Package 'causalBatch'

May 8, 2026

Type Package

Title Causal Batch Effects

Version 1.3.0

Date 2025-01-07

Maintainer Eric W. Bridgeford <ericwb95@gmail.com>

Description Software which provides numerous functionalities for detecting and removing group-level effects from high-dimensional scientific data which, when combined with additional assumptions, allow for causal conclusions, as-described in our manuscripts Bridgeford et al. (2024) <doi:10.1101/2021.09.03.458920> and Bridgeford et al. (2023) <doi:10.48550/arXiv.2307.13868>. Also provides a number of useful utilities for generating simulations and balancing covariates across multiple groups/batches of data via matching and propensity trimming for more than two groups.

Depends R (>= 4.2.0)

Imports cdcsis, sva, MatchIt, nnet, dplyr, magrittr, genefilter, BiocParallel, utils

URL https://github.com/neurodata/causal_batch

Encoding UTF-8

VignetteBuilder knitr

Suggests tidyr, ggpubr, knitr, rmarkdown, parallel, testthat (>= 3.0.0), covr, roxygen2, ks, ggplot2

License GPL-3

RoxygenNote 7.3.0

Config/testthat/edition 3

NeedsCompilation no

Author Eric W. Bridgeford [aut, cre], Michael Powell [ctb], Brian Caffo [ctb], Joshua T. Vogelstein [ctb]

Repository CRAN

Date/Publication 2025-01-07 12:50:07 UTC

Contents

`cb.align.kway_match` *K-Way matching*

Description

A function for performing k-way matching using the matchIt package. Looks for samples which have corresponding matches across all other treatment levels.

Usage

```
cb.align.kway_match(
  Ts,
  Xs,
  match.form,
  reference = NULL,
 match.args = list(method = "nearest", exact = NULL, replace = FALSE, caliper = 0.1),
  retain.ratio = 0.05
)
```

Arguments

`Ts`	`[n]` the labels of the samples, with `K` < `n` levels, as a factor variable.
`Xs`	`[n, r]` the `r` covariates/confounding variables, for each of the `n` samples, as a data frame with named columns.
`match.form`	A formula of columns from `Xs`, to be passed directly to `matchit` for subsequent matching. See `formula` argument from `matchit` for details.
`reference`	the name of the reference/control batch, against which to match. Defaults to `NULL`, which treats the reference batch as the smallest batch.
`match.args`	A named list arguments for the `matchit` function, to be used to specify specific matching strategies, where the list names are arguments and the corresponding values the value to be passed to `matchit`. Defaults to inexact nearest-neighbor caliper (width 0.1) matching without replacement.

`retain.ratio` If the number of samples retained is less than `retain.ratio*n`, throws a warning. Defaults to `0.05`.

Value

a list, containing the following:

- `Retained.Ids [m]` vector consisting of the sample ids of the `n` original samples that were retained after matching.
- `Reference` the reference batch.

Details

For more details see the help vignette: `vignette("causal_balancing", package = "causalBatch")`

Author(s)

Eric W. Bridgeford

References

Eric W. Bridgeford, et al. "A Causal Perspective for Batch Effects: When is no answer better than a wrong answer?" Biorxiv (2024).

Daniel E. Ho, et al. "MatchIt: Nonparametric Preprocessing for Parametric Causal Inference" JSS (2011).

Examples

```
library(causalBatch)
sim <- cb.sims.sim_linear(a=-1, n=100, err=1/8, unbalancedness=1.5)
cb.align.kway_match(sim$Ts, data.frame(Covar=sim$Xs), "Covar")
```

`cb.align.vm_trim` *Vector Matching*

Description

A function for implementing the vector matching procedure, a pre-processing step for causal conditional distance correlation. Uses propensity scores to strategically include/exclude samples from subsequent inference, based on whether (or not) there are samples with similar propensity scores across all treatment levels (conceptually, a k-way "propensity trimming"). It is imperative that this function is used in conjunction with domain expertise to ensure that the covariates are not colliders, and that the system satisfies the strong ignorability condiiton to derive causal conclusions.

