixed effects variable as an exogenous predictor that does not interact with the treatment, IV, or any other covariate. The difference is that computation is much faster when the fixed effect has many levels because demeaning is used rather than including the fixed effect variable as a collection of dummy variables. When using URI, the weights will be the same regardless of whether the fixed effect variable is included as a covariate or supplied to fixef; when using MRI, results will differ because the fixed effect variable does not interact with treatment. The fixed effects variable will not appear in the summary.lmw() output (but can be added using addlvariables argument) or in the model output of lmw_est() or summary.lmw_est(). Because it does not interact with the treatment, the distribution of the fixed effect variable may not correspond to the target population, so caution should be used if it is expected the treatment effect varies across levels of this variable (in which case it should be included as a predictor). Currently only one fixed effect variable is allowed.

Value

An lmw_iv object, which inherits from lmw objects and contains the following components:

treat	the treatment variable, given as a factor.
weights	the computed implied regression weights.
covs	a data frame containing the covariates included the model formula.
estimand	the requested estimand.
method	the method used to estimate the weights (″URI″ or ″MRI″).
base.weights	the weights supplied to base.weights.
s.weights	the weights supplied to s.weights.
call	the original call to lmw_iv().
fixef	the fixed effects variable if supplied to fixef.
formula	the model formula.
iv	the instrumental variables, given as a one-sided formula.
target	the supplied covariate target values when estimand = ″CATE″, after some initial processing.
contrast	the contrasted treatment groups.
focal	the focal treatment levels when estimand is ″ATT″ or ″ATC″.

All functions that lack a specific lmw_iv method work with lmw_iv objects as they do for lmw objects, such as summary.lmw(), plot.lmw(), etc.

References

Chattopadhyay, A., & Zubizarreta, J. R. (2023). On the implied weights of linear regression for causal inference. *Biometrika*, 110(3), 615–629. doi:10.1093/biomet/asac058

See Also

summary.lmw() for summarizing balance and representativeness; plot.lmw() for plotting features of the weights; lmw_est() for estimating treatment effects from lmw_iv objects; influence.lmw() for influence measures; ivreg() in the **ivreg** package for fitting 2SLS models.

Examples

```
# URI for the ATT using instrument `Ins`
lmw.out <- lmw_iv(~ treat + age + education + race +
                    re74, data = lalonde,
                  estimand = "ATT", method = "URI",
                  treat = "treat", iv = ~Ins)
lmw.out
summary(lmw.out, addlvariables = ~married + re75)
```

plot.lmw   *Plots diagnosing regression-implied weights*

Description

Produces plots to diagnose properties of the weights, including their distribution, to what degree the distribution of covariates involves extrapolation in the weighted sample, and how much influence each unit has on the effect estimate.

Usage

```
## S3 method for class 'lmw'
plot(x, type = "weights", ...)
```

Arguments

x   an lmw object; the output of a call to `lmw()`.

type   the type of plot to display. Allowable options include `"weights"`, `"extrapolation"`, and `"influence"`. See Details. Abbreviations allowed.

...   further arguments passed to specific types of plots.

When type = `"weights"`, the following are accepted:

rug `logical`; whether to display a rug plot of the weights. Default is TRUE.

mean whether to display a red line indicating the mean of the weights. Default is TRUE.

ess whether to display the original and weighted effective sample size in the top right corner. Default is TRUE.

Other arguments are passed to `density()`.

When type = `"extrapolation"`, the following are accepted:

variables required; a right-sided formula or character vector containing the names of the covariates for which extrapolation is to be assessed.

data an optional data frame containing the variables named in variables.

When type = `"influence"`, the following are accepted:

outcome the name of the outcome variable. Can be supplied as a string containing the name of the outcome variable or as the outcome variable itself. If not supplied, the outcome variable in the `formula` supplied to `lmw()`, if any, will be used.

data an optional data frame containing the outcome variable named in outcome.

id.n the number of points to be labelled in the plot, starting with the most extreme.

Details

When `type = "weights"`, `plot.lmw()` produces a density plot of the weights within each treatment group. By construction, these weights will have a mean of 1. Some weights may be negative. The effective sample size (ESS) and original sample size (N) will be displayed in the upper right corner of the plot when `ess = TRUE`.

