Package ‘funmediation’

December 15, 2021

Type Package

Title Functional Mediation for a Distal Outcome

Version 1.0.1

Date 2021-12-13

VignetteBuilder knitr

Depends refund, tvem, boot

Suggests knitr, rmarkdown

Maintainer John J. Dziak <dziakj1@gmail.com>

Copyright 2021 by The Pennsylvania State University

Description Fits a functional mediation model with a scalar distal outcome. The method is described in detail by Coffman, Dziak, Litson, Chakraborti, Piper & Li (2021) <arXiv:2112.03960>. The model is similar to that of Lindquist (2012) <doi:10.1080/01621459.2012.695640> although allowing a binary outcome as an alternative to a numerical outcome. The development of this package was part of a research project supported by National Institutes of Health grants P50 DA039838 from the National Institute of Drug Abuse and 1R01 CA229542-01 from the National Cancer Institute and the NIH Office of Behavioral and Social Science Research. Content is solely the responsibility of the authors and does not necessarily represent the official views of the funding institutions mentioned above. This software is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation no

Author John J. Dziak [aut, cre] (<https://orcid.org/0000-0003-0762-5495>),
Donna L. Coffman [aut] (<https://orcid.org/0000-0001-6305-6579>),
Kaylee Litson [aut],
Yajnaseni Chakraborti [aut],
Runze Li [aut]

Repository CRAN

Date/Publication 2021-12-15 09:10:02 UTC

R topics documented:

`funmediation` *funmediation: Fit funmediation model*

Description

Calculate indirect effect of a binary treatment on a scalar response as mediated by a longitudinal functional trajectory (see Baron & Kenny, 1986; Lindquist, 2012; Coffman et al., 2021).

Usage

```
funmediation(
  data,
  treatment,
  mediator,
  outcome,
  id,
  time,
  tve_covariates_on_mediator = NULL,
  tie_covariates_on_mediator = NULL,
  covariates_on_outcome = NULL,
  interpolate = TRUE,
  tvem_penalize = TRUE,
  tvem_penalty_order = 1,
  tvem_spline_order = 3,
  tvem_num_knots = 3,
  tvem_do_loop = FALSE,
  tvem_use_bic = FALSE,
  binary_mediator = FALSE,
  binary_outcome = FALSE,
  nboot = 200,
  boot_level = 0.05,
  show_progress = FALSE
)
```

Arguments

`data` The dataset containing the data to be analyzed, in long format (one row per observation, multiple per individual).

treatment
: The name of the variable containing the treatment assignment, assumed to be unidimensional (either binary or else numeric). We recommend a binary (dichotomous) treatment with 0 for control and 1 for experimental). The values of this variable should be the same for each row for a given subject. If there are more than one treatment variables, such as a dummy-coded exposure with more than two levels, specify them as a formula such as ~x1+x2.

mediator
: The name of the mediator variable. The values of this variable can (and should) vary within each subject.

outcome
: The name of the outcome variable. The values of this variable should be the same for each row for a given subject.

id
: The name of the variable identifying each subject.

time
: The name of the time variable.

tve_covariates_on_mediator
: The covariates with time-varying-effects, if any, to be included in the model predicting the mediator from the treatment.

tie_covariates_on_mediator
: The covariates with time-invariant effects, if any, to be included in the model predicting the mediator from the treatment.

covariates_on_outcome
: The covariates, if any, to be included in the model predicting the outcome from the treatment. They are assumed to be subject-level (time-invariant both in value and in effect).

interpolate
: What kind of presmoothing to use in the penalized functional regression – specifically, whether to interpolate each subject's trajectory on the mediator (TRUE) or fit a spline to each subject's trajectory on the mediator (FALSE). This will be counted as TRUE if binary_mediator is TRUE because it does not make as much sense to interpolate a binary outcome.

tvem_penalize
: Input to be passed on to the tvem function

tvem_penalty_order
: Input to be passed on to the tvem function

tvem_spline_order
: Input to be passed on to the tvem function

tvem_num_knots
: If tvem_do_loop is FALSE, then tvem_num_knots is passed on to the tvem function as num_knots, an integer representing the number of interior knots per B-spline. If tvem_do_loop is TRUE then tvem_num_knots is reinterpreted as the highest number of interior knots to try.

tvem_do_loop
: Whether to use a loop to select the number of knots with a pseudo-AIC or pseudo-BIC, passed on to the tvem function

tvem_use_bic
: This parameter only matters if tvem_do_loop is TRUE. If tvem_do_loop is TRUE and tvem_use_bic is TRUE, then the information criterion used will be a pseudolikelihood version of BIC. If tvem_do_loop is TRUE and tvem_use_bic is FALSE, then the information criterion used will be a pseudolikelihood version of AIC instead. If tvem_do_loop is FALSE then tvem_use_bic is ignored.

binary_mediator
: Whether the mediator should be modeled as dichotomous with a logistic model (TRUE), or numerical with a normal model (FALSE).

