Package 'oalasso'

July 21, 2026

Type Package

Title Outcome-Adaptive Lasso Propensity Scores

Version 1.0.0

Description Estimates propensity scores by the outcome-adaptive lasso of Shortreed and Ertefaie (2017) <doi:10.1111/biom.12679> and the generalized outcome-adaptive lasso (GOAL) of Balde, Yang and Lefebvre (2023) <doi:10.1111/biom.13683>, using 'glmnet' with an exact penalty-scale correction so that the published objectives and tuning grids are reproduced. Tuning is by the weighted absolute mean difference balance criterion. The resulting score is designed to be supplied directly to the matchit() function of 'MatchIt' as a distance measure, to the weightit() function of 'WeightIt' as a propensity score, or to the psave() function of 'psAve' as an appended candidate.

Depends R (>= 4.1)

Imports glmnet (>= 4.1-2), cobalt (>= 4.6.0), stats, utils, graphics

Suggests MatchIt, WeightIt, psAve, rpart, testthat (>= 3.0.0), knitr, rmarkdown

VignetteBuilder knitr

Config/testthat/edition 3

License GPL (>= 2)

URL https://kabajiro.github.io/oalasso/, https://github.com/kabajiro/oalasso

BugReports https://github.com/kabajiro/oalasso/issues

Encoding UTF-8

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Daijiro Kabata [aut, cre, cph]

Maintainer Daijiro Kabata <daijiro.kabata@port.kobe-u.ac.jp>

Repository CRAN

Date/Publication 2026-07-21 10:30:14 UTC

Contents

`bal.tab.oal` *Balance tables for oal objects*

Description

A method for `cobalt::bal.tab()`: assesses balance on the (original-scale) covariates of an `oal()` fit under the inverse-probability weights implied by the outcome-adaptive lasso propensity score, which is supplied as a `distance` measure.

Usage

```
## S3 method for class 'oal'
bal.tab(x, ...)
```

Arguments

`x`	An `oal` object.
`...`	Further arguments passed on to `cobalt::bal.tab()` (e.g., `un = TRUE`, `thresholds = c(m = 0.1)`).

Details

The call delegates to the default **cobalt** machinery as `cobalt::bal.tab(<covariates>, treat = x$treat, weights = x$w` so all the usual **cobalt** arguments (`un`, `stats`, `thresholds`, ...) are available and display conventions are **cobalt**'s own. The *selection criterion* inside `oal()` is the papers' wAMD instead, computed natively on the standardized covariates; see `oal_wamd()`.

Value

A `bal.tab` object; see `cobalt::bal.tab()`.

References

Shortreed SM, Ertefaie A (2017). Outcome-adaptive lasso: variable selection for causal inference. *Biometrics*, 73(4), 1111-1122. doi:10.1111/biom.12679

See Also

`oal()`, `cobalt::bal.tab()`, `plot.oal()`

Examples

```
data("lalonde", package = "MatchIt")
fit <- oal(treat ~ age + educ + married + re74, data = lalonde,
           outcome = ~ re78)
cobalt::bal.tab(fit, un = TRUE)
```

`coef.oal` *Extract propensity score model coefficients*

Description

Returns the selected outcome-adaptive lasso propensity score coefficients (intercept first), either on the original covariate scale (default; the scale used by `predict.oal()`) or on the standardized scale that the penalized objective was actually optimized on (the scale of the adaptive weights and of the wAMD – methods-paper material).

Usage

```
## S3 method for class 'oal'
coef(object, scale = c("original", "standardized"), ...)
```

Arguments

`object`	An `oal` object.
`scale`	`"original"` (default) or `"standardized"`.
`...`	Ignored.

Value

A named numeric vector of length p + 1.

