Package: inferference (via r-universe)

November 4, 2024

Type Package

Title Methods for Causal Inference with Interference

Version 1.0.2

Date 2021-04-21

Author Bradley Saul

Maintainer Bradley Saul <bradleysaul@gmail.com>

Description Provides methods for estimating causal effects in the presence of interference described in B. Saul and M. Hugdens (2017) <doi:10.18637/jss.v082.i02>. Currently it implements the inverse-probability weighted (IPW) estimators proposed by E.J. Tchetgen Tchetgen and T.J. Vanderweele (2012) <doi:10.1177/0962280210386779>.

Depends R (>= 3.1)

Imports numDeriv (>= 2012.9-1), lme4 (>= 1.1-6), Formula (>= 1.1-2), methods, spaMM

License GPL (>= 2)

Suggests testthat, knitr, markdown

VignetteBuilder knitr

LazyData true

RoxygenNote 7.1.1

Encoding UTF-8

Repository https://bsaul.r-universe.dev

RemoteUrl https://github.com/bsaul/inferference

RemoteRef HEAD

RemoteSha 6bb643ba8b45d7816891d05cb8dee31591c8bf3f

Contents

`diagnose_weights` *Plot histograms of weights from an interference object*

Description

Plot histograms of weights from an interference object

Usage

```
diagnose_weights(obj, allocations = NULL, ...)
```

Arguments

`obj`	an `interference` object
`allocations`	optional numeric vector of allocations for which to print histogram. If NULL (the default), five allocations selected evenly from the first allocation to the last are printed.
`...`	additional arguments passed to `hist`

Value

histogram of group-level weights

direct_effect *Retreive Direct Effect estimates*

Description

Retrieves the population average direct causal effect for a specified allocation: $\hat{Y}(0, alpha) - \hat{Y}(1, alpha)$.

Usage

```
direct_effect(object, allocation = NULL, trt.lvl1 = 0)
```

Arguments

object	an object of class interference
allocation	the allocation scheme for which to estimate direct effects. If NULL, then returns all direct effects.
trt.lvl1	Defaults to 0.

Value

a data.frame with requested values

get_args *Get arguments from a function*

Description

Extracts the names of the arguments from a function, and creates a list of those arguments where they exist in ... .

Usage

```
get_args(FUN, args_list = NULL, ...)
```

Arguments

FUN	function for which to find arguments
args_list	a list of arguments. Defaults to NULL.
...	any arguments. Those necessary for FUN must be named as appropriate for FUN

Value

list of arguments for FUN

Examples

```
myargs <- get_args(lm, formula = Sepal.Length ~ Sepal.Width, data = iris )
summary(do.call('lm', myargs))
```

`indirect_effect` *Retreive Indirect Effect estimates*

Description

Retrieves the population average indirect causal effect for specified allocations: $\hat{Y}(0, alpha1) - \hat{Y}(0, alpha2)$. This is the effect due to the coverage (allocation) levels.

Usage

```
indirect_effect(object, allocation1, allocation2 = NULL, trt.lvl = 0)

ie(object, allocation1, allocation2 = NULL, trt.lvl = 0)
```

Arguments

`object`	an object of class `interference`
`allocation1`	the allocation scheme for which to estimate indirect effects
`allocation2`	the allocation scheme for which to estimate indirect effects. If NULL, then returns all indirect effects compared to allocation1.
`trt.lvl`	Defaults to 0.

Value

a data.frame with requested values

`inferference` *Methods for causal inference with interference*

Description

Interference occurs when the treatment of one unit affects outcomes of other units. This package provides methods for estimating causal effects in the presence of interference. Currently it implements the IPW estimators proposed by Tchetgen Tchetgen and Vanderweele (2012) (doi:10.1177/0962280210386779) and developed further in Heydrich-Perez et al. (2014) (doi:10.1111/biom.12184).

References

Saul, B. and Hugdens, M. G. (2017). A Recipe for inferference: Start with Causal Inference. Add Interference. Mix Well with R. Journal of Statistical Software, 82(2), 1-21. doi:10.18637/jss.v082.i02

`interference` *Estimate Causal Effects in presence of interference*

Description

Estimate Causal Effects in presence of interference

Usage

```
interference(
  formula,
  propensity_integrand = "logit_integrand",
  loglihood_integrand = propensity_integrand,
  allocations,
  data,
  model_method = "glmer",
  model_options = list(family = stats::binomial(link = "logit")),
  causal_estimation_method = "ipw",
  causal_estimation_options = list(variance_estimation = "robust"),
  conf.level = 0.95,
  rescale.factor = 1,
  integrate_allocation = TRUE,
  runSilent = TRUE,
  ...
)
```

Arguments

`formula` The formula used to define the causal model. Has a minimum of 4 parts, separated by `|` and `~` in a specific structure: `outcome | exposure ~ propensity covariates | group`. The order matters, and the pipes split the data frame into corresponding pieces. The part separated by `~` is passed to the chosen `model_method` used to estimate or fix propensity parameters.

