Package ‘SimTimeVar’

October 12, 2022

Type Package

Title Simulate Longitudinal Dataset with Time-Varying Correlated Covariates

Version 1.0.0

Date 2017-09-07

Author Maya B. Mathur, Kristopher Kapphahn, Ariadna Garcia, Manisha Desai, Maria E. Montez-Rath

Maintainer Maya B. Mathur <mmathur@stanford.edu>

Description Flexibly simulates a dataset with time-varying covariates with user-specified exchangeable correlation structures across and within clusters. Covariates can be normal or binary and can be static within a cluster or time-varying. Time-varying normal variables can optionally have linear trajectories within each cluster. See ?make_one_dataset for the main wrapper function. See Montez-Rath et al. <arXiv:1709.10074> for methodological details.

LazyData true

License GPL-2

Imports metafor, mvtnorm, ICC, miscTools, car, plyr, corpcor, psych, stats, utils

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2017-09-30 16:22:10 UTC

R topics documented:

add_one_categorical .............................................................. 2
add_time_function_vars .......................................................... 3
BN.rBound ................................................................. 3
cat.params ............................................................... 4
closest ................................................................. 4
complete_parameters ....................................................... 5
expand_matrix ............................................................ 5
add_one_categorical

Generate linear predictor from logistic model

Description
An internal function not intended for the user. Given a dataset and multinomial regression parameters, generates a categorical variable and adds it to the dataset.

Usage
add_one_categorical(.d, n, obs, cat.parameters)

Arguments
.d The dataset to which to add the categorical variable.

n The number of clusters.

obs The number of observations per cluster.

cat.parameters A dataframe of parameters for generating the categorical variable. See Details.

Examples
# mini dataset with 3 observations per person
data = data.frame( male = rep( rbinom(n=10, size=1, prob=0.5), each=3 ) )
add_one_categorical( data, 10, 3, cat.params)
add_time_function_vars

Creates linear time-function variables

Description

Given variable-specific slopes and intercepts for a cluster, creates continuous variables that increase
or decrease linearly in time (with normal error with standard deviation error.SD) and adds them to
the dataframe.

Usage

add_time_function_vars(d4, obs, parameters)

Arguments

d4
The dataframe to which to add the time-function variables.
obsc
The number of observations per cluster.
parameters
The parameters matrix.

Details

See make_one_dataset for additional information.

BN.rBound

Maximum correlation between binary and normal random variables

Description

Given parameter p for a Bernoulli random variable, returns its maximum possible correlation with
an arbitrary normal random variable. Used to adjust correlation matrices whose entries are not
theoretically possible.

Usage

BN.rBound(p)

Arguments

p
Parameter of Bernoulli random variable.

Examples

# find the largest possible correlation between a normal
# variable and a binary with parameter 0.1
BN.rBound(0.1)
**cat.params**  
*An example dataframe for categorical variable parameters*

**Description**

An example of how to set up the categorical variable parameters dataframe.

**Usage**

```
cat.params
```

**Format**

An object of class `data.frame` with 5 rows and 3 columns.

---

**closest**  
*Return closest value*

**Description**

An internal function not intended for the user. Given a number `x` and vector of permitted values, returns the closest permitted value to `x` (in absolute value).

**Usage**

```
closest(x, candidates)
```

**Arguments**

- `x` The number to be compared to the permitted values.
- `candidates` A vector of permitted values.

**Examples**

```
closest( x = 5, candidates = c(-3, 8, 25) )
```
**complete_parameters**  
*Fill in partially incomplete parameters matrix*

**Description**

Fills in "strategic" NA values in a user-provided parameters matrix by (1) calculating SDs for proportions using the binomial distribution; (2) calculating variances based on SDs; and (3) setting within-cluster variances to 1/3 of the across-cluster variances (if not already specified).

**Usage**

```r
complete_parameters(parameters, n)
```

**Arguments**

- `parameters`  
  Initial parameters matrix that may contain NA values.
- `n`  
  The number of clusters

**Details**

For binary variables, uses binomial distribution to compute across-cluster standard deviation of proportion. Where there are missing values, fills in variances given standard deviations and vice-versa. Where there are missing values in `within.var`, fills these in by defaulting to 1/3 of the corresponding across-cluster variance.

**Examples**

```r
complete_parameters(params, n=10)
```

---

**expand_matrix**  
*Longitudinally expand a matrix of single observations by cluster*

**Description**

An internal function not intended for the user. Given a matrix of single observations for a cluster, repeats each cluster’s entry in each `.obs` times.