See Also

`oal()`, `predict.oal()`

`oal` *Outcome-adaptive lasso propensity scores*

Description

`oal()` estimates a propensity score by the outcome-adaptive lasso (OAL) of Shortreed and Ertefaie (2017) or its generalized elastic-net form (GOAL) of Balde, Yang and Lefebvre (2023): an adaptive lasso on the treatment (propensity score) log-likelihood whose penalty weights come from an *outcome* regression, so that covariates unrelated to the outcome – instruments and noise variables – are excluded from the propensity score model. Tuning parameters are selected by the papers' weighted absolute mean difference (wAMD) balance criterion. The result is deliberately modest: a numeric score vector designed to be handed to `MatchIt::matchit()` as `distance`, to `WeightIt::weightit()` as `ps`, or to `psAve::psave()` as an appended candidate.

Usage

```
oal(
  formula,
  data,
  outcome,
  method = c("oal", "goal"),
  estimand = c("ATE", "ATT"),
  family = gaussian(),
  lambda = c(-10, -5, -2, -1, -0.75, -0.5, -0.25, 0.25, 0.49),
  gamma = NULL,
  gamma.factor = 2,
  lambda2 = c(0, 10^c(-2, -1.5, -1, -0.75, -0.5, -0.25, 0, 0.25, 0.5, 1)),
  outcome.coef = NULL,
  refit = FALSE,
  clip = c(0.01, 0.99),
  keep.path = TRUE,
  keep.fits = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

`formula`	A two-sided formula `treat ~ x1 + x2 + ...`, exactly as in `MatchIt::matchit()`. The right-hand side defines the propensity score covariates; the same covariates (plus the treatment) form the outcome model that supplies the adaptive penalty weights.
`data`	A data frame containing the variables in `formula` and `outcome`. Complete cases in all used variables are REQUIRED; any missing value is an error naming the offending variables, never a silent row drop.

`outcome` A one-sided formula `~ y` naming the outcome variable (exactly one variable). Required unless `outcome.coef` is supplied, in which case it is unused. A two-sided formula is an error: OAL derives its penalty weights from an outcome model on the *same* covariates as the propensity score model, so there is no separate outcome design to specify.

`method` `"oal"` (default): the outcome-adaptive lasso; `"goal"`: the generalized outcome-adaptive lasso, which adds an elastic-net $\lambda_2 \sum_j \alpha_j^2$ term (grouping effect and numerical stability under correlated covariates and near-positivity violations).

`estimand` `"ATE"` (default) or `"ATT"`; determines the inverse-probability weights used inside the wAMD criterion and returned in `weights`. The ATE default is Shortreed and Ertefaie's wAMD weighting and deliberately differs from `psAve::psave()`'s ATT default.

`family` The outcome (weight) model family: `gaussian()` (default; the specification validated by the papers' simulations) or `binomial()`. `binomial()` emits a one-time warning: binomial outcome models are beyond the simulations validated by Shortreed and Ertefaie (2017); GLM theory per Balde (2025).

`lambda` Numeric vector of EXPONENTS δ; the penalties actually applied are $\lambda_n = n^{\delta}$. The default is Shortreed and Ertefaie's grid `c(-10, -5, -2, -1, -0.75, -0.5, -0.25, 0.25, 0.49)`. A guard warns when `max(lambda) > 3`: such values look like raw penalty values, not exponents (convert a raw grid via `delta = log(lambda_n)/log(n)`).

`gamma` `NULL` (default) pairs γ with each δ as $\gamma = 2(\texttt{gamma.factor} - \delta + 1)$ (the Shortreed-Ertefaie/rejoinder convention); a single number (e.g. `2.5`) fixes γ and crosses it with the whole `lambda` grid (the Schnitzer-style fixed-γ mode). No vector form is accepted.

`gamma.factor` The convergence factor `gcf` in the pairing formula (default `2`); ignored when `gamma` is given.

`lambda2` Numeric grid of elastic-net ridge constants for `method = "goal"` only (supplying it otherwise is an error). Default `c(0, 10^c(-2, -1.5, -1, -0.75, -0.5, -0.25, 0, 0.25, 0.5, 1))` – the author's published grid (Balde 2025 supplement, "taken from Zou and Hastie (2005)"), verified against the official GOAL code on 2026-07-02; `0` is always a member and nests plain OAL exactly. `lambda2` is selected jointly with `(lambda, gamma)` by the wAMD.