`propensity_integrand`

A function, which may be created by the user, used to compute the IP weights. This defaults to `logit_integrand`, which calculates the product of inverse logits for individuals in a group: $\prod_{j=1}^{n_i}\{r \times h_{ij}(b_i)^{A_{ij}}\}\{1-r \times h_{ij}(b_i)\}^{1-A_{ij}} f_b(b_i;\theta_s)$ where

$$h_{ij}(b_i) = logit^{-1}(\mathbf{X}_{ij}\theta_a + b_i)$$

and b_i is a group-level random effect, f_b is a $N(0,\theta_s)$ density, and r is a known randomization probability which may be useful if a participation vector is included in the `formula`. If no random effect was included in the `formula`, `logit_integrand` essentially ignores the random effect and $f_b(b_i,\theta_s)$ integrates to 1. See details for arguments that can be passed to `logit_integrand`

loglihood_integrand
: A function, which may be created by the user, that defines the log likelihood of the logit model used for robust variance estimation. Generally, this will be the same function as propensity_integrand. Indeed, this is the default.

allocations
: a vector of values in (0, 1). Increasing the number of elements of the allocation vector greatly increases computation time; however, a larger number of allocations will make plots look nicer. A minimum of two allocations is required.

data
: the analysis data frame. This must include all the variables defined in the formula.

model_method
: the method used to estimate or set the propensity model parameters. Must be one of 'glm', 'glmer', or 'oracle'. Defaults to 'glmer'. For a fixed effects only model use 'glm', and to include random effects use 'glmer'. logit_integrand only supports a single random effect for the grouping variable, so if more random effects are included in the model, different propensity_integrand and loglihood_integrand functions should be defined. When the propensity parameters are known (as in simulations) or if estimating parameters by other methods, use the 'oracle' option. See model_options for details on how to pass the oracle parameters.

model_options
: a list of options passed to the function in model_method. Defaults to list(family = binomial(link = 'logit')). When model_method = 'oracle', the list must have two elements (1) fixed_effects and (2) random_effects. If the model did not include random effects, set random.effects = NULL.

causal_estimation_method
: currently only supports 'ipw'.

causal_estimation_options
: A list. Current options are: (1) variance_estimation is either 'naive' or 'robust'. See details. Defaults to 'robust'.

conf.level
: level for confidence intervals. Defaults to 0.95.

rescale.factor
: a scalar multiplication factor by which to rescale outcomes and effects. Defaults to 1.

integrate_allocation
: Indicator of whether the integrand function uses the allocation parameter. Defaults to TRUE.

runSilent
: if FALSE, status of computations are printed to console. Defaults to TRUE.

...
: Used to pass additional arguments to internal functions such as numDeriv::grad() or integrate(). Additionally, arguments can be passed to the propensity_integrand and loglihood_integrand functions.

Details

The following formula includes a random effect for the group: outcome | exposure ~ propensity covariates + (1|group) | group. In this instance, the group variable appears twice. If the study design includes a "participation" variable, this is easily added to the formula: outcome | exposure | participation ~ propensity covariates | group.

logit_integrand has two options that can be passed via the ... argument:

- `randomization`: a scalar. This is the r in the formula just above. It defaults to 1 in the case that a `participation` vector is not included. The vaccine study example demonstrates use of this argument.
- `integrate_allocation`: `TRUE/FALSE`. When group sizes grow large (over 1000), the product term of `logit_integrand` tends quickly to 0. When set to `TRUE`, the IP weights tend less quickly to 0. Defaults to `FALSE`.

If the true propensity model is known (e.g. in simulations) use `variance_estimatation = 'naive'`; otherwise, use the default `variance_estimatation = 'robust'`. Refer to the web appendix of Heydrich-Perez et al. (2014) (doi:10.1111/biom.12184) for complete details.

Value

Returns a list of overall and group-level IPW point estimates, overall and group-level IPW point estimates (using the weight derivatives), derivatives of the loglihood, the computed weight matrix, the computed weight derivative array, and a summary.

References

Saul, B. and Hugdens, M. G. (2017). A Recipe for inferference: Start with Causal Inference. Add Interference. Mix Well with R. Journal of Statistical Software, 82(2), 1-21. doi:10.18637/jss.v082.i02

Perez-Heydrich, C., Hudgens, M. G., Halloran, M. E., Clemens, J. D., Ali, M., & Emch, M. E. (2014). Assessing effects of cholera vaccination in the presence of interference. Biometrics, 70(3), 731-741.