**Usage**

```r
expand_matrix(.matrix, .obs)
```

**Arguments**

- `.matrix`  
  The matrix of observations to be expanded.
- `.obs`  
  The number of observations to generate per cluster.
**Examples**

```r
mat = matrix( seq(1:10), nrow=2, byrow=FALSE)
expand_matrix(mat, 4)
```

---

### Description

An internal function not intended for the user. Given a matrix of cluster means for each variable to be simulated, "expands" them into time-varying observations.

### Usage

```r
expand_subjects(mus3, n.OtherNorms, n.OtherBins, n.TBins, wcor, obs, parameters, zero = 1e-04)
```

### Arguments

- **mus3**: A matrix of cluster means for each variable.
- **n.OtherNorms**: The number normal variables (not counting those used for generating a time-varying binary variable).
- **n.OtherBins**: The number of static binary variables.
- **n.TBins**: The number of time-varying binary variables.
- **wcor**: The within-cluster correlation matrix.
- **obs**: The number of observations to generate per cluster.
- **parameters**: The parameters dataframe.
- **zero**: A small number just larger than 0.

### Examples

```r
# subject means matrix (normally would be created internally within make_one_dataset)
mus3 = structure(c(1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1e-04, 1e-04, 1e-04, 0.886306145591761, 1e-04, 1e-04, 1e-04, 1e-04, 0.875187001140343, 0.835990583043838, 1e-04, 1e-04, 1e-04, 1e-04, 1e-04, 1e-04, 1e-04, 1e-04, 1e-04, 1e-04, 1e-04, 1e-04, 1e-04, 1e-04, 69.7139993804559, 61.3137637852213, 68.3375516615242, 57.7893277997516, 66.3744152975352, 63.7829561873355, 66.3864252981679, 68.8513253460358, 67.4120718557, 67.8332651858068, 192.366192293195, 128.048983102848, 171.55040113259, 120.348392753954, 158.840864356998, 170.1348760994, 113.512220338299, 162.715528382999, 138.47687345895, 159.84106973242, 115.026417822477, 109.527137142158, 117.087914485084, 121.153861460319, 109.959735841412, 122.96960673409, 90.5100006255084, 107.53229006061, 108.97167788246, 115.641818648526, -4.33184270434101, -5.45143483618415, -2.5331188314257, -1.38284452333064, -1.6744564863871, 1.8391123741448, 2.048833883998, -0.237095062415858, -5.4749750687878, -3.53078955238741), .Dim = c(10L, 7L))
```
has_drug_suffix

expand_subjects( mus3 = mus3, n.OtherNorms = 4, n.OtherBins = 1, n.TBins = 2,
                 wcor = wcor, obs = 3, parameters = complete_parameters(params, n=10) )

has_drug_suffix  Checks whether string has "_s" suffix

Description
An internal function not intended for the user.

Usage
has_drug_suffix(var.name)

Arguments
  var.name  The string to be checked

Examples
  has_drug_suffix("myvariable_s")
  has_drug_suffix("myvariable")

make_one_dataset  Simulate time-varying covariates

Description
Simulates a dataset with correlated time-varying covariates with an exchangeable correlation structure. Covariates can be normal or binary and can be static within a cluster or time-varying. Time-varying normal variables can optionally have linear trajectories within each cluster.

Usage
make_one_dataset(n, obs, n.TBins, pcor, wcor, parameters, cat.parameters)

Arguments
  n         The number of clusters.
  obs       The number of observations per cluster.
  n.TBins   Number of time-varying binary variables.
  pcor      The across-subject correlation matrix. See Details.
  wcor      The within-subject correlation matrix. See Details.
  parameters A dataframe containing the general simulation parameters. See Details.
  cat.parameters A dataframe containing parameters for the categorical variables. See Details.
Details

SPECIFYING THE PARAMETERS MATRIX

The matrix parameters contains parameters required to generate all non-categorical variables. It must contain column names name, type, across.mean, across.SD, within.var, prop, and error.SD. (To see an example, use data(params).) Each variable to be generated requires either one or two rows in parameters, depending on the variable type.

The possible variable types and their corresponding specifications are:

- **Static binary variables** do not change over time within a cluster. For example, if clusters are subjects, sex would be a static binary variable. Generating such a variable requires a single row of type static.binary with prop corresponding to the proportion of clusters for which the variable equals 1 and all other columns set to NA. (The correct standard deviation will automatically be computed later.) For example, if the variable is an indicator for a subject’s being male, then prop specifies the proportion of males to be generated.