`outcome.coef` The extension hook: a NAMED numeric vector of outcome coefficients b_j over the **standardized** model-matrix columns (names must exactly cover the columns), overriding the internal outcome model. **The values are interpreted on the standardized scale of the internal design matrix** – adaptive weights are scale-dependent. All values must be finite and not all zero. Zeros ARE allowed and mean a hard drop: $|0|^{-\gamma} = \infty$ and glmnet natively converts an infinite penalty factor to an exclusion. No epsilon or cap is ever applied. Using this hook flips the provenance label to "screening use only" (see Details) and sets `info$gamma.mode = "user-coef"`. If `outcome` is also supplied, it is still validated and the outcome-leakage guard still runs (the outcome variable may not appear among the PS covariates); if `outcome` is omitted, the package **cannot** detect outcome leakage in `formula` – ensuring the coefficients were derived without post-treatment or outcome information is then entirely the caller's responsibility.

refit	FALSE (default): the propensity score comes from the penalized fit itself (the Shortreed-Ertefaie/rejoinder convention). TRUE: at every grid point an *unpenalized* logistic regression is refit on the selected covariates, and both the propensity score and the wAMD come from the refit (the Schnitzer et al. 2025 variant, adapted from a longitudinal setting); per-grid-point convergence is recorded in path$refit.converged and any non-convergence triggers one warning – never a silent fallback.
clip	Length-2 numeric: propensity scores are clipped to [clip[1], clip[2]] BEFORE the wAMD IPW weights are formed and in the returned ps (default c(0.01, 0.99), equal to psAve::psave()'s default so that psave's re-clipping is a no-op). The default is a Shortreed-Ertefaie-lineage safety choice; Balde's official reference code runs UNCLIPPED weights (1/e, 1/(1 - e) with no truncation). To reproduce that behavior, effectively disable clipping with clip = c(1e-12, 1 - 1e-12).
keep.path	If TRUE (default), the full per-grid-point results are stored in path, sets, and coef.path.
keep.fits	If TRUE, the fitted **glmnet** paths, the outcome model, and (with refit = TRUE) the selected refit are retained in fits. Default FALSE.
verbose	If TRUE, progress messages report the grid evaluation and the selected point.
...	Reserved for future use; supplying unused arguments triggers a warning.

Details

Objective and exact glmnet evaluation:

OAL solves, on the TOTAL log-likelihood scale,

$$\hat{\alpha} = \arg\min_{\alpha} \; -\ell(\alpha; A, X_s) + \lambda_n \sum_j |b_j|^{-\gamma} |\alpha_j|,$$

where ℓ is the binomial log-likelihood of treatment on the standardized covariates and b_j are the outcome-model coefficients. **glmnet** minimizes the *per-observation* loss and internally rescales penalty factors to sum to the number of variables, so each grid point is evaluated **exactly** at

$$s = \overline{\text{pen}} \cdot \lambda_n / n, \qquad \text{pen}_j = |b_j|^{-\gamma},$$

via coef(fit, s = s, exact = TRUE, x = , y = , penalty.factor =), which re-solves at that penalty (the KKT-verified recipe of the Shortreed/Ertefaie rejoinder code). When outcome.coef contains zeros, some $\text{pen}_j = \infty$: glmnet converts those to exclusions and internally resets their factor to 1 *before* the sum-to-nvars rescale, so the constant generalizes to $s = (\sum_{j:\text{finite}} \text{pen}_j + \#\{\text{pen}_j = \infty\})/p \cdot \lambda_n/n$ – identical to $\overline{\text{pen}}\,\lambda_n/n$ when all factors are finite. This correction is always on; it is what makes the n^δ grid carry its published meaning.

Standardization protocol (fixed, not an argument):

The model matrix X <- model.matrix(formula, data)[, -1] (default treatment contrasts) is standardized ONCE, Xs <- scale(X, TRUE, TRUE), all columns including dummies (the rejoinder convention). The outcome model AND glmnet are both fit on this SAME Xs with glmnet(standardize = FALSE). Per-subset scaling is structurally impossible. Coefficients are back-transformed for output: $\alpha_j = \alpha_j^{std}/s_j$ and $\alpha_0 = \alpha_0^{std} - \sum_j \alpha_j^{std} c_j / s_j$, with centers c and scales s stored in info.

