Package 'DTVEM'

October 5, 2020

Type Package

Title A package to identify and model optimal time lags

Version 1.0009

Date 2020-10-05

Author Nicholas Jacobson <njacobson88@gmail.com>

Maintainer Nicholas Jacobson <njacobson88@gmail.com>

Description This package is for identifying and modeling optimal time lags using the Differential Time-Varying Effect Model (DTVEM). This package combines exploratory and confirmatory steps to (1) identify optimal time lags of processes predicting themselves or one another (i.e. univariate or multivariate), (2) find lags within data from continuous time or discrete time (note that all final output is in discrete time). See Jacobson, Chow, and Newman (2019) for full description.

License MIT

Imports mgcv,
plyr,
zoo,
reshape2,
Rcpp,
OpenMx

Repository CRAN

RoxygenNote 7.1.1

R topics documented:

`datamanipulation` *Data manipulation*

Description

PLEASE USE THE LAG FUNCTION RATHER THAN THIS UNLESS YOU WOULD LIKE TO SPECIFY THIS MANUALLY. Manipulates the data so that it creates a vector that is both in wide and long formats

Usage

```
datamanipulation(
  ...,
  numberofvars = numberofvars,
  input_list = input_list,
  differentialtimevaryingpredictors = NULL,
  outcome = NULL,
  controlvariables = NULL,
  data = NULL,
  ID = "ID",
  Time = NULL,
  controllag = NULL,
  standardized = TRUE,
  predictionstart = NULL,
  predictionsend = NULL,
  predictionsinterval = NULL,
  beforeblock = TRUE
)
```

Arguments

`...`	A list of variable names used in the function e.g. "X","Y"
`numberofvars`	The number of variables entered (normally passed from LAG function)
`input_list`	identical to the elipses above, A list of variable names used in the function e.g. "X","Y"
`differentialtimevaryingpredictors`	A list of predictor variables to be used as predictor variables e.g. c("X","Y")
`outcome`	The outcome(s) to be predicted e.g. c("X","Y")
`controlvariables`	The variables to be controlled for (not lagged).
`data`	The data frame to be converted
`ID`	The name of the ID variable. E.G. ID = "ID" (must be specified).
`Time`	The name of the Time variable. E.G. Time = "Time" (must be specified).
`controllag`	The time of the lag which coviarates should be controlled for (NOT CURRENTLY FUNCTIONAL)

`standardized` This specifies whether all of the variables (aside from Time) should be standardized. Options are TRUE, FALSE, and "center". TRUE means within-person standardize each variable, FALSE means use the raw data, "center" means to only within-person center the variables. Default = TRUE

`predictionstart` The differential time value to start with, default is NULL, and the lowest time difference in the time series will be used (use lower value if you're first value if you're interested in a smaller interval prediction)

`predictionsend` The differential time value to end with. This should usually be set by the user.

`predictionsinterval` The intervals to predict between differential time points, default = 1

`beforeblock` Is this before any data-blocking occurs? default = TRUE

Value

The output of this will be (1) The data in a very long stacked format (called laglongreducedummy), (2) the data in a wide format with all lags for all variables, time, and time differences (called Timelagsdummy), (3) the name of the differentialtimevarying predictors (namesofnewpredictorvariables), and (4) the length of the long matrix (laglongmatrixlength)

`decimalplaces` *Determine the number of decimal places*

Description

Determines the max number of decimal places based on the last non-zero digit.

Usage

```
decimalplaces(x)
```

Arguments

`x` A vector of numbers to examine for the last number of decimal places

Value

The number of decimal places used in a round command

DTVEM *Differential Time-Varying Effect Model (DTVEM) The package DTVEM is a package to identify optimal time lags.*

Description

This package is for identifying and modeling optimal time lags using the Differential Time-Varying Effect Model (DTVEM). This package combines exploratory and confirmatory steps to (1) identify optimal time lags of processes predicting themselves or one another (i.e. univariate or multivariate), (2) find lags within data from continuous time or discrete time (note that all final output is in discrete time).

Details

Package:	DTVEM
Type:	Package
Version:	1.0
Date:	2015-07-10
License:	MIT

This package identifies optimal time lags for processes predicting themselves or one another. The primary function is LAG (see ?LAG) for help.