`binary_outcome`	Whether the outcome should be modeled as dichotomous with a logistic model (TRUE), or numerical with a normal model (FALSE).
`nboot`	Number of bootstrap samples for bootstrap significance test of the overall effect. This test is done using the boot function from the boot package by Angelo Canty and Brian Ripley. It differs somewhat from the bootstrap approach used in a similar context by Lindquist (2012). We recommend using at least 200 bootstrap samples and preferably 500 or more.
`boot_level`	One minus the nominal coverage for the bootstrap confidence interval estimates.
`show_progress`	Whether to display intermediate updates on the progress of the bootstrap simulations. If show_progress==FALSE then the funmediation function runs silently but results can be viewed via the print and plot methods. If show_progress==TRUE then progress messages will be printed.

Value

An object of type funmediation. The components of an object of type funmediation are as follows:

original_results The estimates from the fitted models for predicting the mediator from the treatment, predicting the outcome from the mediator and treatment, and predicting the outcome from the treatment alone.

bootstrap_results The estimate and confidence interval of the indirect effect using a bootstrap approach.

The original_results component has these components within it:

time_grid Grid of time points on which the functional coefficients are estimated.

alpha_int_estimate Estimated intercept function (as a vector of estimates) from the TVEM regression of the mediator, M, on treatment, X.

alpha_int_se Estimated pointwise standard errors associated with the above.

alpha_X_estimate Estimated time-varying treatment effect from the TVEM regression of the mediator, M, on the treatment, X.

alpha_X_se Estimated pointwise standard errors associated with the above.

beta_int_estimate Estimated scalar intercept from the scalar-on- function regression of the outcome, Y, on the mediator, M, and treatment, X.

beta_int_se Estimated standard error for the above.

beta_X_estimate Estimated scalar coefficient for the treatment, X, from the scalar-on-function regression of the outcome, Y, on the mediator, M, and treatment, X.

beta_X_se Estimated standard error for the above.

beta_M_estimate Estimated functional coefficient for the mediator, M, from the scalar-on-function regression of the outcome, Y, on the mediator, M, and treatment, X.

beta_M_se Estimated pointwise standard errors associated with the above

beta_M_pvalue The p-value for significance of the mediator, M, in predicting outcome, Y, after adjusting for treatment, X.

tau_int_estimate Intercept from simple model predicting outcome, Y, directly from treatment, X.

tau_int_se Estimated standard error for the above.

tau_X_estimate Coefficient for treatment in model predicting outcome, Y, directly from treatment, X.

tau_X_se Estimated standard error for the above.

indirect_effect_estimate Estimated indirect effect, calculated as the dot product of the effect of treatment on mediator and the treatment- adjusted effect of mediator on outcome. It is a scalar, even though the two component effects are functions of time.

tvem_XM_details Detailed output from the tvem function for the time- varying-effect model predicting the mediator, M, from the treatment, X.

funreg_MY_details Detailed output from the refund::pfr function for the scalar-on-function functional regression predicting the outcome, Y, from the treatment, X, and mediator, M.

total_effect_details Detailed output from the linear or generalized linear model predicting the outcome from the treatment alone, ignoring the mediator (i.e., total effect)

The bootstrap_results component has these components within it:

indirect_effect_boot_estimate Bootstrap point estimate of the indirect effect (average of bootstrap sample estimates).

indirect_effect_boot_se Bootstrap standard error for the indirect effect (standard deviation of bootstrap sample estimates).

indirect_effect_boot_norm_lower Lower end of the bootstrap confidence interval using the normal method in boot.ci in the boot package.

indirect_effect_boot_norm_upper Upper end of the bootstrap confidence interval using the normal method.

indirect_effect_boot_basic_lower Lower end of the bootstrap confidence interval using the basic method in boot.ci in the boot package.

indirect_effect_boot_basic_upper Upper end of the bootstrap confidence interval using the basic method.

indirect_effect_boot_perc_lower Lower end of the bootstrap confidence interval using the percentile method in boot.ci in the boot package.

indirect_effect_boot_perc_upper Upper end of the bootstrap confidence interval using the percentile method.

boot_level The alpha level used for the bootstrap confidence interval.

boot1 The output returned from the boot function.

time.required The amount of time spent doing the bootstrap test, including generating and analyzing all samples.