When `type = "extrapolation"`, `plot.lmw()` produces a plot of the distribution of weights and covariates for each treatment group. Each dot represents a unit, with values arranged on the x-axis according to their covariate value and the size of the dots corresponding to the magnitude of the weight. Units with positive weights are displayed in black in the upper portion of the plot, and units with negative weights are displayed in red in the lower portion. Having many and large red points indicates a high degree of extrapolation. All points are equally transparent, so darker regions indicate multiple points with the same value. The vertical lines indicates the weighted mean of the covariate in each group, and the X indicates the mean of the covariate in the target sample as determined by the `estimand` argument in the original call to `lmw()`. A large discrepancy between the vertical lines and Xs indicates a lack of balance between the treatment group and target sample. When `estimand = "CATE"` in the original call to `lmw()`, any variables supplied to `variables` that were not given a target value will not have the target mean displayed.

When `type = "influence"`, `plot.lmw()` produces a plot of the scaled sample influence curve (SIC) for each unit by index. It does so by calling [influence.lmw()](#), which fits the outcome model to extract residuals and compute the SIC as `SIC = (N-1) * w * r / (1 - h)`, where N is the sample size, w are the units' implied regression weights, r are the residuals, and h are the hat values. SIC values are scaled to have a maximum of 1. Higher values indicate greater relative influence.

Value

A plot is displayed, and `x` is invisibly returned.

See Also

[lmw()](#), [summary.lmw()](#), [plot.summary.lmw()](#)

Examples

```
data("lalonde")

# URI regression for ATT
lmw.out1 <- lmw(~ treat + age + education + race + married +
                  nodegree + re74 + re75, data = lalonde,
              estimand = "ATT", method = "URI",
              treat = "treat")
lmw.out1

# Distribution of weights
plot(lmw.out1, type = "weights")

# Extrapolation/representativeness for age and married
plot(lmw.out1, type = "extrapolation",
     variables = ~age + married)
```

```
# Extrapolation/representativeness for race
plot(lmw.out1, type = "extrapolation",
     variables = ~race)

# Influence for re78 outcome
plot(lmw.out1, type = "influence", outcome = "re78")
```

---

plot.lmw_est   *Plot diagnostics for an* lmw_est *object*

Description

Produces plots to diagnose the regression model fit to estimate the treatment effect. These include
an influence plot based on the sample influence curve (SIC) and the regression diagnostics plots
available for lm objects in [plot.lm()](#).

Usage

```
## S3 method for class 'lmw_est'
plot(x, type = "influence", ...)
```

Arguments

x	an lmw_est object; the output of a call to [lmw_est()](#).
type	the type of plot to display. Allowable options include "influence" and "lm". See Details. Abbreviations allowed.
...	When type = "influence", the following are accepted:
	outcome the name of the outcome variable. Can be supplied as a string containing the name of the outcome variable or as the outcome variable itself. If not supplied, the outcome variable in the formula supplied to lmw(), if any, will be used.
	data an optional data frame containing the outcome variable named in outcome.
	id.n the number of points to be labelled in the plot, starting with the most extreme.
	When type = "lm", any arguments passed to [plot.lm()](#) are accepted and passed directly to plot.lm.

Details

When type = "influence", plot.lmw_est() produces a plot of the scaled sample influence curve
(SIC) for each unit by index. It does so by calling [influence.lmw_est()](#), which extract the model
residuals and computes the SIC as SIC = (N-1) * w * r / (1 - h), where N is the sample size, w are
the units' implied regression weights, r are the residuals, and h are the hat values. SIC values are
scaled to have a maximum of 1. Higher values indicate greater relative influence.

When type = "lm", plot.lmw_est() produces several plots displayed sequentially according to the
arguments supplied to plot(). These plots are produced by [plot.lm()](#) to diagnose the distribution
of residuals and other measures of leverage and influence.

Value

A plot is displayed, and x is invisibly returned.

See Also

[lmw_est()](), [influence.lmw_est()](), [plot.lm()]()

Examples

```
data("lalonde")

# URI regression for ATT
lmw.out1 <- lmw(~ treat + age + education + race + married +
                  nodegree + re74 + re75, data = lalonde,
              estimand = "ATT", method = "URI",
              treat = "treat")

lmw.fit1 <- lmw_est(lmw.out1, outcome = "re78")
lmw.fit1

# Influence using SIC
plot(lmw.fit1, type = "influence")

# Usual regression diagnostics
plot(lmw.fit1, type = "lm", which = 1)
```

---

plot.summary.lmw   *Produce a Love plot of balance statistics*

Description

Produces Love plots (also known as dot plots) of balance statistics to summarize balance visually. The plots are generated using [dotchart()]() and [points()]().

