

# Package ‘panelWranglR’

October 3, 2019

**Title** Panel Data Wrangling Tools

**Version** 1.2.13

**BugReports** <https://github.com/JSzitas/panelWranglR/issues>

**Description** Leading/lagging a panel, creating dummy variables, taking panel differences, looking for panel autocorrelations, and more. Implemented via a 'data.table' back end.

**License** GPL-3

**Depends** R (>= 3.2.0)

**Suggests** testthat (>= 2.1.0)

**Encoding** UTF-8

**LazyData** true

**URL** <https://github.com/JSzitas/panelWranglR>

**RoxygenNote** 6.1.1

**Imports** data.table, Hmisc, caret

**NeedsCompilation** no

**Author** Juraj Szitas [aut, cre]

**Maintainer** Juraj Szitas <szitas.juraj13@gmail.com>

**Repository** CRAN

**Date/Publication** 2019-10-03 08:30:02 UTC

## R topics documented:

corr_finder . . . . .	2
panel_collect . . . . .	2
panel_correl . . . . .	4
panel_diff . . . . .	5
panel_dummify . . . . .	6
panel_lag . . . . .	7
panel_lead . . . . .	9

<b>Index</b>	<b>11</b>
--------------	-----------

---

corr_finder	<i>Wrapper for find correlations</i>
-------------	--------------------------------------

---

**Description**

Just a helper function for `correl_panel`.

**Usage**

```
corr_finder(df, corr_cutoff)
```

**Arguments**

<code>df</code>	The dataframe to use.
<code>corr_cutoff</code>	The correlation cutoff to pass to <code>findCorrelations</code>

**Examples**

```
X_1 <- rnorm(1000)
X_2 <- rnorm(1000) + 0.6 * X_1
X_3 <- rnorm(1000) - 0.4 * X_1

data_fm <- do.call( cbind, list( X_1,
                                X_2,
                                X_3 ))

corr_finder( df = data_fm,
             corr_cutoff = 0.3 )
```

---

panel_collect	<i>Collect a panel, from wide to long</i>
---------------	-------------------------------------------

---

**Description**

Transforms cross sectional/time dummies to unified variables

**Usage**

```
panel_collect(data, cross.section = NULL, cross.section.columns = NULL,
             time.variable = NULL, time.variable.columns = NULL)
```

**Arguments**

**data**                    The panel to transform  
**cross.section**        The name of the transformed cross sectional variable supply as chracter.  
**cross.section.columns**  
                           The names of the columns indicating cross sections to collect.  
**time.variable**        The name of the transformed time variable supply as character.  
**time.variable.columns**  
                           The names of the columns indicating time variables to collect.

**Details**

For time variables named like "Time\_Var\_i" with arbitrary i, the program will check that all time variables are named using this convention, and strip this convention

**Value**

A collected data.table, with new columns constructed by collecting from the wide format.

**Examples**

```

x_1 <- rnorm( 10 )
cross_levels <- c( "AT", "DE" )
time <- seq(1:5)
time <- rep(time, 2)
geo_list <- list()
for(i in 1:length(cross_levels))
{
  geo <- rep( cross_levels[i],
             100 )
  geo_list[[i]] <- geo
}
geo <- unlist(geo_list)
geo <- as.data.frame(geo)

example_data <- cbind( time,
                      x_1 )
example_data <- as.data.frame(example_data)

example_data <- cbind( geo,
                      example_data)
names(example_data) <- c("geo", "time", "x_1")

# generate dummies using panel_dummify()
test_dummies <- panel_dummify( data = example_data,
                              cross.section = "geo",
                              time.variable = "time")
panel_collect( data = test_dummies,
              cross.section = "geo",
              cross.section.columns = c( "AT", "DE"))

```

---

 panel\_correl

*Panel linear combinations*


---

### Description

A function to find highly correlated variables in a panel of data, both by cross sections and by time dummies.