Author(s)

Author: Nicholas Jacobson ¡njacobson88@gmail.com¿ Maintainer: Nicholas Jacobson ¡njacobson88@gmail.com¿

References

Jacobson, N. C., Chow, S, & Newman, M.G (2015). The Differential Time-Varying Effect Model (DTVEM): A Tool for Diagnosing Optimal Measurement and Modeling Intervals in Intensive Longitudinal Data.

Examples

```
#LAG("X",differntialtimevaryingpredictors=c("X"),outcome=c("X"),data=expppdat,Time="Time",k=8,ID="ID",pred
```

---

exampledat1 *Simulated Example Dataset 1: Univariate*

Description

A simulated example data set. This data set contains 100 persons, 14 time points, 35

Usage

```
data(exampledat1)
```

Format

A data frame with 1400 rows and 3 variables

Details

- ID. The ID variable representing each person.
- Time. The Time variable representing times 1 - 14.
- X. The variable of interest

`flatfunction` *Are the predicted intervals flat?*

Description

Is the smooth really flat (meaning that the max and the min values of the smooth are not significantly different?)

Usage

```
flatfunction(
  timepred = timepred,
  lowerboundvcfx = lowerboundvcfx,
  upperboundvcfx = upperboundvcfx,
  j = j
)
```

Arguments

`timepred`	This is the time prediction values from the start to the end of the predictions in intervals dictated by the intervals
`lowerboundvcfx`	The lower confidence interval of the variable of interest
`upperboundvcfx`	The upper confidence interval of the variable of interest
`j`	The index number of the flat function

Value

The output of this function includes whether flatness is an issue if the output will equal TRUE if it is and FALSE if it isn't.

gamstage2 *Stage 2 DTVEM: Multilevel model with a non-linear spline term for time*

Description

PLEASE USE THE LAG FUNCTION RATHER THAN THIS UNLESS YOU WOULD LIKE TO SPECIFY THIS MANUALLY. This runs the confirmatory stage of DTVEM using multilevel models and the spline effect of time.

Usage

```
gamstage2(
  Timelagsdummy = Timelagsdummy,
  variablenamesonlysigall = variablenamesonlysigall,
  independentpredictors = independentpredictors,
  k3 = k3,
  software = software
)
```

Arguments

Timelagsdummy	The data frame with lags from data manipulation
variablenamesonlysigall	The variables to be included in the confirmatory model
independentpredictors	This is whether or not the wide model comparisons should be run independently and combined via stepwise regression with backward selection. This can be useful to reduce the amount of lags included in the confirmatory model.
k3	The number of k selection points used in the model for the time spline (NOTE THAT THIS CONTROLS FOR TIME TRENDS OF THE POPULATION) (see ?choose.k in mgcv package for more details). Default is 3. (OPTIONAL)
software	This is the software used to run the secondary analysis. State-space models are implemented by the argument "OpenMx". The option "gam" can be used to run a traditional multilevel model with a spline that controls for non-linear time trends at the population level. The option "hybrid" first runs a multilevel model then runs an state-space model. Model. Note that the state-space approach can be very slow with large amounts of lags, and consequently "gam" should be used with large amounts of lags are included. However, state-space model estimation is generally marginally superior to the multilevel modeling approach, and if using small amounts of lags or time is not an issue the state-space option is recommended. The default is "OpenMx" which implements multilevel models. (OPTIONAL)

Value

The output of this function is: The output from the second stage of DTVEM in mgcv's gam

LAG *Differential Time-Varying Effect model (DTVEM)*

Description

This is the PRIMARY DTVEM function. It combines (1) exploratory lag identication, followed by confirmatory lag identifcation. This function will work with discrete or continuous time data (note that the underlying model is assumes to be discrete even when randomly measured). This function can handle multivariate predictors and multivariate outcomes (or univariate for both). The function will also automatically apply data manipulation required to run the models specified in Jacobson, Chow, and Newman (2019).