Usage

```
cb.align.vm_trim(
  Ts,
  Xs,
  prop.form = NULL,
  retain.ratio = 0.05,
  ddx = FALSE,
  reference = NULL
)
```

Arguments

Ts	[n] the labels of the samples, with K < n levels, as a factor variable.
Xs	[n, r] the r covariates/confounding variables, for each of the n samples.
prop.form	a formula specifying a propensity scoring model. Defaults o NULL, which uses all of the covariates as exposure predictors.
retain.ratio	If the number of samples retained is less than retain.ratio*n, throws a warning. Defaults to 0.05.
ddx	whether to show additional diagnosis messages. Defaults to FALSE. Can help with debugging if unexpected results are obtained.
reference	the name of a reference label, against which to align other labels. Defaults to NULL, which identifies a shared region of covariate overlap across all labels..

Value

a [m] vector containing the indices of samples retained after vector matching.

Details

For more details see the help vignette: `vignette("causal_balancing", package = "causalBatch")`

Author(s)

Eric W. Bridgeford

References

Michael J. Lopez, et al. "Estimation of Causal Effects with Multiple Treatments" Statistical Science (2017). ran

Examples

```
library(causalBatch)
sim <- cb.sims.sim_linear(a=-1, n=100, err=1/8, unbalancedness=3)
cb.align.vm_trim(sim$Ts, sim$Xs)
```

`cb.correct.aipw_cComBat` *Augmented Inverse Probability Weighting Conditional ComBat*

Description

A function for implementing the AIPW conditional ComBat (AIPW cComBat) algorithm. This algorithm allows users to remove batch effects (in each dimension), while adjusting for known confounding variables. It is imperative that this function is used in conjunction with domain expertise (e.g., to ensure that the covariates are not colliders, and that the system could be argued to satisfy the ignorability condition) to derive causal conclusions. See citation for more details as to the conditions under which conclusions derived are causal.

Usage

```
cb.correct.aipw_cComBat(
  Ys,
  Ts,
  Xs,
  aipw.form,
  covar.out.form = NULL,
  retain.ratio = 0.05
)
```

Arguments

`Ys`	an `[n, d]` matrix, for the outcome variables with `n` samples in `d` dimensions.
`Ts`	`[n]` the labels of the samples, with at most two unique batches.
`Xs`	`[n, r]` the `r` covariates/confounding variables, for each of the `n` samples, as a data frame with named columns.
`aipw.form`	A covariate model, given as a formula. Applies for the estimation of propensities for the AIPW step.
`covar.out.form`	A covariate model, given as a formula. Applies for the outcome regression step of the `ComBat` algorithm. Defaults to `NULL`, which re-uses `aipw.form` for the covariate/outcome model.
`retain.ratio`	If the number of samples retained is less than `retain.ratio*n`, throws a warning. Defaults to `0.05`.

Details

Note: This function is experimental, and has not been tested on real data. It has only been tested with simulated data with binary (0 or 1) exposures.

Value

a list, containing the following:

- `Ys.corrected` an `[m, d]` matrix, for the `m` retained samples in `d` dimensions, after correction.
- `Ts` `[m]` the labels of the `m` retained samples, with `K < n` levels.
- `Xs` the `r` covariates/confounding variables for each of the `m` retained samples.
- `Model` the fit batch effect correction model.
- `Corrected.Ids` the ids to which batch effect correction was applied.

Details

For more details see the help vignette: `vignette("causal_ccombat", package = "causalBatch")`

Author(s)

Eric W. Bridgeford

References

Eric W. Bridgeford, et al. "A Causal Perspective for Batch Effects: When is no answer better than a wrong answer?" Biorxiv (2024).

W Evan Johnson, et al. "Adjusting batch effects in microarray expression data using empirical Bayes methods" Biostatistics (2007).

Examples

```
library(causalBatch)
sim <- cb.sims.sim_linear(a=-1, n=100, err=1/8, unbalancedness=2)
cb.correct.aipw_cComBat(sim$Ys, sim$Ts, data.frame(Covar=sim$Xs), "Covar")
```

---

cb.correct.apply_aipw_cComBat

Fit AIPW ComBat model for batch effect correction

Description

This function applies an Augmented Inverse Probability Weighting (AIPW) ComBat model for batch effect correction to new data.