Weight (outcome) model and the degenerate-beta policy:

By default b comes from `lm(y ~ A + Xs)` (or `glm(y ~ A + Xs, binomial)`) on the FULL sample, both arms pooled – the Shortreed-Ertefaie/rejoinder convention (not control-arm-only). Only the covariate coefficients are used; the treatment coefficient is discarded and the treatment is never penalized anywhere. Any `NA` or *exactly zero* internal coefficient is an ERROR naming the columns: OLS betas are almost surely nonzero, so an exact zero or `NA` signals rank deficiency or aliasing that would silently corrupt the penalty-scale correction. Remove the collinear columns or supply `outcome.coef`.

Tuning grid and the γ pairing:

`lambda` supplies exponents δ with $\lambda_n = n^\delta$, `n = nrow(model.matrix)` always. With `gamma = NULL` each δ is paired with $\gamma = 2(\mathrm{gcf} - \delta + 1)$ (default $\mathrm{gcf} = 2$), the formula derivable from $\lambda_n\, n^{\gamma/2-1} = n^{\mathrm{gcf}}$; there are no $\lambda \times \gamma$ cross-products in paired mode. A scalar `gamma` is crossed with all δ (Schnitzer's $\gamma = 2.5$ is reachable this way; convert their raw λ grid via $\delta = \log \lambda_n / \log n$).

GOAL:

For $\lambda_2 > 0$ the elastic-net term is solved by the Zou-Hastie augmentation: $X_{aug} = \mathrm{rbind}(X_s, \sqrt{\lambda_2} I_p)$ with p pseudo-responses 0, followed by rescaling ALL coefficients INCLUDING the intercept by $(1 + \lambda_2)$ (Balde 2025 supplement: $PS = \mathrm{expit}(\mathrm{cbind}(1, X)\,(1 + \lambda_2)\,\hat{\alpha})$); the propensity score uses the original n rows only. On augmented grid points the raw penalty constant is $\lambda_n = (n+q)^\delta$ with $q = p$ augmentation rows (Balde's `adaptive.lasso(lambda = n.q^(il))`, `n.q = n + q`), and the penalty-scale constant uses $1/(n + q)$ in place of $1/n$, i.e. $s = \overline{\mathrm{pen}}(n + q)^\delta/(n + q)$. $\lambda_2 = 0$ grid points run the plain-OAL code path (no augmentation, $\lambda_n = n^\delta$), so `method = "goal"` with `lambda2 = 0` nests `method = "oal"` exactly. (Balde's script literally augments with q zero rows and the $(n + q)^\delta$ base even at $\lambda_2 = 0$; the zero rows still shift the intercept, so exact nesting is deliberately preferred here.) λ_2 is selected jointly with (δ, γ) by the wAMD: the flat first-minimum over the λ_2-outer $\times$ δ-inner grid, which is equivalent to Balde's nested rule (per-λ_2 minima over (δ, γ), then the first minimum over λ_2).

wAMD criterion and selection:

Every grid point is scored by `oal_wamd()`: propensity scores clipped to `clip` first (note that Balde's reference implementation is unclipped; see the `clip` argument), IPW weights at `estimand`, per-column weighted absolute mean differences on the standardized matrix summed with weights $|b_j|$ over ALL columns (including excluded ones). The selected grid point is the FIRST minimum (relative tolerance `1e-9`) in the grid order λ_2 ascending (0 first) outer $\times$ δ ascending inner – ties therefore prefer plain OAL and then the smallest penalty. The wAMD is a balance heuristic: no published proof guarantees it lands λ_n inside the window required by the theory ($\lambda_n/\sqrt{n} \to 0$, $\lambda_n n^{\gamma/2-1} \to \infty$, $\gamma > 1$), and it inherits the outcome model's misspecification vulnerability.

Why exclude instruments?:

Covariates that predict treatment but not outcome should be excluded from a propensity score model: they increase the variance of the effect estimate without decreasing bias (Brookhart et al. 2006) and amplify the bias from any unmeasured confounding (Myers et al. 2011). OAL's penalty implements exactly this exclusion.

Provenance labels:

`print()` and `summary()` always print a provenance line first: `"OAL (Shortreed & Ertefaie 2017, doi:10.1111/biom.12679)"` or `"GOAL (Balde, Yang & Lefebvre 2023, doi:10.1111/biom.13683)"`

with a clause stating whether the lambda2 grid is the author's published grid (Balde 2025 supplement) or user-specified. Supplying outcome.coef appends "user-supplied outcome coefficients – screening use only; no oracle property"; family = binomial() and refit = TRUE append their own flags.