Tchetgen Tchetgen, E. J., & VanderWeele, T. J. (2012). On causal inference in the presence of interference. Statistical Methods in Medical Research, 21(1), 55-75.

`logit_integrand` *Default integrand for the group-level propensity score*

Description

Computes the following function:

$$\prod_{j=1}^{n}(rh_j(b))^{A_j}(1-rh_j(b))^{1-A_j}f_b(b;\theta_b)$$

where r is the randomization scheme. X is the covariate(s) vectors. $fixef$ is the vector of fixed effects. b is the random (group-level) effect. $ranef$ is the random effect variance.

Usage

```
logit_integrand(b, X, A, parameters, allocation = A, randomization = 1)
```

Arguments

b	vector argument of values necessary for integrate.
X	n by length(fixed effects) matrix of covariates.
A	vector of binary treatments
parameters	vector of fixed effect (and random effect if applicable). Random effect should be last element in vector.
allocation	The allocation strategy. Defaults to A so that is essentially ignored if allocation is not set to a value within (0, 1).
randomization	Randomization probability. Defaults to 1.

Value

value of the integrand

---

log_likelihood *Log Likelihood*

Description

Used by score_matrix to compute the log likelihood.

Usage

```
log_likelihood(parameters, integrand, ...)
```

Arguments

parameters	vector of parameters passed to integrand
integrand	Defaults to logit_integrand
...	additional arguments passed to integrand function.

Value

value of log likelihood

overall_effect *Retrieve Overall Effect Estimates*

Description

Retrieves the population average overall causal effect: $\hat{Y}(alpha1) - \hat{Y}(lpha2)$

Usage

```
overall_effect(object, allocation1, allocation2 = NULL)

oe(object, allocation1, allocation2 = NULL)
```

Arguments

object	an object of class interference
allocation1	the allocation scheme for which to estimate overall effects
allocation2	the allocation scheme for which to estimate overall effects

Value

a data.frame with a single row with requested values

print.interference *Prints a summary of an interference object*

Description

Prints a summary of an interference object

Usage

```
## S3 method for class 'interference'
print(x, ...)
```

Arguments

x	object of class 'interference'
...	ignored

score_calc *Compute scores for a single group*

Description

Used by score_matrix to log likelihood derivatives for a single group.

Usage

```
score_calc(parameters, integrand, hide.errors = TRUE, ...)
```

Arguments

parameters	vector of parameters passed to integrand
integrand	function to used for the integrand. Defaults to logit_integrand.
hide.errors	Hide errors printed from grad. Defaults to true.
...	additional arguments pass to the integrand function.

Value

length(theta) vector of scores

score_matrix *Calculate matrix of log Likelihood derivatives*

Description

Calculate matrix of log Likelihood derivatives

Usage

```
score_matrix(integrand, X, A, G, parameters, runSilent = TRUE, ...)
```

Arguments

integrand	function passed to log_likelihood. Defaults to logit_integrand
X	covariate matrix
A	vector of treatment assignments
G	vector of group assignments
parameters	vector of parameters passed to integrand
runSilent	If FALSE, prints errors to console. Defaults to TRUE.
...	additional arguments passed to integrand or grad. For example, one can change the method argument in grad.

Value

N X length(params) matrix of scores

`total_effect` *Retrieve Total Effect estimates*

Description

Retrieves the population average total causal effect for specified allocations: $\hat{Y}(0, alpha1) - \hat{Y}(1, alpha2)$

Usage

```
total_effect(object, allocation1, allocation2 = NULL, trt.lvl1 = 0)

te(object, allocation1, allocation2 = NULL, trt.lvl1 = 0)
```

Arguments

`object`	an object of class `interference`
`allocation1`	the allocation scheme for which to estimate total effects
`allocation2`	the allocation scheme for which to estimate total effects If NULL, then returns all indirect effects compared to allocation1.
`trt.lvl1`	Defaults to 0.

Value

a data.frame with requested values

`vaccinesim` *Vaccine Study Sample Data*

Description

A sample dataset based on the simulations of a cholera vaccine trial in Heydrich-Perez et al. (2014) (doi:10.1111/biom.12184) except with 3000 individuals in 250 groups rather than 10000 in 500.