Usage

```
cb.correct.apply_aipw_cComBat(Ys, Ts, Xs, Model)
```

Arguments

Ys	an [n, d] matrix, for the outcome variables with n samples in d dimensions.
Ts	[n] the labels of the samples, with K < n levels, as a factor variable.
Xs	[n, r] the r covariates/confounding variables, for each of the n samples, as a data frame with named columns.
Model	a list containing the following parameters: • Prop_model Fitted propensity model • Putcome_models List of fitted outcome models for each feature and treatment level • Levels Levels of the treatment factor • Treatment_effects Estimated treatment effects • AIPW_est AIPW estimator results • covar.out.form Formula used for the outcome model This model is output after fitting with cb.correct.aipw_cComBat.

Details

Note: This function is experimental, and has not been tested on real data. It has only been tested with simulated data with binary (0 or 1) exposures.

Value

an [n, d] matrix, the batch-effect corrected data.

Examples

```
library(causalBatch)
sim <- cb.sims.sim_linear(a=-1, n=200, err=1/8, unbalancedness=3)
# fit batch effect correction for first 100 samples
cb.fit <- cb.correct.matching_cComBat(sim$Ys[1:100,,drop=FALSE], sim$Ts[1:100],
                                      data.frame(Covar=sim$Xs[1:100,,drop=FALSE]), "Covar")
# apply to all samples
cor.dat <- cb.correct.apply_cComBat(sim$Ys, sim$Ts, data.frame(Covar=sim$Xs), cb.fit$Model)
```

---

cb.correct.apply_cComBat

Adjust for batch effects using an empirical Bayes framework

Description

ComBat allows users to adjust for batch effects in datasets where the batch covariate is known, using methodology described in Johnson et al. 2007. It uses either parametric or non-parametric empirical Bayes frameworks for adjusting data for batch effects. Users are returned an expression matrix that has been corrected for batch effects. The input data are assumed to be cleaned and normalized before batch effect removal.

Usage

```
cb.correct.apply_cComBat(Ys, Ts, Xs, Model)
```

Arguments

Ys	an [n, d] matrix, for the outcome variables with n samples in d dimensions.
Ts	[n] the labels of the samples, with K < n levels, as a factor variable.
Xs	[n, r] the r covariates/confounding variables, for each of the n samples, as a data frame with named columns.
Model	a list containing the following parameters: • Var the pooled variance • Grand.mean the overall mean of the data • B.hat the fit regression coefficients • Gamma additive batch effects • Delta multiplicative batch effects • Levels the order of levels for each batch • Covar.Mod the covariate model for adjustment This model is output after fitting with cb.correct.matching_cComBat.

Details

Note: this code is adapted directly from the ComBat algorithm featured in the 'sva' package.

Value

an [n, d] matrix, the batch-effect corrected data.

Examples

```
library(causalBatch)
sim <- cb.sims.sim_linear(a=-1, n=200, err=1/8, unbalancedness=3)
# fit batch effect correction for first 100 samples
cb.fit <- cb.correct.matching_cComBat(sim$Ys[1:100,,drop=FALSE], sim$Ts[1:100],
                                      data.frame(Covar=sim$Xs[1:100,,drop=FALSE]), "Covar")
# apply to all samples
cor.dat <- cb.correct.apply_cComBat(sim$Ys, sim$Ts, data.frame(Covar=sim$Xs), cb.fit$Model)
```

`cb.correct.matching_cComBat` *Matching Conditional ComBat*

Description

A function for implementing the matching conditional ComBat (matching cComBat) algorithm. This algorithm allows users to remove batch effects (in each dimension), while adjusting for known confounding variables. It is imperative that this function is used in conjunction with domain expertise (e.g., to ensure that the covariates are not colliders, and that the system could be argued to satisfy the ignorability condition) to derive causal conclusions. See citation for more details as to the conditions under which conclusions derived are causal.