Handoff contract:
$ps is a numeric vector of length n, named by rownames(data), strictly inside (0, 1) after clipping – it satisfies psAve::psave(ps.append =), MatchIt::matchit(distance =), and WeightIt::weightit(ps =) by construction.

There is no seed argument: the whole pipeline is deterministic.

Value

An object of class "oal": a list with components

ps numeric(n), named by rownames(data): the propensity score, clipped to clip, strictly in (0, 1) – **the deliverable**.

weights numeric(n): the IPW weights at estimand implied by ps.

coefficients named numeric(p+1): selected propensity score coefficients on the ORIGINAL covariate scale (intercept first).

coefficients.std the same on the standardized scale actually optimized.

selected data frame, one row per model-matrix column: term, outcome.coef (signed b_j, standardized scale), penalty.factor, coef (original scale), selected (logical), and role ("retained"/"excluded").

lambda named numeric: delta, lambda.n ($= n^{\delta}$; $= (n+p)^{\delta}$ on GOAL grid points with $\lambda_2 > 0$), gamma, lambda2 (NA for method = "oal") at the selected point.

criterion, criterion.value "wamd" and its value at the selection.

path data frame (or NULL without keep.path), one row per grid point in grid order: lambda2, delta, lambda.n, gamma, s (the exact glmnet s used), wamd, n.selected, refit.converged (NA unless refit), selected.

sets list (or NULL): the selected variable set at every grid point.

coef.path (p+1) x G matrix (or NULL): propensity score coefficients across the grid, original scale.

outcome.coef named numeric(p): the b_j actually used (internal or user), standardized scale.

outcome.model compact list: coefficient table, family, and a label ("lm(y ~ A + X)", "glm(y ~ A + X, binomial)", or "user-supplied"); the full fit object only under keep.fits.

balance data frame per ORIGINAL covariate column: smd.un, smd.wt, ks.un, ks.wt via **cobalt** – identical layout to psAve::psave()$balance.

provenance the fixed provenance label, printed first by print()/summary().

treat integer(n) 0/1 treatment as coerced.

covs n x p numeric original-scale model matrix with attr(, "bin.vars").

estimand, method, refit, clip as resolved.

formula, data, terms, xlevels stored to power oal_match(), oal_weight() and predict.oal().

fits list (or NULL): glmnet (one path per (λ_2, γ)), outcome, refit – iff keep.fits.

info list: n, n.treated, p, grid.size, family, gamma.mode ("paired"/"fixed"/"user-coef"), center, scale, contrasts, glmnet.version, oalasso.version.

call the matched call.

References

Shortreed SM, Ertefaie A (2017). Outcome-adaptive lasso: variable selection for causal inference. *Biometrics*, 73(4), 1111-1122. doi:10.1111/biom.12679

Balde I, Yang Y, Lefebvre G (2023). Reader reaction to "Outcome-adaptive lasso: variable selection for causal inference" by Shortreed and Ertefaie (2017). *Biometrics*, 79(1), 514-520. doi:10.1111/biom.13683

Jones B, Ertefaie A, Shortreed SM (2023). Rejoinder to reader reaction "On the use of the outcome-adaptive lasso for propensity score estimation". *Biometrics*, 79(1), 521-525. doi:10.1111/biom.13681

Balde I (2025). Oracle properties of the generalized outcome-adaptive lasso. *Statistics & Probability Letters*. doi:10.1016/j.spl.2025.110379

Zou H (2006). The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101(476), 1418-1429. doi:10.1198/016214506000000735

Brookhart MA, Schneeweiss S, Rothman KJ, Glynn RJ, Avorn J, Sturmer T (2006). Variable selection for propensity score models. *American Journal of Epidemiology*, 163(12), 1149-1156. doi:10.1093/aje/kwj149

Myers JA, Rassen JA, Gagne JJ, Huybrechts KF, Schneeweiss S, Rothman KJ, Joffe MM, Glynn RJ (2011). Effects of adjusting for instrumental variables on bias and precision of effect estimates. *American Journal of Epidemiology*, 174(11), 1213-1222. doi:10.1093/aje/kwr364

Schnitzer ME, Talbot D, Liu Y, Berger C, Wang G, O'Loughlin J, Sylvestre M-P, Ertefaie A (2025). Outcome-adaptive LASSO for longitudinal data. *Statistics in Medicine* (arXiv:2410.08283).

See Also

`oal_match()`, `oal_weight()`, `oal_wamd()`, `predict.oal()`, `plot.oal()`, `bal.tab.oal()`

Examples

