Package: tvem (via r-universe)

August 25, 2024

Type Package

Title Time-Varying Effect Models

Version 1.4

Date 2023-07-01

VignetteBuilder knitr

Depends mgcv

Suggests knitr, rmarkdown

Encoding UTF-8

Description Fits time-varying effect models (TVEM). These are a kind of application of varying-coefficient models in the context of longitudinal data, allowing the strength of linear, logistic, or Poisson regression relationships to change over time. These models are described further in Tan, Shiyko, Li, Li & Dierker (2012) <doi:10.1037/a0025814>. We thank Kaylee Litson, Patricia Berglund, Yajnaseni Chakraborti, and Hanjoo Kim for their valuable help with testing the package and the documentation. 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)

RoxygenNote 7.2.3

Repository https://dziakj1.r-universe.dev

RemoteUrl https://github.com/dziakj1/tvem

RemoteRef HEAD

RemoteSha 08b095c1780bf98d8ec7448c3c07bb14b991edbc

Contents

plot.tvem 2
print.tvem 3
select_tvem 3
simulate_cross_sectional_tvem_example 4
simulate_tvem_example 6
tvem 9

Index **13**

`plot.tvem` *plot.tvem: Produce plots for a tvem model.*

Description

Produces plots from a tvem object produced by the tvem 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.tvem with a call to png(), pdf(), or other R graphic file output functions.

Usage

```
## S3 method for class 'tvem'
plot(
  x,
  use_panes = TRUE,
  which_plot = NULL,
  diagnostics = FALSE,
  exponentiate = FALSE,
  ...
)
```

Arguments

`x`	The TVEM object to be plotted.
`use_panes`	Whether to plot multiple coefficient functions in a single image.
`which_plot`	The coefficient number to plot, if only one plot is desired at a time.
`diagnostics`	If this is set to TRUE, then instead of plotting coefficient functions, the function will show a histogram of residuals and a plot of fitted values versus residuals. These may be useful in checking for outliers or skew in TVEM with a numeric outcome. They are not likely to be as useful in TVEM with a binary or other discrete outcome.

exponentiate	If this is set to TRUE and if the TVEM had a binary outcome, then the exponentiated coefficient functions (representing odds and odds ratios) will be plotted rather than the usual coefficient functions (representing log odds and log odds ratios).
...	Further arguments currently not supported

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

Description

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

Usage

```
## S3 method for class 'tvem'
print(x, ornate = TRUE, ...)
```

Arguments

x	The tvem object (output of the tvem or select_tvem function)
ornate	Whether to print lines between different sections of the output for easier reading.
...	Further arguments currently not supported

`select_tvem` *select_tvem: Select number of interior knots for an unpenalized TVEM.*

Description

select_tvem: Select number of interior knots for an unpenalized TVEM.

Usage

```
select_tvem(
  max_knots = 5,
  keep_going_if_too_few = FALSE,
  use_bic = FALSE,
  penalize = FALSE,
  print_output = TRUE,
  ...
)
```

Arguments

max_knots	The maximum number of interior knots to try (0 through max_knots)
keep_going_if_too_few	Whether to continue in a stepwise fashion if the max_knots does not seem to be high enough
use_bic	Whether to use BIC (TRUE) instead of AIC (FALSE) when selecting the best model. Note that both of these IC's are calculated from the working-independence pseudolikelihood rather than the unknown true likelihood. However, for BIC, the sample size is taken to be the number of subjects, not the number of observations.
penalize	Whether to include a penalty function in estimation
print_output	Whether to print the pseudolikelihoods obtained for each candidate number of interior knots.
...	Other inputs to be sent along to each call to the tvem function.

Value

A TVEM object for the fitted model, with an additional component containing a table of information criteria.

Examples

```
set.seed(123)
the_data <- simulate_tvem_example(n_subjects=200)
tvem_model <- tvem(data=the_data,
              formula=y~1,
              id=subject_id,
              time=time)
print(tvem_model)
plot(tvem_model)
```

---

simulate_cross_sectional_tvem_example

simulate_cross_sectional_tvem_example: Simulate a cross-sectional dataset for demonstrating tvem.

Description

This differs from what simulate_tvem_example does, in that each participant (subject) is only measured at a single measurement time, but the measurement time of each subject is random. This might simulate a sample of people of many different ages, where age is treated as a continuous number.