Usage

```
cb.correct.matching_cComBat(
  Ys,
  Ts,
  Xs,
  match.form,
  covar.out.form = NULL,
  prop.form = NULL,
  reference = NULL,
 match.args = list(method = "nearest", exact = NULL, replace = FALSE, caliper = 0.1),
  retain.ratio = 0.05,
  apply.oos = FALSE
)
```

Arguments

`Ys`	an `[n, d]` matrix, for the outcome variables with `n` samples in `d` dimensions.
`Ts`	`[n]` the labels of the samples, with `K < n` levels, as a factor variable.
`Xs`	`[n, r]` the `r` covariates/confounding variables, for each of the `n` samples, as a data frame with named columns.
`match.form`	A formula of columns from `Xs`, to be passed directly to `matchit` for subsequent matching. See `formula` argument from `matchit` for details.
`covar.out.form`	A covariate model, given as a formula. Applies for the outcome regression step of the `ComBat` algorithm. Defaults to `NULL`, which re-uses `match.form` for the covariate/outcome model.
`prop.form`	A propensity model, given as a formula. Applies for the estimation of propensities for the propensity trimming step. Defaults to `NULL`, which re-uses `match.form` for the covariate/outcome model.
`reference`	the name of the reference/control batch, against which to match. Defaults to `NULL`, which treats the reference batch as the smallest batch.

match.args	A named list arguments for the matchit function, to be used to specify specific matching strategies, where the list names are arguments and the corresponding values the value to be passed to matchit. Defaults to inexact nearest-neighbor caliper (width 0.1) matching without replacement.
retain.ratio	If the number of samples retained is less than retain.ratio*n, throws a warning. Defaults to 0.05.
apply.oos	A boolean that indicates whether or not to apply the learned batch effect correction to non-matched samples that are still within a region of covariate support. Defaults to FALSE.

Value

a list, containing the following:

- Ys.corrected an [m, d] matrix, for the m retained samples in d dimensions, after correction.
- Ts [m] the labels of the m retained samples, with K < n levels.
- Xs the r covariates/confounding variables for each of the m retained samples.
- Model the fit batch effect correction model. See ComBat for details.
- InSample.Ids the ids which were used to fit the batch effect correction model.
- Corrected.Ids the ids to which batch effect correction was applied. Differs from InSample.Ids if apply.oos is TRUE.

Details

For more details see the help vignette: vignette("causal_ccombat", package = "causalBatch")

Author(s)

Eric W. Bridgeford

References

Eric W. Bridgeford, et al. "A Causal Perspective for Batch Effects: When is no answer better than a wrong answer?" Biorxiv (2024).

Daniel E. Ho, et al. "MatchIt: Nonparametric Preprocessing for Parametric Causal Inference" JSS (2011).

W Evan Johnson, et al. "Adjusting batch effects in microarray expression data using empirical Bayes methods" Biostatistics (2007).

Leek JT, Johnson WE, Parker HS, Fertig EJ, Jaffe AE, Zhang Y, Storey JD, Torres LC (2024). sva: Surrogate Variable Analysis. R package version 3.52.0.

Examples

```
library(causalBatch)
sim <- cb.sims.sim_linear(a=-1, n=100, err=1/8, unbalancedness=2)
cb.correct.matching_cComBat(sim$Ys, sim$Ts, data.frame(Covar=sim$Xs), "Covar")
```

`cb.detect.caus_cdcorr` *Causal Conditional Distance Correlation*

Description

A function for implementing the causal conditional distance correlation (causal cDCorr) algorithm. This algorithm allows users to identify whether a treatment causes changes in an outcome, given assorted covariates/confounding variables. It is imperative that this function is used in conjunction with domain expertise (e.g., to ensure that the covariates are not colliders, and that the system satisfies the strong ignorability condiiton) to derive causal conclusions. See citation for more details as to the conditions under which conclusions derived are causal.

Usage

```
cb.detect.caus_cdcorr(
  Ys,
  Ts,
  Xs,
  prop.form = NULL,
  R = 1000,
  dist.method = "euclidean",
  distance = FALSE,
  seed = 1,
  num.threads = 1,
  retain.ratio = 0.05,
  ddx = FALSE
)
```