```
data("lalonde", package = "MatchIt")

## 1) Matching
fit <- oal(treat ~ age + educ + race + married + nodegree + re74 + re75,
           data = lalonde, outcome = ~ re78)
fit
m <- oal_match(fit)   # = MatchIt::matchit(f, lalonde, distance = fit$ps)

## 2) Weighting (requires WeightIt)
# w <- oal_weight(fit)  # = WeightIt::weightit(f, lalonde, ps = fit$ps, "ATE")

## 3) psAve composition (requires psAve); cbind() labels the candidate
##    while keeping the row names aligned with rownames(data)
# ma <- psAve::psave(treat ~ age + educ + race + married + nodegree +
#                      re74 + re75,
#                    data = lalonde, outcome = ~ re78,
#                    ps.append = cbind(oal = fit$ps))
```

`oal_match` *Match on an outcome-adaptive lasso propensity score*

Description

Convenience pass-through to `MatchIt::matchit()`: matches on the propensity score of an `oal()` fit, reusing the formula and data stored in the object. Equivalent to the canonical explicit call

```
MatchIt::matchit(<formula>, data = <data>, distance = fit$ps, ...)
```

but with no opportunity for row misalignment between the two steps. All `...` arguments are forwarded verbatim; the return value is an ordinary `matchit` object, so the full **MatchIt/cobalt** toolkit applies.

Usage

```
oal_match(object, ...)
```

Arguments

`object`	An `oal` object.
`...`	Arguments forwarded verbatim to `MatchIt::matchit()` (e.g., `method`, `caliper`, `ratio`, `replace`).

Value

A `matchit` object; see `MatchIt::matchit()`.

See Also

`oal()`, `oal_weight()`, `MatchIt::matchit()`

Examples

```
data("lalonde", package = "MatchIt")
fit <- oal(treat ~ age + educ + married + re74, data = lalonde,
           outcome = ~ re78)
m <- oal_match(fit, method = "nearest", caliper = 0.2)
```

`oal_wamd` *Weighted absolute mean difference (wAMD) of Shortreed and Ertefaie (2017)*

Description

Scores an arbitrary candidate propensity score on the exact wAMD balance criterion used by `oal()` to select its tuning parameters. This function is the package's **single source of truth** for the criterion: the internal grid search calls it verbatim, and exporting it lets any candidate score – from any package – be scored on the same yardstick (the mirror of `psAve::psave_criteria()`).

Usage