Usage

```
LAG(
  ...,
  differentialtimevaryingpredictors = NULL,
  outcome = NULL,
  controlvariables = NULL,
  data = NULL,
  ID = "ID",
  Time = NULL,
  k = 10,
  k2 = 10,
  k3 = 3,
  k4 = 3,
  controllag = NULL,
  standardized = TRUE,
  predictionstart = NULL,
  predictionsend = NULL,
  predictionsinterval = NULL,
  software = "OpenMx",
  independentpredictors = FALSE,
  minimumpracticalsignificance = NULL,
  gamma = 1,
  minN = 30,
  debug = FALSE,
  OpenMxStartingValues = 0.3,
  ResidualAnalysis = "Group",
  blockdata = FALSE,
  rounddecimals = TRUE,
  maxintermediaterounds = 10,
  differntialtimevaryingpredictors = NULL
)
```

Arguments

`...` A list of variable names used in the function e.g. "X","Y" (REQUIRED)

`differentialtimevaryingpredictors`
The variables that will be a varying-coefficient of differential time (AKA the lags you want to know what times they predict the outcome). This

must be specified as a vector using c("variables here"). e.g. c("X","Y") (REQUIRED)

`outcome` This is each of the outcome variables. Specified as outcome="outcomevariablename" for a single variable or outcome=c("outcomevariablename1","outcomevariablename2") (REQUIRED)

`controlvariables` The variables to be controlled for (not lagged). These are traditional covariates in the analysis. These are the variables that will be controlled for in a stationary fashion. To use this use controlvariables = c("list","here") (OPTIONAL)

`data` Specify the data frame that contains the data e.g. data=dataframename (REQUIRED)

`ID` The name of the ID variable. E.G. ID = "ID" (must be specified). (REQUIRED)

`Time` The name of the Time variable. E.G. Time = "Time" (must be specified). (REQUIRED)

`k` The number of k selection points used in the model for stage 1 (see ?choose.k in mgcv package for more details) (note that this is for the raw data k2 refers to the k for the re-blocked data), default is 10. The ideal k is the maximum number of data points per person, but this slows down DTVEM and is often not required. (OPTIONAL, BUT RECOMMENDED)

`k2` The number of k selection points used in the model for stage 1 of the blocked data (see ?choose.k in mgcv package for more details). Default is 10. The ideal k is the maximum number of data points per person, but this slows down DTVEM and is often not required. (OPTIONAL)

`k3` The number of k selection points used in the model for the time spline (NOTE THAT THIS CONTROLS FOR TIME TRENDS OF THE POPULATION) (see ?choose.k in mgcv package for more details). Default is 3. (OPTIONAL)

`k4` The number of k selection points used in the model for the varying coefficient in the intermediate stage (see ?choose.k in mgcv package for more details). Default is 3. (OPTIONAL, BUT RECOMMENDED)

`controllag` The time of the lag which coviarates should be controlled for (NOT CURRENTLY FUNCTIONAL)

`standardized` This specifies whether all of the variables (aside from Time) should be standardized. Options are TRUE, FALSE, and "center". TRUE means within-person standardize each variable (aka get the person-centered z-scores), FALSE means use the raw data, "center" means to only within-person mean-center the variables. Default = TRUE. FALSE is not recommended unless you have done these transformations yourself (OPTIONAL)

`predictionstart` The differential time value to start with, default is NULL, and the lowest time difference in the time series will be used (use lower value if you're first value if you're interested in a smaller interval prediction) e.g. predictionstart = 1. If this is not specified and using a continuous time model, make sure to set blockdata = TRUE so that it will be automatically chosen. (OPTIONAL)

`predictionsend` The differential time value to end with. This means how long you want your largest time difference in the study to be (i.e. if you wanted to predict up to allow time predictions up to 24 hours and your time intervals were specified in hours, you would set predictionsend = 24). If this is not specified and using a continuous time model, make sure to set blockdata = TRUE so that it will be automatically chosen. (OPTIONAL)

`predictionsinterval`
The intervals to predict between differential time points. If using discrete time do you want the intervals to be specified every discrete interval, if so set this to 1. If this is not specified and using a continuous time model, make sure to set blockdata = TRUE so that it will be automatically chosen. (OPTIONAL)

`software` This is the software used to run the secondary analysis. State-space models are implemented by the argument "OpenMx". The option "gam" can be used to run a traditional multilevel model with a spline that controls for non-linear time trends at the population level. The option "hybrid" first runs a multilevel model then runs an state-space model. Model. Note that the state-space approach can be very slow with large amounts of lags, and consequently "gam" should be used with large amounts of lags are included. However, state-space model estimation is generally marginally superior to the multilevel modeling approach, and if using small amounts of lags or time is not an issue the state-space option is recommended. The default is "OpenMx" which implements multilevel models. (OPTIONAL)

`independentpredictors`
This is whether or not the wide model comparisons should be run independently and combined via stepwise regression with backward selection. This can be useful to reduce the amount of lags included in the confirmatory model. Default is FALSE. (OPTIONAL)