Arguments

`Ys`	Either: • `[n, d]` matrix the outcome variables with `n` samples in `d` dimensions. In this case, `distance` should be `FALSE`. • `[n, n]` `dist` object a distance object for the `n` samples. In this case, `distance` should be `TRUE`.
`Ts`	`[n]` the labels of the samples, with `K < n` levels, as a factor variable.
`Xs`	`[n, r]` the `r` covariates/confounding variables, for each of the `n` samples.
`prop.form`	a formula specifying a propensity scoring model. Defaults o `NULL`, which uses all of the covariates as exposure predictors.
`R`	the number of repetitions for permutation testing. Defaults to `1000`.
`dist.method`	the method used for computing distance matrices. Defaults to `"euclidean"`. Other options can be identified by seeing the appropriate documention for the `method` argument for the `dist` function.
`distance`	a boolean for whether (or not) `Ys` are already distance matrices. Defaults to `FALSE`, which will use `dist.method` parameter to compute an `[n, n]` pairwise distance matrix for `Ys`.

seed	a random seed to set. Defaults to 1.
num.threads	The number of threads for parallel processing (if desired). Defaults to 1.
retain.ratio	If the number of samples retained is less than retain.ratio*n, throws a warning. Defaults to 0.05.
ddx	whether to show additional diagnosis messages. Defaults to FALSE. Can help with debugging if unexpected results are obtained.

Value

a list, containing the following:

- Test The outcome of the statistical test, from cdcov.test.
- Retained.Ids The sample indices retained after vertex matching, which correspond to the samples for which statistical inference is performed.

Details

For more details see the help vignette: vignette("causal_cdcorr", package = "causalBatch")

Author(s)

Eric W. Bridgeford

References

Eric W. Bridgeford, et al. "A Causal Perspective for Batch Effects: When is no answer better than a wrong answer?" Biorxiv (2024).

Eric W. Bridgeford, et al. "Learning sources of variability from high-dimensional observational studies" arXiv (2023).

Xueqin Wang, et al. "Conditional Distance Correlation" American Statistical Association (2015).

Examples

```
library(causalBatch)
sim <- cb.sims.sim_linear(a=-1, n=100, err=1/8, unbalancedness=3)
cb.detect.caus_cdcorr(sim$Ys, sim$Ts, sim$Xs)
```

`cb.sims.sim_impulse` *Impulse Simulation*

Description

Impulse Simulation

Usage

```
cb.sims.sim_impulse(
  n = 100,
  pi = 0.5,
  eff_sz = 1,
  alpha = 2,
  unbalancedness = 1,
  err = 1/2,
  null = FALSE,
  a = -0.5,
  b = 1/2,
  c = 4,
  nbreaks = 200
)
```

Arguments

`n`	the number of samples. Defaults to `100`.
`pi`	the balance between the classes, where samples will be from group 1 with probability `pi`, and group 2 with probability `1 - pi`. Defaults to `0.5`.
`eff_sz`	the treatment effect between the different groups. Defaults to `1`.
`alpha`	the alpha for the covariate sampling procedure. Defaults to `2`.
`unbalancedness`	the level of covariate dissimilarity between the covariates for each of the groups. Defaults to `1`.
`err`	the level of noise for the simulation. Defaults to `1/2`.
`null`	whether to generate a null simulation. Defaults to `FALSE`. Same behavior can be achieved by setting `eff_sz = 0`.
`a`	the first parameter for the covariate/outcome relationship. Defaults to `-0.5`.
`b`	the second parameter for the covariate/outcome relationship. Defaults to `1/2`.
`c`	the third parameter for the covariate/outcome relationship. Defaults to `1`.
`nbreaks`	the number of breakpoints for computing the expected outcome at a given covariate level for each batch. Defaults to `200`.

Value

a list, containing the following:

`Ys`	an `[n, 2]` matrix, containing the outcomes for each sample. The first dimension contains the "treatment effect".
`Ts`	an `[n, 1]` matrix, containing the group/batch labels for each sample.
`Xs`	an `[n, 1]` matrix, containing the covariate values for each sample.
`Eps`	an `[n, 1]` matrix, containing the error for each sample.
`x.bounds`	the theoretical bounds for the covariate values.
`Ytrue`	an `[nbreaks*2, 2]` matrix, containing the expected outcomes at a covariate level indicated by `Xtrue`.
`Ttrue`	an `[nbreaks*2,1]` matrix, indicating the group/batch the expected outcomes and covariate breakpoints correspond to.
`Xtrue`	an `[nbreaks*2, 1]` matrix, indicating the values of the covariate breakpoints for the theoretical expected outcome in `Ytrue`.
`Effect`	The batch effect magnitude.
`Overlap`	the theoretical degree of overlap between the covariate distributions for each of the two groups/batches.
`oracle_fn`	A function for fitting outcomes given covariates.