- **Time-varying binary variables** can change within a cluster over time, as for an indicator for whether a subject is currently taking the study drug. These variables require two rows in parameters. The first row should be of type static.binary with prop representing the proportion of clusters for which the time-varying binary variable is 1 at least once (and all other columns set to NA). For example, this row in parameters could represent the proportion of subjects who ever take the study drug ("ever-users"). The second row should be of type subject.prop with across.mean representing, for clusters that ever have a 1 for the binary variable, the proportion of observations within the cluster for which the variable is equal to 1. (All other columns should be set to NA.) For example, this row in parameters could represent the proportion of observations for which an ever-user is currently taking the drug. To indicate which pair of variables go together, the subject.prop should have the same name as the static.binary variable, but with the suffix _s appended (for example, the former could be named drug_s and the latter drug).

- **Normal variables** are normally distributed within a cluster such that the within-cluster means are themselves also normally distributed in the population of clusters. Generating a normal variable requires specification of the population mean (across.mean) and standard deviation (across.SD) as well as of the within-cluster standard deviation (within.SD). To generate a static continuous variable, simply set within.SD to be extremely small (e.g., $1 \times 10^{-7}$) and all corresponding correlations in matrix wcor to 0.

- **Time-function variables** are linear functions of time (with normal error) within each cluster such that the within-cluster baseline values are normally distributed in the population of clusters. Generating a time-function variable requires two entries. The first entry should be of type time.function and specifies the population mean (across.mean) and standard deviation (across.SD) of the within-cluster baseline values as well as the error standard deviation (error.SD). The second entry should be of type normal and should have the same name as the time.function entry, but with the "_s" suffix. This entry specifies the mean (across.mean) and standard deviation (across.SD) of the within-cluster slopes.

SPECIFYING THE CATEGORICAL PARAMETERS MATRIX

The matrix cat.parameters contains parameters required to generate the single categorical variable, if any. It must contain column names level, parameter, and beta. (To see an example, use data(cat.params).)
• **The reference level:** Each categorical variable must have exactly one "reference" level. The reference level should have one row in cat.parameters for which parameters is set to NA and beta is set to ref. For example, in the example file cat.params specifying parameters to generate a subject's race, the reference level is white.

• **Other levels:** Other levels of the categorical variable will have one or more rows. One row with parameter set to intercept and beta set to a numeric value represents the intercept term in the corresponding multinomial model. Any subsequent rows, with parameters set to names of other variables in the dataset and beta set to numeric values, represents other coefficients in the corresponding multinomial models.

### SPECIFYING THE POPULATION CORRELATION MATRIX

Matrix pcor specifies the population (i.e., across-cluster) correlation matrix. It should have the same number of rows and columns as parameters as well as the same variable names and ordering of variables.

### SPECIFYING THE WITHIN-CLUSTER CORRELATION MATRIX

Matrix wcor specifies the within-cluster correlation matrix. The order of the variables listed in this file should be consistent with the order in params and pcor. However, static.binary and subject.prop variables should not be included in wcor since they are static within a cluster. Static continuous variables should be included, but all the correlations should be set to zero.

### Examples

```r
data = make_one_dataset(n=10, obs=10, n.TBins=2, pcor=pcor, wcor=wcor,
parameters=complete_parameters(params, n=10), cat.parameters=cat.params)$data
```

### `make_one_linear_pred` 
Generate linear predictor from logistic model

### Description

An internal function not intended for the user. Given a matrix of regression parameters and a dataset, returns the linear predictor based on the given dataset.

### Usage

```r
make_one_linear_pred(m, data)
```

### Arguments

- `m` Part of the parameter matrix for the linear predictor for a single variable.
- `data` The dataframe from which to generate.
Examples

```r
# take part of parameters matrix corresponding to single level of categorical
# variable
m = cat.params[ cat.params$level == "black", ]
data = data.frame( male = rbinom(n=10, size=1, prob=0.5) )
make_one_linear_pred(m, data)
```

### Description


### Usage

```r
mod.jointly.generate.binary.normal(no.rows, no.bin, no.nor, prop.vec.bin,
mean.vec.nor, var.nor, corr.vec, adjust.corrs = TRUE)
```

### Arguments

- `no.rows`: Number of rows
- `no.bin`: Number of binary variables
- `no.nor`: Number of normal variables
- `prop.vec.bin`: Vector of parameters for binary variables
- `mean.vec.nor`: Vector of means for binary variables
- `var.nor`: Vector of variances for binary variables
- `corr.vec`: Vector of correlations
- `adjust.corrs`: Boolean indicating whether theoretically impossible correlations between a binary and a normal variable should be adjusted to their closest theoretically possible value.