```
oal_wamd(
  ps,
  treat,
  covs,
  coef,
  estimand = c("ATE", "ATT"),
  clip = c(0.01, 0.99)
)
```

Arguments

`ps`	Numeric vector of propensity scores, strictly inside (0, 1) before clipping.
`treat`	Treatment vector: numeric 0/1, logical, or a two-level factor/character (second level = treated), as in `oal()`.
`covs`	Numeric matrix of covariates, one row per unit; no missing values.
`coef`	Numeric vector of outcome-model coefficients, one per column of `covs` (matched by name when both are named, else by position). Absolute values are used as the balance weights.
`estimand`	`"ATE"` (default) or `"ATT"`; selects the IPW weight formula above.
`clip`	Length-2 numeric; the scores are clipped to `[clip[1], clip[2]]` **before** the weights are formed (default `c(0.01, 0.99)`).

Details

The propensity scores are first clipped to `clip`; the inverse-probability weights at `estimand` are then (with e_i the clipped score)

$$ATE:\ w_i = A_i/e_i + (1-A_i)/(1-e_i), \qquad ATT:\ w_i = A_i + (1-A_i)\,e_i/(1-e_i).$$

For each covariate column j the weighted absolute mean difference is

$$d_j = \left| \frac{\sum_i w_i X_{ij} A_i}{\sum_i w_i A_i} - \frac{\sum_i w_i X_{ij}(1-A_i)}{\sum_i w_i (1-A_i)} \right|,$$

and the criterion is wAMD $= \sum_j |\beta_j|\, d_j$, summed over **all** columns of `covs` (including any the propensity score model excluded), with β_j the outcome-model coefficients supplied in `coef` (their absolute values are taken internally).

Inside `oal()`, `covs` is the **standardized** model matrix and `coef` the outcome coefficients on that same standardized scale (the wAMD, like the adaptive weights, is scale-dependent). The formula is implemented natively – never delegated to **cobalt** – so that it reproduces the published criterion exactly.

Value

A list with elements

`total` the wAMD, $\sum_j |\beta_j| d_j$;

`by.covariate` named numeric vector of the per-covariate contributions $|\beta_j| d_j$ (they sum to `total`).

References

Shortreed SM, Ertefaie A (2017). Outcome-adaptive lasso: variable selection for causal inference. *Biometrics*, 73(4), 1111-1122. doi:10.1111/biom.12679

See Also

`oal()`, `weights.oal()`

Examples

```
set.seed(7)
X <- matrix(rnorm(200), 100, 2, dimnames = list(NULL, c("x1", "x2")))
A <- rbinom(100, 1, plogis(X[, 1]))
ps <- plogis(0.8 * X[, 1])
oal_wamd(ps, A, X, coef = c(x1 = 0.5, x2 = 0.1), estimand = "ATE")
```

---

oal_weight *Weight by an outcome-adaptive lasso propensity score*

Description

Convenience pass-through to `WeightIt::weightit()`: constructs balancing weights from the propensity score of an `oal()` fit, reusing the stored formula and data. Equivalent to the canonical explicit call

```
WeightIt::weightit(<formula>, data = <data>, ps = fit$ps, estimand = <estimand>, ...)
```

All `...` arguments are forwarded verbatim; the return value is an ordinary `weightit` object.

Usage

```
oal_weight(object, estimand = object$estimand, ...)
```

Arguments

object	An oal object.
estimand	The estimand passed to WeightIt::weightit(); defaults to the estimand of the fit (note that the wAMD *selection* already used the fitted estimand's weights).
...	Arguments forwarded verbatim to WeightIt::weightit().

Value

A weightit object; see WeightIt::weightit().

See Also

oal(), oal_match(), WeightIt::weightit(), WeightIt::get_w_from_ps()

Examples

```
data("lalonde", package = "MatchIt")
fit <- oal(treat ~ age + educ + married + re74, data = lalonde,
           outcome = ~ re78)
w <- oal_weight(fit)
```

---

plot.oal *Plot an oal object*

Description

Three diagnostic displays for an oal() fit:

"wamd" the wAMD selection criterion against the exponent δ (lambda), one curve per lambda2 value for method = "goal"; the selected point is marked. Requires keep.path = TRUE.

"coef" the original-scale propensity score coefficient paths against δ at the selected lambda2, one line per covariate; covariates driven to zero (instruments, noise) are visible as lines absorbed into the axis. Requires keep.path = TRUE.

"balance" a Love plot of covariate balance before/after weighting, via cobalt::love.plot() (dispatched through bal.tab.oal(); **cobalt** is an Import, so always available).

Usage