Details

A sigmoidal relationship between the covariate and the outcome. The first dimension of the outcome is:

$$Y_i = c \times \phi(X_i, \mu = a, \sigma = b) - \text{eff\_sz} \times T_i + \frac{1}{2}\epsilon_i$$

where $\phi(x, \mu, \sigma)$ is the probability density function for the normal distribution with mean μ and standard deviation σ.

where the batch/group labels are:

$$T_i \overset{iid}{\sim} Bern(\pi)$$

The beta coefficient for the covariate sampling is:

$$\beta = \alpha \times \text{unbalancedness}$$

The covariate values for the first batch are:

$$X_i|T_i = 0 \overset{ind}{\sim} 2Beta(\alpha, \beta) - 1$$

and the covariate values for the second batch are:

$$X_i|T_i = 1 \overset{ind}{\sim} 2Beta(\beta, \alpha) - 1$$

Note that $X_i|T_i = 0 \overset{D}{=} -X_i|T_i = 1$, or that the covariates are symmetric about the origin in distribution.

Finally, the error terms are:

$$\epsilon_i \overset{iid}{\sim} Norm(0, \text{err}^2)$$

For more details see the help vignette: `vignette("causal_simulations", package = "causalBatch")`

Author(s)

Eric W. Bridgeford

References

Eric W. Bridgeford, et al. "A Causal Perspective for Batch Effects: When is no answer better than a wrong answer?" Biorxiv (2024).

Examples

```
library(causalBatch)
sim = cb.sims.sim_impulse()
```

---

cb.sims.sim_impulse_asycov

Impulse Simulation with Asymmetric Covariates

Description

Impulse Simulation with Asymmetric Covariates

Usage

```
cb.sims.sim_impulse_asycov(
  n = 100,
  pi = 0.5,
  eff_sz = 1,
  alpha = 2,
  unbalancedness = 1,
  null = FALSE,
  a = -0.5,
  b = 1/2,
  c = 4,
  err = 1/2,
  nbreaks = 200
)
```

Arguments

`n`	the number of samples. Defaults to `100`.
`pi`	the balance between the classes, where samples will be from group 1 with probability `pi`, and group 2 with probability `1 - pi`. Defaults to `0.5`.
`eff_sz`	the treatment effect between the different groups. Defaults to `1`.
`alpha`	the alpha for the covariate sampling procedure. Defaults to `2`.

unbalancedness	the level of covariate dissimilarity between the covariates for each of the groups. Defaults to 1.
null	whether to generate a null simulation. Defaults to FALSE. Same behavior can be achieved by setting eff_sz = 0.
a	the first parameter for the covariate/outcome relationship. Defaults to -0.5.
b	the second parameter for the covariate/outcome relationship. Defaults to 1/2.
c	the third parameter for the covariate/outcome relationship. Defaults to 1.
err	the level of noise for the simulation. Defaults to 1/2.
nbreaks	the number of breakpoints for computing the expected outcome at a given covariate level for each batch. Defaults to 200.

Value

a list, containing the following:

Ys	an [n, 2] matrix, containing the outcomes for each sample. The first dimension contains the "treatment effect".
Ts	an [n, 1] matrix, containing the group/batch labels for each sample.
Xs	an [n, 1] matrix, containing the covariate values for each sample.
Eps	an [n, 1] matrix, containing the error for each sample.
x.bounds	the theoretical bounds for the covariate values.
Ytrue	an [nbreaks*2, 2] matrix, containing the expected outcomes at a covariate level indicated by Xtrue.
Ttrue	an [nbreaks*2,1] matrix, indicating the group/batch the expected outcomes and covariate breakpoints correspond to.
Xtrue	an [nbreaks*2, 1] matrix, indicating the values of the covariate breakpoints for the theoretical expected outcome in Ytrue.
Effect	The batch effect magnitude.
Overlap	the theoretical degree of overlap between the covariate distributions for each of the two groups/batches.
oracle_fn	A function for fitting outcomes given covariates.