`minimumpracticalsignificance`
This can be used to set a minimum amount to pass on from DTVEM stage 1 to stage 2, and stage 1.5 to stage 2. This can be useful if too many variables come back as significant, but they would not meet your criteria for practical significance. Set this to a numerical value (e.g. minimumpracticalsignificance=.2). (OPTIONAL, UNCOMMONLY SPECIFIED)

`gamma` This can be used to change the wiggliness of the model. This can be useful if the model is too smooth (i.e flat). The lower the number the more wiggly this will be (see ?gam in MGCV for more information). The default is equal to 1. (OPTIONAL, UNCOMMONLY SPECIFIED)

`minN` The smallest N that will be considered in the stage 2 model (i.e. this can be important in case you don't have observations at certain differential times, such as overnight observations). Default = 30. (OPTIONAL)

`debug` This will print more useless information as it goes along. Only useful for troubleshooting problems. (OPTIONAL, UNCOMMONLY SPECIFIED)

`OpenMxStartingValues`
Only applies when software = "OpenMx". Specify the starting values for OpenMx. Since OpenMx will 10 different runs before giving up on convergence this does not usually need to be specified. It should mostly only be specified if there is a convergence issue with OpenMx. Default is 0.3. (OPTIONAL, UNCOMMONLY SPECIFIED)

`ResidualAnalysis`
: Only applies when software = "OpenMx". Analyze the residuals of the time series with OpenMx after factoring out the non-linear effect of time (takes time trends into account). Can be run only at the group level (faster), or it can also be run with a random effect splines of time (slower) by setting ResidualAnalysis = "Individual". Default = "Group" (OPTIONAL)

`blockdata`
: This re-organizes the raw data into blocks after an exploratory first stage. Default = FALSE. TRUE = Automatic re-organization of data based on the minimum lag number and the time between two lags peaks/valleys. Including a numeric number will automatically re-block the data into chunks at those specific intervals. (OPTIONAL, UNCOMMONLY SPECIFIED)

`rounddecimals`
: The default option is TRUE which to automatically rounds the decimals to the smallest non-zero decimal place in the data. Can also specify a number to round to a specific decimal place (e.g. 1 = the tenths digit, 2 = the hundredths digit, 0 = the nearest whole number, -1 = the nearest 10th number) (OPTIONAL, UNCOMMONLY SPECIFIED)

`maxintermediaterounds`
: The maximum number of intermediate stages to perform. Default is 10 (OPTIONAL, UNCOMMONLY SPECIFIED)

`differntialtimevaryingpredictors`
: This is depricated. Only retained for backward compatibility.

Value

The output of this function is: (1) the stage 1 (i.e. the exploratory stage) (stage1out), (2) the stage 2 model (gamstage2out or OpenMxstage2out), (3) the data used to run the models in the first and second stage (datamanipulationout). Use str() around the saved object to see the information (it is useful to specify the max.level at a time so that the information does not get overwhelming)

Examples

```
#Load the example data set 1
data(exampledat1)

#Run Univariate DTVEM
#out=LAG("X",differentialtimevaryingpredictors=c("X"),outcome=c("X"),data=exampledat1,Time="Time",k=9,stand
#Example Bivariate DTVEM
#out=LAG("anxiety","depression",differentialtimevaryingpredictors = c("anxiety","depression"),outcome = c("a
```

---

localMaxima *Peak detection*

Description

Find local peaks of a set of values

Usage

```
localMaxima(x)
```

Arguments

x	A vector of numbers to examine for the local peaks

Value

The peaks of the vector of the numbers

---

nearbypeakvalleys *Nearby Peaks and Valleys Function*

Description

Looks for values that are nearby the peaks and valleys and are not significantly different. Note that all values should be specified, no values should be left to the defaults.

Usage

```
nearbypeakvalleys(
  values = peaks,
  lowerCI = lowerboundvcfx,
  upperCI = upperboundvcfx,
  timepred = timepred,
  updown = 1,
  j = j,
  predictionsend = predictionsend
)
```

Arguments

values	This is the vector of peaks or valleys to investigate
lowerCI	The lower confidence interval of the variable of interest
upperCI	The upper confidence interval of the variable of interest
timepred	This is the time prediction values from the start to the end of the predictions in intervals dictated by the intervals
updown	This tells the function whether to look up for the nearby values or down (1 for up, -1 for down)
j	This tells the function which variable is being investigated
predictionsend	The end of the time predictions

Value

The output of this function will be a vector of nearby peaks or valleys that are not significantly different from the inputted values

`OpenMxstage2` *Stage 2 OpenMx: State-Space Confirmatory Model*

Description

PLEASE USE THE LAG FUNCTION RATHER THAN THIS UNLESS YOU WOULD LIKE TO SPECIFY THIS MANUALLY.