```
## S3 method for class 'oal'
plot(x, type = c("wamd", "coef", "balance"), ...)
```

Arguments

x	An oal object.
type	One of "wamd" (default), "coef", "balance".
...	For "balance", further arguments to cobalt::love.plot() (e.g., thresholds = 0.1); otherwise further graphical parameters passed to the base plotting calls.

Value

For "balance", the ggplot object from cobalt::love.plot() (invisibly, after printing); otherwise x, invisibly.

See Also

oal(), bal.tab.oal(), cobalt::love.plot()

---

predict.oal *Predict outcome-adaptive lasso propensity scores for new data*

Description

Computes propensity scores (or the linear predictor) for new observations from the stored terms, factor levels, and ORIGINAL-scale coefficients of an oal() fit. Works without keep.fits = TRUE and for both refit modes; the training standardization is never re-estimated (the original-scale coefficients already absorb it). Missing values in newdata are an error.

Usage

```
## S3 method for class 'oal'
predict(object, newdata = NULL, type = c("ps", "link"), ...)
```

Arguments

object	An oal object.
newdata	A data frame containing the variables of the propensity score formula, or NULL (default) for the in-sample values.
type	"ps" (default): propensity scores clipped to the fit's clip; "link": the unclipped linear predictor.
...	Ignored.

Value

A numeric vector with one value per row, named by rownames. For newdata = NULL and type = "ps" this is exactly object$ps.

See Also

oal(), coef.oal()

`print.oal` *Print an oal object*

Description

Prints a one-screen teaching summary of a fitted `oal()` object: the provenance label first (which method, which tuning provenance), the sample, the selected tuning point and its wAMD, the retained (outcome-related) versus excluded (outcome-unrelated) covariates with their outcome-coefficient magnitudes, the fixed instrument-exclusion rationale, and then the **literal next call** – echoing the formula and data name from your own `oal()` call – that hands the score to `MatchIt::matchit()` or `WeightIt::weightit()`. A near-positivity diagnostic is appended when more than 5\ clipping bound.

Usage

```
## S3 method for class 'oal'
print(x, digits = 3, ...)
```

Arguments

`x`	An `oal` object.
`digits`	Number of significant digits to print. Default 3.
`...`	Ignored.

Value

`x`, invisibly.

See Also

`oal()`, `summary.oal()`

`summary.oal` *Summarize an oal object*

Description

Produces (a) the full per-covariate selection table (`selected`), (b) a path summary (the best grid row per `lambda2` value, when the path was kept), (c) the outcome (weight) model coefficient table, (d) the full balance table (unweighted vs. weighted SMD and KS, with a * marker at weighted SMD > 0.1), and (e) the near-positivity clip diagnostic.

Usage

```
## S3 method for class 'oal'
summary(object, ...)

## S3 method for class 'summary.oal'
print(x, digits = 3, ...)
```

Arguments

object	An oal object.
...	Ignored.
x	A summary.oal object.
digits	Number of significant digits to print. Default 3.

Value

For summary.oal(), an object of class "summary.oal": a list with elements call, provenance, method, estimand, refit, lambda, criterion, criterion.value, selected, path.summary, outcome.model, balance, clip, clip.share, and nn. print.summary.oal() returns x invisibly.

See Also

oal(), print.oal(), bal.tab.oal()

---

weights.oal *Extract the inverse-probability weights of an oal fit*

Description

Returns object$weights: the IPW weights at the *fitted* estimand implied by the clipped propensity score (ATE: $A/e + (1-A)/(1-e)$; ATT: $A + (1-A)\,e/(1-e)$). For other estimands use WeightIt::get_w_from_ps(object$ps, object$treat, estimand = ...) or oal_weight().

Usage

```
## S3 method for class 'oal'
weights(object, ...)
```

Arguments

object	An oal object.
...	Ignored.

Value

The numeric vector object$weights, named by rownames(data).

See Also

oal(), oal_weight(), oal_wamd()

Index