Note

This function calls the tvem function in the tvem package. It also calls the pfr function in the refund package (see Goldsmith et al., 2011) to perform penalized functional regression. Some suggestions on interpreting the output from penalized functional regression are given by Dziak et al. (2019).

References

Baron, R.M., & Kenny, D.A. (1986). The moderator-mediator variable distinction in social psychological research: Conceptual, strategic, and statistical considerations. Journal of Personality & Social Psychology, 51: 1173-1182.

Coffman, D. L., Dziak, J. J., Litson, K., Chakraborti, Y., Piper, M. E., & Li, R. #' (2021). A causal approach to functional mediation analysis with application to a smoking cessation intervention. <arXiv:2112.03960>

Dziak, J. J., Coffman, D. L., Reimherr, M., Petrovich, J., Li, R., Shiffman, S., & Shiyko, M. P. (2019). Scalar-on-function regression for predicting distal outcomes from intensively gathered longitudinal data: interpretability for applied scientists. Statistics Surveys, 13, 150-180. <doi:10.1214/19-SS126>

Goldsmith, J., Bobb, J., Crainiceanu, C., Caffo, B., & Reich, D. (2011). Penalized functional regression. Journal of Computational and Graphical Statistics, 20(4), 830-851. <doi:10.1198/jcgs.2010.10007>

Lindquist, M. A. (2012). Functional Causal Mediation Analysis With an Application to Brain Connectivity. Journal of the American Statistical Association, 107: 1297-1309. <doi:10.1080/01621459.2012.695640>

`plot.funmediation` *plot.funmediation: Produces plots for a funmediation model.*

Description

Produces plots from a funmediation object produced by the funmediation function. These plots will be shown on the default output device (likely the screen); they can of course be written to a file instead, by preceding the call to plot with a call to png(), pdf(), or other R graphic file output functions.

Usage

```
## S3 method for class 'funmediation'
plot(
  x,
  use_panes = TRUE,
  what_plot = c("pfr", "pfrgam", "coefs", "tvem"),
  alpha_level = 0.05,
  ...
)
```

Arguments

`x`	The funmediation object to be plotted.
`use_panes`	Whether to plot multiple coefficient functions in a single image.
`what_plot`	One of "pfr","coefs", or "tvem." These options are as follows: **pfr** For a "pfr" plot, the functional coefficient for predicting the outcome, Y, from the mediator, M (conditional on X), is shown.

pfrgam Similar to a "pfr" plot, but uses the plot method for the penalized functional regression results. See the documentation for pfr() in the refund library and for plot.gam() in the mgcv library for more information.

coefs For a "coefs" plot, the three important functional coefficients in the model (intercept for predicting M, effect of X on M, and the effect of M on Y adjusting for X) are plotted one after another. That is, the plots are shown for the alpha_int_estimate, alpha_X_estimate, and beta_M_estimate, each as a function of time_grid. Approximate pointwise 95 percent confidence intervals are also shown if possible. If there is only one dichotomous treatment variable and panes are being used, the lower right pane will be free, so the indirect effect will be printed there even though it is a scalar.

tvem For a "tvem" plot, the functional coefficients in the TVEM model predicting M from X are displayed.

`alpha_level` Default is .05 for pointwise 95 percent confidence intervals.

`...` Further arguments currently not supported

Value

This function does not return an object, but is called for its side effect of plotting to the active device.

`print.funmediation` *print.funmediation: Print output from a model that was fit by the funmediation function.*

Description

print.funmediation: Print output from a model that was fit by the funmediation function.

Usage

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

Arguments

`x` The funmediation object (output of the funmediation function)

`...` Further arguments currently not supported

Value

This function does not return an object, but is called for its side effect of printing information.

simulate_funmediation_example
simulate_funmediation_example function

Description

Simulates a dataset for demonstrating the funmediation function.