Usage

```
simulate_cross_sectional_tvem_example(
  n_subjects = 300,
  min_time = 7,
  max_time = 7,
  simulate_binary = FALSE,
  sigma_x1 = 2,
  sigma_x2 = 2,
  truncate_for_realism = TRUE,
  round_digits = 3,
  sigma_y = 1.5,
  mu_x1_function = function(t) {
     6 - 2 * sqrt(pmax(0, ((t - min(t))/7) - 0.2))
 },
  mu_x2_function = function(t) {
     3 + sqrt(pmax(0, ((t - min(t))/7) - 0.5))
 },
  beta0_y_function = function(t) {
     1 - 0.3 * sqrt((t - min(t))/7)
 },
  beta1_y_function = function(t) {
     0.5 * ((t - min(t))/7)^2
 },
  beta2_y_function = function(t) {
     rep(0.2, length(t))
 }
)
```

Arguments

n_subjects	Number of subjects in dataset
min_time	The time point at the end of the simulated time interval
max_time	The time point at the end of the simulated time interval
simulate_binary	Whether the simulated data should be binary
sigma_x1	Standard deviation of covariate 1, assumed homoskedastic over time
sigma_x2	Standard deviation of covariate 2, assumed homoskedastic over time
truncate_for_realism	Whether to prevent simulated values from going below 0 or above 10, in order to imitate survey data; used only for normally distributed outcomes. Set this to FALSE if you are running a simulation, in order to avoid losing coverage due to departing from the parametric model.
round_digits	Number of digits at which to round the generated data; used only for normally distributed outcomes
sigma_y	Error standard deviation of y, only used if the outcomes are to be normal rather than binary

`mu_x1_function` Mean of covariate 1 as function of time

`mu_x2_function` Mean of covariate 2 as function of time

`beta0_y_function`
TVEM intercept as function of time

`beta1_y_function`
TVEM coefficient of covariate 1 as function of time

`beta2_y_function`
TVEM coefficient of covariate 2 as function of time

Details

By default, the data-generating model has a time-varying intercept, and two time-varying covariates named x1 and x2. x1 has a time-varying effect and x2 has a time-invariant effect.

Value

A simulated dataset with the following variables:

subject_id Subject ID

time Observation time

x1 First covariate

x2 Second covariate

y Outcome variable

Examples

```
set.seed(16802)
the_data <- simulate_tvem_example(simulate_binary=TRUE)
```

`simulate_tvem_example` *simulate_tvem_example: Simulate a dataset for demonstrating the tvem function.*

Description

By default, the data-generating model has a time-varying intercept, and two time-varying covariates named x1 and x2. x1 has a time-varying effect and x2 has a time-invariant effect.

Usage

```
simulate_tvem_example(
  n_subjects = 300,
  max_time = 7,
  simulate_binary = FALSE,
  n_obs_possible = 141,
  prop_obs_observed = 0.3,
  sigma_x1 = 2,
  sigma_x2 = 2,
  truncate_for_realism = TRUE,
  round_digits = 3,
  x1_short_term_rho = 0.7,
  x2_short_term_rho = 0.7,
  sigma_y = 1.5,
  mu_x1_function = function(t) {
     6 - 2 * sqrt(pmax(0, (t/7) - 0.2))
 },
  mu_x2_function = function(t) {
     3 + sqrt(pmax(0, (t/7) - 0.5))
 },
  beta0_y_function = function(t) {
     1 - 0.3 * sqrt(t/7)
 },
  beta1_y_function = function(t) {
     0.5 * (t/7)^2
 },
  beta2_y_function = function(t) {
     rep(0.2, length(t))
 },
  y_short_term_rho = 0.8
)
```

Arguments

n_subjects	Number of subjects in dataset
max_time	The time point at the end of the simulated time interval
simulate_binary	Whether the simulated data should be binary
n_obs_possible	Total number of possible measurement times per subject
prop_obs_observed	Proportion of these that are actually observed
sigma_x1	Standard deviation of covariate 1, assumed homoskedastic over time
sigma_x2	Standard deviation of covariate 2, assumed homoskedastic over time
truncate_for_realism	Whether to prevent simulated values from going below 0 or above 10, in order to imitate survey data; used only for normally distributed outcomes. Set this to

FALSE if you are running a simulation, in order to avoid losing coverage due to departing from the parametric model.