Details

A sigmoidal relationship between the covariate and the outcome. The first dimension of the outcome is:

$$Y_i = c \times \phi(X_i, \mu = a, \sigma = b) - \text{eff\_sz} \times T_i + \frac{1}{2}\epsilon_i$$

where $\phi(x, \mu, \sigma)$ is the probability density function for the normal distribution with mean μ and standard deviation σ.

where the batch/group labels are:

$$T_i \overset{iid}{\sim} Bern(\pi)$$

The beta coefficient for the covariate sampling is:

$$\beta = \alpha \times \text{unbalancedness}$$

The covariate values for the first batch are asymmetric, in that for the first batch:

$$X_i|T_i = 0 \overset{ind}{\sim} 2Beta(\alpha, \alpha) - 1$$

and the covariate values for the second batch are:

$$X_i|T_i = 1 \overset{ind}{\sim} 2Beta(\beta, \alpha) - 1$$

Finally, the error terms are:

$$\epsilon_i \overset{iid}{\sim} Norm(0, \text{err}^2)$$

For more details see the help vignette: `vignette("causal_simulations", package = "causalBatch")`

Author(s)

Eric W. Bridgeford

References

Eric W. Bridgeford, et al. "A Causal Perspective for Batch Effects: When is no answer better than a wrong answer?" Biorxiv (2024).

Examples

```
library(causalBatch)
sim = cb.sims.sim_impulse_asycov()
```

`cb.sims.sim_linear` *Linear Simulation*

Description

Linear Simulation

Usage

```
cb.sims.sim_linear(
  n = 100,
  pi = 0.5,
  eff_sz = 1,
  alpha = 2,
  unbalancedness = 1,
  err = 1/2,
  null = FALSE,
  a = -2,
  b = -1,
  nbreaks = 200
)
```

Arguments

n	the number of samples. Defaults to 100.
pi	the balance between the classes, where samples will be from group 1 with probability pi, and group 2 with probability 1 - pi. Defaults to 0.5.
eff_sz	the treatment effect between the different groups. Defaults to 1.
alpha	the alpha for the covariate sampling procedure. Defaults to 2.
unbalancedness	the level of covariate dissimilarity between the covariates for each of the groups. Defaults to 1.
err	the level of noise for the simulation. Defaults to 1/2.
null	whether to generate a null simulation. Defaults to FALSE. Same behavior can be achieved by setting eff_sz = 0.
a	the first parameter for the covariate/outcome relationship. Defaults to -2.
b	the second parameter for the covariate/outcome relationship. Defaults to -1.
nbreaks	the number of breakpoints for computing the expected outcome at a given covariate level for each batch. Defaults to 200.

Value

a list, containing the following:

Ys	an [n, 2] matrix, containing the outcomes for each sample. The first dimension contains the "treatment effect".
Ts	an [n, 1] matrix, containing the group/batch labels for each sample.
Xs	an [n, 1] matrix, containing the covariate values for each sample.
Eps	an [n, 1] matrix, containing the error for each sample.
x.bounds	the theoretical bounds for the covariate values.
Ytrue	an [nbreaks*2, 2] matrix, containing the expected outcomes at a covariate level indicated by Xtrue.
Ttrue	an [nbreaks*2,1] matrix, indicating the group/batch the expected outcomes and covariate breakpoints correspond to.
Xtrue	an [nbreaks*2, 1] matrix, indicating the values of the covariate breakpoints for the theoretical expected outcome in Ytrue.
Effect	The batch effect magnitude.
Overlap	the theoretical degree of overlap between the covariate distributions for each of the two groups/batches.
oracle_fn	A function for fitting outcomes given covariates.