Format

a dataset with 6 variables and 3000 rows

- Ythe outcome (0 - no cholera; 1 - cholera)
- X1an individual's age (in decades)
- X2an individual's distance from river
- Aan indicator of vaccination (0 - no vaccine; 1 - vaccine)
- Ban indicator of participation (0 - did not participant in vaccine trial, 1 - did participate)
- groupgroup membership

References

Perez-Heydrich, C., Hudgens, M. G., Halloran, M. E., Clemens, J. D., Ali, M., & Emch, M. E. (2014). Assessing effects of cholera vaccination in the presence of interference. Biometrics, 70(3), 731-741.

`voters` *Voting Contagion Experiment Data*

Description

A dataset of a voting contagion experiment. See Nickerson (2008) for more details. The variables used in the package vignette are documented here.

Format

a dataset with 21 variables and 7722 rows

- familyhousehold ID
- denver1 = subject in Denver, 0 = Minneapolis
- treatment1 = voting encouragement, 2 = recycling message, 3 = not contacted
- reached1 = subject answered door, 0 = not
- hsecontact1 = household contacted by canvassers, 0 = not
- voted02p1 = voted in '02 primary, 0 = not
- partyparty affiliation
- ageage
- gendergender

References

Nickerson, D. W. (2008). Is voting contagious? Evidence from two field experiments. American Political Science Review, 102(01), 49-57. doi:10.1017/S0003055408080039

wght_calc *Compute IPW weight*

Description

Calculates the IPW for a single group. Used by `wght_matrix` to create a matrix of weights for each group and allocation scheme.

Usage

```
wght_calc(parameters, integrand, allocation, integrate_allocation = TRUE, ...)
```

Arguments

`parameters`	vector of parameter values
`integrand`	function to pass to the argument 'f' of `integrate`.
`allocation`	the allocation ratio for which to compute the weight
`integrate_allocation`	Indicator of whether the integrand function uses the allocation parameter. Defaults to TRUE.
`...`	other arguments passed to integrand.

Details

If `allocation` is an argument in the integrand function and `integrate_allocation == TRUE`, then the weight is calcuated as:

$$\frac{1}{Pr(A|X)}$$

Otherwise, the weight is computed by:

$$\frac{\prod_{j=1}^{n} \alpha_j^A (1-\alpha)^(1-A_j)}{Pr(A|X)}$$

Value

scalar result of the integral

wght_deriv_array *Create an array of group weight derivatives*

Description

Uses wght_deriv_calc to compute the weight derivatives for each group per coverage level

Usage

```
wght_deriv_array(
  parameters,
  integrand,
  allocations,
  X,
  A,
  G,
  runSilent = TRUE,
  integrate_allocation = TRUE,
  ...
)
```

Arguments

parameters	vector of parameters passed to integrand
integrand	function to pass to the argument 'f' of integrate.
allocations	coverage levels in [0, 1]. Can be vector.
X	covariate matrix
A	vector of treatment assignments
G	vector of group assignments
runSilent	if FALSE, errors are printed to console. Defaults to TRUE.
integrate_allocation	Indicator of whether the integrand function uses the allocation parameter. Defaults to TRUE.
...	other arguments passed to integrand.

Value

a length(unique(group)) X length(params) X length(alphas) array of group weight derivatives

`wght_deriv_calc` *Compute the derivative(s) of a weight*

Description

Takes the derivative of the `wght_calc` function with respect to each parameter in `params`.

Usage

```
wght_deriv_calc(
  integrand,
  parameters,
  allocation,
  integrate_allocation = TRUE,
  ...
)
```

Arguments

`integrand`	function to pass to the argument 'f' of `integrate`.
`parameters`	vector of parameter values
`allocation`	the allocation ratio for which to compute the weight
`integrate_allocation`	Indicator of whether the integrand function uses the allocation parameter. Defaults to TRUE.
`...`	other arguments passed to integrand.

Value

vector of derivatives with respect to element of params

`wght_matrix` *Creates a number of groups by number of allocation schemes matrix of group weights. Allocation schemes are selected by the user.*

Description

Groups should be numbered 1, ..., N

Usage

```
wght_matrix(
  integrand,
  allocations,
  X,
  A,
  G,
  parameters,
  runSilent = TRUE,
  integrate_allocation = TRUE,
  ...
)
```

Arguments

integrand	function to pass to the argument 'f' of integrate.
allocations	coverage levels in [0, 1]. Can be vector.
X	covariate matrix
A	vector of treatment assignments
G	vector of group assignments
parameters	vector of parameters passed to integrand
runSilent	if FALSE, errors are printed to console. Defaults to TRUE.
integrate_allocation	Indicator of whether the integrand function uses the allocation parameter. Defaults to TRUE.
...	other arguments passed to integrand.

Value

a length(unique(group)) X length(alphas) matrix of group weights

Index