`round_digits` Number of digits at which to round the generated data; used only for normally distributed outcomes

`x1_short_term_rho`
Correlation between adjacent measurements of covariate 1

`x2_short_term_rho`
Correlation between adjacent measurements of covariate 2

`sigma_y` Error standard deviation of y, only used if the outcomes are to be normal rather than binary

`mu_x1_function` Mean of covariate 1 as function of time

`mu_x2_function` Mean of covariate 2 as function of time

`beta0_y_function`
TVEM intercept as function of time

`beta1_y_function`
TVEM coefficient of covariate 1 as function of time

`beta2_y_function`
TVEM coefficient of covariate 2 as function of time

`y_short_term_rho`
Correlation between adjacent measurements of y, only used if it is normal and not binary

Value

A simulated dataset with the following variables:

subject_id Subject ID

time Observation time

x1 First covariate

x2 Second covariate

y Outcome variable

Examples

```
set.seed(123)
the_data <- simulate_tvem_example(simulate_binary=TRUE)
```

tvem *tvem: Fit a time-varying effect model.*

Description

Fits a time-varying effect model (Tan et al., 2012); that is, a varying-coefficients model (Hastie & Tibshirani, 1993) for longitudinal data.

Usage

```
tvem(
  data,
  formula,
  id,
  time,
  invar_effects = NULL,
  family = gaussian(),
  weights = NULL,
  num_knots = 20,
  spline_order = 3,
  penalty_function_order = 1,
  grid = 100,
  penalize = TRUE,
  alpha = 0.05,
  basis = "ps",
  method = "fREML",
  use_naive_se = FALSE,
  print_gam_formula = FALSE,
  normalize_weights = TRUE
)
```

Arguments

data	The dataset containing the observations, assumed to be in long form (i.e., one row per observation, potentially multiple rows per subject).
formula	A formula listing the outcome on the left side, and the time-varying effects on the right-side. For a time- varying intercept only, use y~1, where y is the name of the outcome. For a single time-varying-effects covariate, use y~x, where x is the name of the covariate. For multiple covariates, use syntax like y~x1+x2. Do not include the non- time-varying-effects covariates here. Note that the values of these covariates themselves may either be time-varying or time-invariant. For example, time-invariant biological sex may have a time-varying effect on time-varying height during childhood.
id	The name of the variable in the dataset which represents subject (participant) identity. Observations are considered to be correlated within subject (although the correlation structure is not explicitly modeled) but are assumed independent between subjects.

`time` The name of the variable in the dataset which represents time. The regression coefficient functions representing the time-varying effects are assumed to be smooth functions of this variable.

`invar_effects` Optionally, the names of one or more variables in the dataset assumed to have a non-time-varying (i.e., time-invariant) regression effect on the outcome. The values of these covariates themselves may either be time-varying or time-invariant. The covariates should be specified as the right side of a formula, e.g., ~x1 or ~x1+x2.

`family` The outcome family, as specified in functions like glm. For a numerical outcome you can use the default of gaussian(). For a binary outcome, use binomial(). For a count outcome, you can use poisson(). The parentheses after the family name are there because it is actually a built-in R object.

`weights` An optional sampling weight variable.

`num_knots` The number of interior knots assumed per spline function, not counting exterior knots. This is assumed to be the same for each function. If penalized=TRUE is used, it is probably okay to leave num_knots at its default.

`spline_order` The shape of the function between knots, with a default of 3 representing cubic spline.

`penalty_function_order`
The order of the penalty function (see Eilers and Marx, 1996), with a default of 1 for first-order difference penalty. Eilers and Marx (1996) used second-order difference but we found first-order seemed to perform parsimoniously in this setting. Please feel free to consider setting this to 2 to explore other possible results. The penalty function is something analogous to a prior distribution describing how smooth or flat the estimated coefficient functions should be, with 1 being smoothest.

`grid` The number of points at which the spline coefficients will be estimated, for the purposes of the pointwise estimates and pointwise standard errors to be included in the output object. The grid points will be generated as equally spaced over the observed interval. Alternatively, grid can be specified as a vector instead, in which each number in the vector is interpreted as a time point for the grid itself.

`penalize` Whether to add a complexity penalty; TRUE or FALSE

`alpha` One minus the nominal coverage for the pointwise confidence intervals to be constructed. Note that a multiple comparisons correction is not applied. Also, in some cases the nominal coverage may not be exactly achieved even pointwise, because of uncertainty in the tuning parameter and risk of overfitting. These problems are not unique to TVEM but are found in many curve- fitting situations.

`basis` Form of function basis (an optional argument about computational details passed on to the mgcv::s function as bs=). We strongly recommend leaving it at the default value.

`method` Fitting method (an optional argument about computational details passed on to the mgcv::bam function as method). We strongly recommend leaving it at the default value.

`use_naive_se` Whether to save time by using a simpler, less valid formula for standard errors. Only do this if you are doing TVEM inside a loop for bootstrapping or model selection and plan to ignore these standard errors.

`print_gam_formula`
whether to print the formula used to do the back-end calculations in the bam (large data gam) function in the mgcv package.