Usage

```
simulate_funmediation_example(
  nsub = 500,
  nlevels = 2,
  ntimes = 100,
  observe_rate = 0.4,
  alpha_int = function(t) {     return(t^0.5) },
  alpha_X = function(t) {     return(-(t/2)^0.5) },
  beta_M = function(t) {     (1/2) * (exp(t) - 1) },
  beta_int = 0,
  beta_X = 0.2,
  sigma_Y = 1,
  sigma_M_error = 2,
  rho_M_error = 0.8,
  simulate_binary_Y = FALSE,
  make_covariate_S = FALSE
)
```

Arguments

`nsub`	Number of subjects
`nlevels`	Number of treatment groups or levels on the treatment variable X. Subjects are assumed to be randomly assigned to each level with equal probability (i.e., the probability per level is 1/nlevel). Default is 2 for a randomized controlled trial with a control group X=0 and an experimental group X=1. There should not be less than 2 or more than 5 groups for purposes of this function.
`ntimes`	Number of potential times that could be observed on each subject
`observe_rate`	Proportion of potential times on which there are actually observations. Not all times are observed; this is assumed to be completely random and to be done by design to reduce participant burden.
`alpha_int`	Function representing the time-varying mean of mediator variable for the level of treatment with all treatment dummy codes X set to 0 (e.g., the control group).
`alpha_X`	Function representing the time-varying effect of X on the mediator (if there are two treatment levels) or a list of nlevels-1 functions representing the effect of receiving each nonzero level of X rather than control (if there are more than two treatment levels).

beta_M
: Function representing the functional coefficient for cumulative (scalar-on-function) effect of the mediator M on the treatment Y adjusting for the treatment X

beta_int
: Mean of Y if the X is zero and M is the 0 function

beta_X
: Numeric value representing the direct effect of X on Y after adjusting for M (if there are two treatment levels) or a vector of nlevels-1 numeric values (if there are more than two treatment levels)

sigma_Y
: Error standard deviation of the outcome Y (conditional on treatment and mediator trajectory)

sigma_M_error
: Error standard deviation of the mediator M (conditional on treatment and time)

rho_M_error
: Autoregressive correlation coefficient of the error in the mediator M, from one observation to the next

simulate_binary_Y
: Whether Y should be generated from a binary logistic (TRUE) or Gaussian (FALSE) model

make_covariate_S
: Whether to generate a random binary covariate S at the subject (i.e., time-invariant) level. It will be generated to have zero population-level relationship to the outcome.

Value

A list with the following components:

time_grid The time grid for interpreting functional coefficients.

true_alpha_int True value of the time-varying alpha_int parameter, representing the time-specific mean of the mediator M when the treatment value X is 0.

true_alpha_X True value of the time-varying alpha_X parameter, representing the effect of X on M. This is a single number if nlevels=2, or a vector of effects if nlevels>2.

true_beta_int True value of the beta_M parameter, representing the mean of the outcome Y when X=0 and M=0.

true_beta_M True value of the beta_M parameter, representing the functional effect of treatment on the outcome Y.

true_beta_X True value of the beta_X parameter, representing the effect of treatment on the outcome Y adjusting for the mediator. This is a single function if nlevels=2, or a vector of functions if nlevels>2.

true_indirect True value of the indirect parameter, representing the indirect (mediated) effect of treatment on the outcome Y. This is a single number if nlevels=2, or a vector of effects if nlevels>2.

dataset The simulated longitudinal dataset in long form.

Examples

```
set.seed(123)
# Simplest way to call the function:
simulation_all_defaults <- simulate_funmediation_example()
summary(simulation_all_defaults)
```

```
head(simulation_all_defaults)
# Changing the sample size to be larger:
simulation_larger <- simulate_funmediation_example(nsub=10000)
summary(simulation_larger)
# Changing the effect of the mediator to be null:
simulation_null <- simulate_funmediation_example(beta_M=function(t) {return(0*t)})
summary(simulation_null)
# Simulating a exposure variable with three levels (two dichotomous dummy codes)
simulation_three_group <- simulate_funmediation_example(nlevels=3,
                              alpha_X = list(function(t) {return(.1*t)},
                                             function(t) {return(-(t/2)^.5)}),
                              beta_X = c(-.2,.2))
print(summary(simulation_three_group));
```

Index