Usage

```
## S3 method for class 'summary.lmw'
plot(
  x,
  stats,
  abs = TRUE,
  var.order = "data",
  threshold = NULL,
  layout = "vertical",
  ...
)
```

Arguments

x	a summary.lmw object; the output of a call to summary.lmw() with standardize = TRUE.
stats	a vector of the names of the columns in the summary.lmw output to plot; more than one is allowed. Abbreviations allowed. When unspecified, the TSMD statistics for each treatment group will be plotted.
abs	logical; whether the statistics should be plotted in absolute value (TRUE) or not (FALSE). Default is TRUE. This does not affect the display of KS statistics (which are always non-negative). When TRUE and standardized mean differences are displayed, the x-axis title will be "TASMD", i.e., target absolute standardized mean difference.
var.order	how the variables should be ordered. Allowable options include "data", ordering the variables as they appear in the summary output, "alphabetical", ordering the variables alphabetically, and, when un = TRUE in the call to summary.lmw(), "unadjusted", ordering the variables by the first statistic in stats in the unadjusted sample. Default is "data". Abbreviations allowed.
threshold	numeric values at which to place vertical lines indicating a balance threshold. These can make it easier to see for which variables balance has been achieved given a threshold. Multiple values can be supplied to add multiple lines. When abs = FALSE, the lines will be displayed on both sides of zero. The lines are drawn with abline with the linetype (lty) argument corresponding to the order of the entered variables (see options at par()). Enter a value as NA to skip that value of lty (e.g., c(NA, .05) to have only a dashed vertical line at .05).
layout	how the multiple plots should be laid out. Allowable options include "vertical" (the default) and "horizontal". Abbreviations allowed.
...	further arguments passed to dotplot().

Details

Love plots will be produced for the requested statistics in the summary.lmw output. How these plots are arranged depends on the value supplied to layout, which uses layout() to arrange the plots.

Value

A plot is displayed, and x is invisibly returned.

See Also

summary.lmw()

Examples

```
data("lalonde")

# URI regression for ATT
lmw.out1 <- lmw(~ treat + age + education + race + married +
                  nodegree + re74 + re75, data = lalonde,
```

```
                      estimand = "ATT", method = "URI",
                      treat = "treat")
lmw.out1
(s <- summary(lmw.out1))

plot(s)
plot(s, stats = "SMD", abs = FALSE)
```

---

summary.lmw   *Assess balance for an* lmw *object*

Description

Computes balance statistics for an lmw object created by [lmw()](). Balance involves not only the similarity between the treatment groups but also the similarity between each treatment group and the target population.

Usage

```
## S3 method for class 'lmw'
summary(
  object,
  un = TRUE,
  addlvariables = NULL,
  standardize = TRUE,
  data = NULL,
  stat = "balance",
  ...
)

## S3 method for class 'lmw_multi'
summary(
  object,
  un = TRUE,
  addlvariables = NULL,
  standardize = TRUE,
  data = NULL,
  contrast = NULL,
  stat = "balance",
  ...
)

## S3 method for class 'summary.lmw'
print(x, digits = max(3, getOption("digits") - 4), ...)
```

Arguments

object	an lmw object; the output of a call to [lmw()]().
un	logical; whether to display balance statistics for the sample prior to weighting and, additionally, with base weights applied (if supplied). If s.weights were supplied to lmw(), the unadjusted sample will be weighted by the sampling weights.
addlvariables	additional variables for which balance statistics are to be computed along with the covariates in the lmw object. Can be entered in one of three ways: as a data frame of covariates with as many rows as there were units in the original lmw() call, as a string containing the names of variables in data, or as a right-sided formula with the additional variables (and possibly their transformations) found in data, the environment, or the lmw object.
standardize	logical; whether to compute standardized (TRUE) or unstandardized (FALSE) mean differences. Default is TRUE.
data	a optional data frame containing variables named in addlvariables if specified as a string or formula.
stat	character; whether to display balance statistics (i.e., standardized mean differences and Kolmogorv-Smirnov statistics; "balance") or distribution statistics (i.e., means and standard deviations; "distribution"). Default is "balance". Abbreviations allowed.
...	ignored.
contrast	for multi-category treatments with method = "MRI", which two groups should be compared. If NULL, only target balance statistics will be displayed. Ignored with binary treatments or when method = "URI".
x	a summary.lmw object.
digits	the number of digits to print.