### References

override_static  Override static variable

Description
An internal function not intended for the user. For static variables, overrides any time-varying values to ensure that they are actually static.

Usage
override_static(.static.var.name, .id.var.name = "id", .d, .obs)

Arguments
.static.var.name  Name of static variable.
.id.var.name  Name of variable defining clusters in dataset.
.d  Dataset
.obs  The number of observations per cluster.

Examples
# example with 10 subjects each with 3 observations
# generate sex in a way where it might vary within a subject
data = data.frame( id = rep(1:10, each=3),
                   male = rbinom( n=10*3, size=1, prob=0.5 ) )
override_static("male", "id", data, 3)

override_tbin_probs  Override probabilities for time-varying binary variables

Description
An internal function not intended for the user. For clusters assigned to have a given time-varying binary variable always equal to 0, overrides to 0 the corresponding proportion of observations with the binary variable equal to 1.

Usage
override_tbin_probs(mus0, n.TBins, n.OtherBins, zero = 1e-04)

Arguments
mus0  The matrix of cluster means.
n.TBins  Number of time-varying binary variables.
n.OtherBins  The number of static binary variables.
zero  A number very close to 0, but slightly larger.
Examples

# make example subject means matrix for 1 static binary,
# 1 time-varying binary, and 1 normal
# 50 subjects and 5 observations (latter plays into variance)
set.seed(451)
mus0 = mod.jointly.generate.binary.normal( no.rows = 50, no.bin = 2, no.nor = 2,
    prop.vec.bin = c( .5, .35 ),
    mean.vec.nor = c( .4, 100 ),
    var.nor = c( (0.4 * 0.6) / 5, 10 ),
    corr.vec = c(0.05, 0.08, 0, 0, -0.03, 0) )

# note that we have ever-users with non-zero propensities to be on drug: not okay
any( mus0[,1] == 0 & mus0[,3] != 0 )

# fix them
mus1 = override_tbin_probs( mus0, 1, 1 )

# all better!
any( mus1[,1] == 0 & mus1[,3] > 0.0001 )

---

params

An example parameters dataframe

Description

An example of how to set up the parameters dataframe.

Usage

params

Format

An object of class data.frame with 12 rows and 8 columns.

---

pcor

An example across-cluster correlation dataframe

Description

An example of how to set up the across-cluster correlation dataframe.

Usage

pcor

Format

An object of class data.frame with 9 rows and 9 columns.
**proportionize**

*Turn a number into a valid proportion*

**Description**

An internal function not intended for the user. Turns an arbitrary number into a valid proportion by setting the number equal to the closest value in [0,1].

**Usage**

```r
proportionize(x, zero = 1e-05, one = 0.999)
```

**Arguments**

- `x`: The number to be turned into a proportion.
- `zero`: A very small number that is just larger than 0.
- `one`: A number that is just smaller than 1.

**Examples**

```r
proportionize(-0.03)
proportionize(1.2)
proportionize(.63)
```

---

**upper_tri_vec**

*Turn symmetric matrix into vector*

**Description**

An internal function not intended for the user. Turns a matrix into a vector of the upper-triangular elements (arranged by row).

**Usage**

```r
upper_tri_vec(m)
```

**Arguments**

- `m`: Matrix

**Examples**

```r
# make a simple correlation matrix
x = rnorm(10); y = rnorm(10); z = rnorm(10)
mat = cor( data.frame(x,y,z) )

# turn into into vector
upper_tri_vec(mat)
```
wcor

An example within-cluster correlation dataframe

Description

An example of how to set up the within-cluster correlation dataframe.

Usage

wcor

Format

An object of class data.frame with 6 rows and 6 columns.
Index

* datasets
  cat.params, 4
  params, 12
  pcor, 12
  wcor, 14

add_one_categorical, 2
add_time_function_vars, 3

BN.rBound, 3

cat.params, 4
closest, 4
complete_parameters, 5

expand_matrix, 5
expand_subjects, 6

has_drug_suffix, 7

make_one_dataset, 7
make_one_linear_pred, 9
mod.jointly.generate.binary.normal, 10

override_static, 11
override_tbin_probs, 11

params, 12
pcor, 12
proportionize, 13

upper_tri_vec, 13

wcor, 14