### Usage

```
panel_correl(data, cross.section = NULL, time.variable = NULL,
  corr.threshold = 0.7, autocorr.threshold = 0.5,
  cross.threshold = 0.7, select.cross.sections = NULL,
  select.time.periods = NULL)
```

### Arguments

data	The data to use, a data.frame or a data.table.
cross.section	The name of the cross sectional variable.
time.variable	The name of the time variable.
corr.threshold	The correlation threshold for finding significant correlations in the base specification, disregarding time or cross sectional dependencies.
autocorr.threshold	The correlation threshold for autocorrelation (splitting the pooled panel into cross sections).
cross.threshold	The correlation threshold for finding significant correlations in the cross sections.
select.cross.sections	An optional subset of cross sectional units.
select.time.periods	An optional subset of time periods

### Examples

```
x_1 <- rnorm( 100 )
x_2 <- rnorm( 100 ) + 0.5 * x_1
cross_levels <- c( "AT", "DE" )
time <- seq(1:50)
time <- rep(time, 2)
geo_list <- list()
for(i in 1:length(cross_levels))
{ geo <- rep( cross_levels[i], 50 )
  geo_list[[i]] <- geo }
geo <- unlist(geo_list)
geo <- as.data.frame(geo)
```

```

example_data <- do.call ( cbind, list( time, x_1, x_2))
example_data <- as.data.frame(example_data)
example_data <- cbind( geo,
                      example_data)

                      names(example_data) <- c("geo", "time", "x_1",
                                                "x_2")

panel_correl( data = example_data,
              cross.section = "geo",
              time.variable = "time",
              corr.threshold = 0.2,
              autocorr.threshold = 0.5,
              cross.threshold = 0.1)

```

---

panel\_diff

*Tidy panel differencing*


---

### Description

Efficient, tidy panel differencing

### Usage

```

panel_diff(data, cross.section, time.variable = NULL, diff.order = 1,
           lags = 1, variables.selected = NULL, keep.original = FALSE)

```

### Arguments

data	The data input, anything coercible to a data.table.
cross.section	The cross section argument, see examples.
time.variable	The variable to indicate time in your panel. Defaults to NULL, though it is recommended to have a time variable.
diff.order	The number of applications of the difference operator to use in panel differencing. Defaults to 1.
lags	The number of lags to use for differences. Defaults to 1.
variables.selected	A variable selection for variables to difference, defaults to NULL and differences ALL variables.
keep.original	Whether to keep the original undifferenced data, defaults to FALSE.

### Details

Works on a full data.table backend for maximum speed wherever possible.

**Value**

The differenced data.table which contains either only the differenced variables, or also the original variables.

**Examples**

```
X <- matrix(rnorm(4000),800,5)
tim <- seq(1:400)
geo_AT <- rep(c("AT"), length = 400)
geo_NO <- rep(c("NO"), length = 400)
both_vec_1 <- cbind(tim,geo_NO)
both_vec_2 <- cbind(tim,geo_AT)
both <- rbind(both_vec_1,both_vec_2)
names(both[, "geo_NO"]) <- "geo"
X <- cbind(both,X)

panel_diff(data = X,
           cross.section = "geo_NO",
           time.variable = "tim",
           diff.order = 1,
           lags = 1,
           variables.selected = c("V3","V4"),
           keep.original = TRUE)
```

---

panel\_dummify

*Tidy time/variable dummies for panel data*

---

**Description**

A simple function to dummify cross sections or time variables in panel data.

**Usage**

```
panel_dummify(data, cross.section = NULL, time.variable = NULL)
```

**Arguments**

`data`                The panel to dummify  
`cross.section`      The cross section variable in the panel. Defaults to NULL.  
`time.variable`      The variable to indicate time in your panel. Defaults to NULL.

**Details**

The encoding is binary, whether this is more appropriate than using a factor variable is up to the user.

**Value**

A new data.table, with the original variables to dummify removed, and new dummy columns included.