Details

summary.lmw() produces covariate balance or distribution statistics and effective samples sizes before and after adjustment by the regression weights and base weights, if supplied. For each covariate, the following balance statistics are computed when stat = "balance":

- SMD - the standardized mean difference (SMD) between the treated and control groups
- TSMD Treated - the target standardized mean difference (TSMD) between the treated group and target sample
- TSMD Control - the TSMD between between the control group and target sample
- KS - the Kolmogorov-Smirnov (KS) statistic between the treated and control groups
- TKS Treated - the target KS (TKS) statistic between the treated group and target sample
- TKS Control - the TKS statistic between the control group and target sample

For multi-category treatments with method = "MRI", balance statistics are are computed between each treatment group and the target sample.

When `stat = "distribution"` the mean and standard deviation of each covariate is compute before and after adjustment and for the target sample. (Standard deviations are printed in parentheses for visual clarity.)

After weighting with the regression weights, the mean difference between the treated and control groups of each covariate included in the original call to `lmw()` will be equal to zero. However, the mean difference between each treatment group and the target sample may not be equal to zero when `method = "URI"` in the call to `lmw()`, and covariates supplied to `addlvariables` not included in the call to `lmw()` may not be well balanced.

When `s.weights` are supplied to `lmw()`, the unadjusted statistics (if requested) will incorporate the sampling weights. When `base.weights` are supplied to `lmw()`, the unadjusted statistics will *not* incorporate the base weights; rather, balance with base weights applied (if supplied) will be produced in a separate balance table (see Value below).

SMDs are computed as the difference between the (weighted) means divided by a standardization factor, which is the standard deviation of the covariate in the target sample. When `estimand = "ATT"` in the call to `lmw()`, the standardization factor is the standard deviation in the treated group; when `estimand = "ATC"`, the standardization factor is the standard deviation in the control group; when `estimand = "ATE"` or when `estimand = "CATE"` and a target profile is supplied, the standardization factor is the square root of the average of the variances of both treatment groups; when `estimand = "CATE"` and a target dataset is supplied, the standardization factor is the standard deviation in the target dataset. When `s.weights` is supplied, the standardization factor is computed including the sampling weights; otherwise it is computed in the unweighted sample.

For binary covariates, the KS statistic is equal to the unstandardized difference in means and is computed as such.

When `estimand = "CATE"` in the original call to `lmw()`, any variables supplied to `addlvariables` that were not given a target value will not have any target statistics computed (e.g., TSMD, TKS, target means, etc.).

The effective sample size (ESS) is computed within each group as $(\sum w)^2 / \sum w^2$. With uniform weights, this is equal to the sample size.

Value

A `summary.lmw` object, which contains the following components:

`call`	The original call to `lmw()`.
`nn`	The (effective) sample sizes before and after weighting.
`bal.un`	When `stat = "balance"` and `un = TRUE`, the balance statistics prior to weighting.
`bal.base.weighted`	When `stat = "balance"`, `un = TRUE` and base weights were supplied to `lmw()`, the balance statistics with the base weights applied.
`bal.weighted`	When `stat = "balance"`, the balance statistics with the implied regression weights applied.
`dist.un`	When `stat = "distribution"` and `un = TRUE`, the distribution statistics prior to weighting.

dist.base.weighted

        When `stat = "distribution"`, `un = TRUE` and base weights were supplied to `lmw()`, the distribution statistics with the base weights applied.

dist.weighted   When `stat = "distribution"`, the distribution statistics with the implied regression weights applied.

method   The method used to estimate the weights (i.e., URI or MRI)

base.weights.origin

        If base weights were supplied through the `obj` argument to `lmw()`, their origin (i.e, **MatchIt** or **WeightIt**)

With multi-category treatments and `method = "MRI"`, the object will also inherit from class `summary.lmw_multi`.