Usage

```
OpenMxstage2(
  Timelagsdummy = Timelagsdummy,
  predictionstart = predictionstart,
  predictionsend = predictionsend,
  predictionsinterval = predictionsinterval,
  variablenamesonlysigall = variablenamesonlysigall,
  independentpredictors = independentpredictors,
  k3 = k3,
  numberofvars = numberofvars,
  OpenMxStartingValues = OpenMxStartingValues,
  ResidualAnalysis = ResidualAnalysis,
  varnames = varnames,
  outcome = outcome,
  numberofpeople = numberofpeople
)
```

Arguments

`Timelagsdummy` The Timelagsdummy data frame

`predictionstart`
: The differential time value to start with, default is NULL, and the lowest time difference in the time series will be used (use lower value if you're first value if you're interested in a smaller interval prediction) e.g. predictionstart = 1. If this is not specified and using a continuous time model, make sure to set blockdata = TRUE so that it will be automatically chosen. (OPTIONAL)

`predictionsend` The differential time value to end with. This means how long you want your largest time difference in the study to be (i.e. if you wanted to predict up to allow time predictions up to 24 hours and your time intervals were specified in hours, you would set predictionsend = 24). If this is not specified and using a continuous time model, make sure to set blockdata = TRUE so that it will be automatically chosen. (OPTIONAL)

`predictionsinterval`
: The intervals to predict between differential time points. If using discrete time do you want the intervals to be specified every discrete interval, if so set this to 1. If this is not specified and using a continuous time model, make sure to set blockdata = TRUE so that it will be automatically chosen. (OPTIONAL)

`variablenamesonlysigall`
: The variables to be included in the confirmatory model

`independentpredictors`
: This is whether or not the wide model comparisons should be run independently and combined via stepwise regression with backward selection. This can be useful to reduce the amount of lags included in the confirmatory model.

`k3`
: The number of k selection points used in the model for the time spline (NOTE THAT THIS CONTROLS FOR TIME TRENDS OF THE POPULATION) (see ?choose.k in mgcv package for more details). Default is 3. (OPTIONAL)

`numberofvars`
: This is the number of variables in the model

`OpenMxStartingValues`
: Only applies when software = "OpenMx". Specify the starting values for OpenMx. Since OpenMx will 10 different runs before giving up on convergence this does not usually need to be specified. It should mostly only be specified if there is a convergence issue with OpenMx. Default is 0.3. (OPTIONAL, UNCOMMONLY SPECIFIED)

`ResidualAnalysis`
: Only applies when software = "OpenMx". Analyze the residuals of the time series with OpenMx after factoring out the non-linear effect of time (takes time trends into account). Can be run only at the group level (faster), or it can also be run with a random effect splines of time (slower) by setting ResidualAnalysis = "Individual". Default = "Group" (OPTIONAL)

`varnames`
: What are the variable names?

`outcome`
: what are the outcome variables?

`numberofpeople`
: How many people are there?

`...`
: A list of variable names used in the function e.g. "X","Y" (REQUIRED)

Value

The output of this function is: The output from the second stage of DTVEM using the state-space approach

---

`stage1` *Stage 1 DTVEM*

Description

PLEASE USE THE LAG FUNCTION RATHER THAN THIS UNLESS YOU WOULD LIKE TO SPECIFY THIS MANUALLY. This is the function for the stage 1 of DTVEM. Manipulates the data so that it creates a vector that is both in wide and long formats

Usage

```
stage1(
  differentialtimevaryingpredictors = differentialtimevaryingpredictors,
  outcome = outcome,
  predictionstart = predictionstart,
  predictionsend = predictionsend,
```

```
  predictionsinterval = predictionsinterval,
  namesofnewpredictorvariables = namesofnewpredictorvariables,
  laglongreducedummy = laglongreducedummy,
  gamma = gamma,
  numberofknots = numberofknots,
  k3 = k3,
  controlvariables = controlvariables,
  debug = debug,
  minimumpracticalsignificance = minimumpracticalsignificance,
  minimumpracticalsignificanceneg = minimumpracticalsignificanceneg,
  intermediatestage = FALSE,
  lengthcovariates = lengthcovariates,
  blockdata = blockdata
)
```