**Examples**

```
x_1 <- rnorm( 10 )
cross_levels <- c( "AT", "DE" )
time <- seq(1:5)
time <- rep(time, 2)
geo_list <- list()
for(i in 1:length(cross_levels))
{
  geo <- rep( cross_levels[i],
             100 )
  geo_list[[i]] <- geo
}
geo <- unlist(geo_list)
geo <- as.data.frame(geo)

example_data <- cbind( time,
                      x_1 )
example_data <- as.data.frame(example_data)

example_data <- cbind( geo,
                      example_data)
names(example_data) <- c("geo", "time", "x_1")

test_dummies <- panel_dummify( data = example_data,
                              cross.section = "geo",
                              time.variable = "time")
```

---

panel\_lag

*Tidy panel lagging*

---

**Description**

Efficient, tidy panel lagging

**Usage**

```
panel_lag(data, cross.section, time.variable = NULL, lags = 1,
          variables.selected = NULL, keep.original = TRUE)
```

## Arguments

<code>data</code>	The data input, anything coercible to a <code>data.table</code> .
<code>cross.section</code>	The cross section argument, see examples.
<code>time.variable</code>	The variable to indicate time in your panel. Defaults to <code>NULL</code> , though it is recommended to have a time variable.
<code>lags</code>	The lags to use in panel lagging.
<code>variables.selected</code>	A variable selection for variables to lag, defaults to <code>NULL</code> and lags ALL variables.
<code>keep.original</code>	Whether to keep the original unlagged data, defaults to <code>TRUE</code> .

## Details

Works on a full `data.table` backend for maximum speed wherever possible.

## Value

The lagged `data.table` which contains either only the lagged variables, or also the original variables.

## Examples

```
X <- matrix(rnorm(4000), 800, 5)
tim <- seq(1:400)
geo_AT <- rep(c("AT"), length = 400)
geo_NO <- rep(c("NO"), length = 400)
both_vec_1 <- cbind(tim, geo_NO)
both_vec_2 <- cbind(tim, geo_AT)
both <- rbind(both_vec_1, both_vec_2)
names(both[, "geo_NO"]) <- "geo"
X <- cbind(both, X)

panel_lag(data = X,
          cross.section = "geo_NO",
          time.variable = "tim",
          lags = 5,
          variables.selected = c("V5", "tim", "V7"),
          keep.original = TRUE)
```



---

panel_lead	<i>Tidy panel leading</i>
------------	---------------------------

---

### Description

Efficient, tidy panel leading

### Usage

```
panel_lead(data, cross.section, time.variable = NULL, leads = 1,
           variables.selected = NULL, keep.original = TRUE)
```

### Arguments

data	The data input, anything coercible to a data.table.
cross.section	The cross section argument, see examples.
time.variable	The variable to indicate time in your panel. Defaults to NULL, though it is recommended to have a time variable.
leads	The leads to use in panel leading.
variables.selected	A variable selection for variables to lead, defaults to NULL and leads ALL variables.
keep.original	Whether to keep the original unleaded data, defaults to TRUE.

### Details

Works on a full data.table backend for maximum speed wherever possible.

### Value

The leading data.table which contains either only the leading variables, or also the original variables.

### Examples

```
X <- matrix(rnorm(4000),800,5)
tim <- seq(1:400)
geo_AT <- rep(c("AT"), length = 400)
geo_NO <- rep(c("NO"), length = 400)
both_vec_1 <- cbind(tim,geo_NO)
both_vec_2 <- cbind(tim,geo_AT)
both <- rbind(both_vec_1,both_vec_2)
names(both[, "geo_NO"]) <- "geo"
X <- cbind(both,X)

panel_lead(data = X,
```

```
cross.section = "geo_NO",  
time.variable = "tim",  
leads = 5,  
variables.selected = c("V5", "tim", "V7"),  
keep.original = TRUE)
```

# Index

`corr_finder`, 2

`panel_collect`, 2

`panel_correl`, 4

`panel_diff`, 5

`panel_dummify`, 6

`panel_lag`, 7

`panel_lead`, 9