`normalize_weights`
Whether to rescale (standardize) the weights variable to have a mean of 1 for the dataset used in the analysis. Setting this to FALSE might lead to invalid standard errors caused by misrepresentation of the true sample size. This option is irrelevant and ignored if a weight variable is not specified, because in that case all the weights are effectively 1 anyway. An error will result if the function is asked to rescale weights and any of the weights are negative; however, it is very rare for sampling weights to be negative.

Value

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

time_grid A vector containing many evenly spaced time points along the interval between the lowest and highest observed time value. The exact number of points is determined by the input parameter 'grid'.

grid_fitted_coefficients A list of data frames, one for each smooth function which was fit (including the intercept). Each data frame contains the fitted estimates of the function at each point of time_grid, along with pointwise standard errors and pointwise confidence intervals.

invar_effects_estimates If any variables are specified in invar_effects, their estimated regression coefficients and standard errors are shown here.

model_information A list summarizing the options specified in the call to the function, as well as fit statistics based on the log-pseudo-likelihood function. The term pseudo here means that the likelihood function is evaluated as though the correct knot locations were known, as though the observations were independent and, if applicable, as though sampling weights were multiples of a participant rather than inverse probabilities. This allows tvem to be used without specifying a fully parametric probability model.

back_end_model The full output from the bam() function from the mgcv package, which was used to fit the penalized spline regression model underlying the TVEM.

Note

The interface is based somewhat on the TVEM 3.1.1 SAS macro by the Methodology Center (Li et al., 2017). However, that macro uses either "P-splines" (penalized truncated power splines) or "B-splines" (unpenalized B[asic]-splines, like those of Eilers and Marx, 1996, but without the smoothing penalty). The current function uses penalized B-splines, much more like those of Eilers and Marx (1996). However, their use is more like the "P-spline" method than the "B-spline" method in the TVEM 3.1.1 SAS macro, in that the precise choice of knots is not critical, the tuning is done automatically, and the fitted model is intended to be interpreted in a population-averaged (i.e., marginal) way. Thus, random effects are not allowed, but sandwich standard errors are used in attempt to account for within-subject correlation, similar to working-independence GEE (Liang and Zeger, 1986).

Note that as in ordinary parametric regression, if the range of the covariate does not include values near zero, then the interpretation of the intercept coefficient may be somewhat difficult and its standard errors may be large (i.e., due to extrapolation).

The bam ("Big Additive Models") function in the mgcv package ("Mixed GAM Computation Vehicle with GCV/AIC/REML smoothness estimation and GAMMs by REML/PQL") by Simon Wood is used for back-end calculations (see Wood, Goude, & Shaw, 2015).

References

Eilers, P. H. C., & Marx, B. D. (1996). Flexible smoothing with B-splines and penalties. Statistical Science, 11: 89-121. <doi:10.1214/ss/1038425655>

Hastie, T, Tibshirani, R. (1993). Varying-coefficient models. Journal of the Royal Statistical Socety, B, 55:757-796. <doi:10.1057/9780230280830_39>

Li, R., Dziak, J. J., Tan, X., Huang, L., Wagner, A. T., & Yang, J. (2017). TVEM (time-varying effect model) SAS macro users' guide (Version 3.1.1). University Park: The Methodology Center, Penn State. Retrieved from <http://methodology.psu.edu>. Available online at <https://aimlab.psu.edu/tvem/tvem-sas-macro/> and archived at <https://github.com/dziakj1/MethodologyCenterTVEMmacros> and <https://scholarsphere.psu.edu/collections/v41687m23q>.

Liang, K. Y., Zeger, S. L. Longitudinal data analysis using generalized linear models. Biometrika. 1986; 73:13-22. <doi:10.1093/biomet/73.1.13>

Tan, X., Shiyko, M. P., Li, R., Li, Y., & Dierker, L. (2012). A time-varying effect model for intensive longitudinal data. Psychological Methods, 17: 61-77. <doi:10.1037/a0025814>

Wood, S. N., Goude, Y., & Shaw, S. (2015). Generalized additive models for large data sets. Applied Statistics, 64: 139-155. ISBN 10 1498728332, ISBN 13 978-1498728331.

Examples

```
set.seed(123)
the_data <- simulate_tvem_example()
tvem_model <- tvem(data=the_data,
              formula=y~x1,
              invar_effects=~x2,
              id=subject_id,
              time=time)
print(tvem_model)
plot(tvem_model)
```

Index

plot.tvem, 2
print.tvem, 3

select_tvem, 3
simulate_cross_sectional_tvem_example, 4
simulate_tvem_example, 6

tvem, 9