Arguments

`differentialtimevaryingpredictors`
: The variables that will be a varying-coefficient of differential time (AKA the lags you want to know what times they predict the outcome). This must be specified as a vector using c("variables here"). e.g. c("X","Y") (REQUIRED)

`outcome`
: This is each of the outcome variables. Specified as outcome="outcomevariablename" for a single variable or outcome=c("outcomevariablename1","outcomevariablename2") (REQUIRED)

`predictionstart`
: The differential time value to start with, default is NULL, and the lowest time difference in the time series will be used (use lower value if you're first value if you're interested in a smaller interval prediction) e.g. predictionstart = 1. If this is not specified and using a continuous time model, make sure to set blockdata = TRUE so that it will be automatically chosen. (OPTIONAL)

`predictionsend`
: The differential time value to end with. This means how long you want your largest time difference in the study to be (i.e. if you wanted to predict up to allow time predictions up to 24 hours and your time intervals were specified in hours, you would set predictionsend = 24). If this is not specified and using a continuous time model, make sure to set blockdata = TRUE so that it will be automatically chosen. (OPTIONAL)

`predictionsinterval`
: The intervals to predict between differential time points. If using discrete time do you want the intervals to be specified every discrete interval, if so set this to 1. If this is not specified and using a continuous time model, make sure to set blockdata = TRUE so that it will be automatically chosen. (OPTIONAL)

`namesofnewpredictorvariables`
: This is the name of the predictors.

`laglongreducedummy`
: This is the long data output from the data manipulation.

`gamma`
: This can be used to change the wiggliness of the model. This can be useful if the model is too smooth (i.e flat). The lower the number the more wiggly this will be (see ?gam in MGCV for more information). The default is equal to 1. (OPTIONAL, UNCOMMONLY SPECIFIED)

`numberofknots` The number of k selection points used in the model for stage 1 (see ?choose.k in mgcv package for more details) (note that this is for the raw data k2 refers to the k for the re-blocked data), default is 10. The ideal k is the maximum number of data points per person, but this slows down DTVEM and is often not required. (OPTIONAL)

`k3` The number of k selection points used in the model for the time spline (NOTE THAT THIS CONTROLS FOR TIME TRENDS OF THE POPULATION) (see ?choose.k in mgcv package for more details). Default is 3. (OPTIONAL)

`controlvariables`
The variables to be controlled for (not lagged). These are traditional covariates in the analysis. These are the variables that will be controlled for in a stationary fashion. To use this use controlvariables = c("list","here") (OPTIONAL)

`debug` This will print more useless information as it goes along. Only useful for troubleshooting problems. (OPTIONAL, UNCOMMONLY SPECIFIED)

`minimumpracticalsignificance`
This can be used to set a minimum amount to pass on from DTVEM stage 1 to stage 2, and stage 1.5 to stage 2. This can be useful if too many variables come back as significant, but they would not meet your criteria for practical significance. Set this to a numerical value (e.g. minimumpracticalsignificance=.2). (OPTIONAL, UNCOMMONLY SPECIFIED)

`minimumpracticalsignificanceneg`
minimumpracticalsignificanceneg*-1

`intermediatestage`
Should not be changed unless running from DTVEM function.

`lengthcovariates`
the number of covariates (+2)

`blockdata` IMPORTANT FOR CONTINUOUS-TIME DATA. This re-organizes the raw data into blocks after an exploratory first stage. Default = FALSE. TRUE = Automatic re-organization of data based on the minimum lag number and the time between two lags peaks/valleys. Including a numeric number will automatically re-block the data into chunks at those specific intervals. (REQUIRED FOR CONTINUOUS DATA, OPTIONAL OTHERWISE)

... A list of variable names used in the function e.g. "X","Y" (REQUIRED)

Value

The output of this function is: The output from the first stage of DTVEM

`summary.DTVEM` *Summary of DTVEM Object*

Description

Summary of DTVEM Object

Usage

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

Arguments

object	The DTVEM object saved from the LAG function e.g. (out¡-LAG(); summary(out))
...	other arguments

Index