See Also

[lmw()](#) for computing the implied regression weights, [plot.summary.lmw()](#) for plotting the balance statistics in a Love plot, [plot.lmw()](#) for assessing the representativeness and extrapolation of the weights

Examples

```
data("lalonde")

# URI regression for ATT
lmw.out1 <- lmw(~ treat + age + education + race + married +
                  nodegree + re74 + re75, data = lalonde,
                  estimand = "ATT", method = "URI",
                  treat = "treat")

lmw.out1

summary(lmw.out1)

summary(lmw.out1, stat = "distribution")

# Adding additional variables to summary, removing unweighted
summary(lmw.out1, un = FALSE,
        addlvariables = ~I(age^2) + I(nodegree*re74))


# MRI regression for ATT after PS matching
m.out <- MatchIt::matchit(treat ~ age + education + race + married +
                            nodegree + re74 + re75,
                          data = lalonde, method = "nearest",
                          estimand = "ATT")

lmw.out2 <- lmw(~ treat + age + education + race + married +
                  nodegree + re74 + re75, data = lalonde,
                method = "MRI", treat = "treat", obj = m.out)

lmw.out2
```

```
summary(lmw.out2)

# MRI for a multi-category treatment ATE
lmw.out3 <- lmw(~ treat_multi + age + education + race + married +
                  nodegree + re74 + re75, data = lalonde,
               estimand = "ATE", method = "MRI",
               treat = "treat_multi")

lmw.out3

summary(lmw.out3)

summary(lmw.out3, contrast = c("2", "1"))
```

---

summary.lmw_est_aipw   *Extract effect estimates and standard errors from* lmw_est *fits*

Description

summary() computes the treatment effect and potential outcome mean estimates from the sup-
plied lmw_est object. It functions similarly to summary.lm() in producing estimate tables with
the estimates, standard errors, t-statistics, and p-values. Other model statistics can be additionally
requested.

Usage

```
## S3 method for class 'lmw_est_aipw'
summary(object, model = FALSE, ci = TRUE, alpha = 0.05, ...)

## S3 method for class 'lmw_est'
summary(object, model = FALSE, ci = TRUE, alpha = 0.05, ...)
```

Arguments

object	an lmw_est object; the output of a call to lmw_est.
model	logical; whether to produce a coefficient table for the outcome model coeffi-cients. Note that these values should not be interpreted or reported so they are not produced by default.
ci	logical; whether to include confidence intervals in the output.
alpha	when ci = TRUE, the alpha value used to compute the critical test statistic for the confidence interval; equivalently, 1 minus the confidence level (e.g., for a 99% confidence interval, alpha = .01 should be specified). Default is .05 for a 95% confidence interval.
...	ignored.

Details

summary.lmw_est() produces a table of treatment effect estimates corresponding to all possible pairwise contrasts between the treatment levels. These treatment effects generalize to the population implied by the regression weights, which depends on the supplied estimand, whether sampling weights were provided, and which of the MRI or URI models was requested. The treatment effects are computed using linear contrasts of the outcome model coefficients.

When method = "MRI", the potential outcome mean estimates are also reported. These correspond to the potential outcome means in the population implied by the regression weights. When method = "URI", only the treatment effects are estimated; the model-implied outcome means do not correspond to the potential outcome means for the population implied by the regression weights. That is, while the treatment effect generalizes to the population defined by the regression weights, the estimated potential outcome means do not and so are not reported.

When model = TRUE, the model coefficients and their tests statistics are additionally produced. It is inappropriate to interpret or report these values as they have no causal interpretation. This is especially true when using AIPW, as the model coefficients do not incorporate the augmentation terms.

Value

A summary.lmw_est object with the following components:

call	the original call to lmw_est()
means	a matrix containing the estimated potential outcome means, their standard errors, confidence interval limits (if requested with ci = TRUE), t-statistics, and p-values. Omitted when method = "URI" or fixef is not NULL and for lmw_iv objects.
coefficients	a matrix containing the treatment effect estimates and their standard errors, t-statistics, and p-values.When ci = TRUE, the confidence limits "95%" CI L (lower) and "95%" CI U (upper) will be included between the standard error and t-statistic columns. When AIPW is used, z-statistics and z-tests are reported instead.
model.coefficients	
	when model = TRUE, the coefficient table of the model coefficients, which has the same columns as coefficients.
aliased	when model = TRUE, a named logical vector showing if the original coefficients are aliased (i.e., NA).
sigma, df, r.squared, adj.r.squared	
	the residual standard deviation, degrees of freedom components, R-squared, and adjusted R-squared. See [summary.lm()](#). When AIPW is used, sigma and df are omitted.

Other components containing information for printing are also included.

See Also

[lmw_est()](#) for fitting the outcome regression model, [summary.lm()](#) for more information on the output components

Examples

```
# See examples at `help("lmw_est")`
```

Index