Details

A linear relationship between the covariate and the outcome. The first dimension of the outcome is:

$$Y_i = a \times (X_i + b) - \text{eff\_sz} \times T_i + \frac{1}{2}\epsilon_i$$

where the batch/group labels are:

$$T_i \overset{iid}{\sim} Bern(\pi)$$

The beta coefficient for the covariate sampling is:

$$\beta = \alpha \times \text{unbalancedness}$$

The covariate values for the first batch are:

$$X_i|T_i = 0 \overset{ind}{\sim} 2Beta(\alpha, \beta) - 1$$

and the covariate values for the second batch are:

$$X_i|T_i = 1 \overset{ind}{\sim} 2Beta(\beta, \alpha) - 1$$

Finally, the error terms are:

$$\epsilon_i \overset{iid}{\sim} Norm(0, \text{err}^2)$$

For more details see the help vignette: `vignette("causal_simulations", package = "causalBatch")`

Author(s)

Eric W. Bridgeford

References

Eric W. Bridgeford, et al. "A Causal Perspective for Batch Effects: When is no answer better than a wrong answer?" Biorxiv (2024).

Examples

```
library(causalBatch)
sim = cb.sims.sim_linear()
```

`cb.sims.sim_sigmoid` *Sigmoidal Simulation*

Description

Sigmoidal Simulation

Usage

```
cb.sims.sim_sigmoid(
  n = 100,
  pi = 0.5,
  eff_sz = 1,
  alpha = 2,
  unbalancedness = 1,
  null = FALSE,
  a = -4,
  b = 8,
  err = 1/2,
  nbreaks = 200
)
```

Arguments

n	the number of samples. Defaults to 100.
pi	the balance between the classes, where samples will be from group 1 with probability pi, and group 2 with probability 1 - pi. Defaults to 0.5.
eff_sz	the treatment effect between the different groups. Defaults to 1.
alpha	the alpha for the covariate sampling procedure. Defaults to 2.
unbalancedness	the level of covariate dissimilarity between the covariates for each of the groups. Defaults to 1.
null	whether to generate a null simulation. Defaults to FALSE. Same behavior can be achieved by setting eff_sz = 0.
a	the first parameter for the covariate/outcome relationship. Defaults to -4.
b	the second parameter for the covariate/outcome relationship. Defaults to 8.
err	the level of noise for the simulation. Defaults to 1/2.
nbreaks	the number of breakpoints for computing the expected outcome at a given covariate level for each batch. Defaults to 200.

Value

a list, containing the following:

Y	an [n, 2] matrix, containing the outcomes for each sample. The first dimension contains the "treatment effect".
Ts	an [n, 1] matrix, containing the group/batch labels for each sample.
Xs	an [n, 1] matrix, containing the covariate values for each sample.
Eps	an [n, 1] matrix, containing the error for each sample.
x.bounds	the theoretical bounds for the covariate values.
Ytrue	an [nbreaks*2, 2] matrix, containing the expected outcomes at a covariate level indicated by Xtrue.
Ttrue	an [nbreaks*2,1] matrix, indicating the group/batch the expected outcomes and covariate breakpoints correspond to.

`Xtrue`	an `[nbreaks*2, 1]` matrix, indicating the values of the covariate breakpoints for the theoretical expected outcome in `Ytrue`.
`Effect`	The batch effect magnitude.
`Overlap`	the theoretical degree of overlap between the covariate distributions for each of the two groups/batches.
`oracle_fn`	A function for fitting outcomes given covariates.

Details

A sigmoidal relationship between the covariate and the outcome. The first dimension of the outcome is:

$$Y_i = a \times \text{sigmoid}(b \times X_i) - a - \text{eff\_sz} \times T_i + \frac{1}{2}\epsilon_i$$

where the batch/group labels are:

$$T_i \overset{iid}{\sim} Bern(\pi)$$

The beta coefficient for the covariate sampling is:

$$\beta = \alpha \times \text{unbalancedness}$$

The covariate values for the first batch are:

$$X_i | T_i = 0 \overset{ind}{\sim} 2Beta(\alpha, \beta) - 1$$

and the covariate values for the second batch are:

$$X_i | T_i = 1 \overset{ind}{\sim} 2Beta(\beta, \alpha) - 1$$

Finally, the error terms are:

$$\epsilon_i \overset{iid}{\sim} Norm(0, \text{err}^2)$$

For more details see the help vignette: `vignette("causal_simulations", package = "causalBatch")`

Author(s)

Eric W. Bridgeford

References

Eric W. Bridgeford, et al. "A Causal Perspective for Batch Effects: When is no answer better than a wrong answer?" Biorxiv (2024).

Examples

```
library(causalBatch)
sim = cb.sims.sim_sigmoid()
```

Index