Package: bdsm (via r-universe)

March 13, 2025

Title Bayesian Dynamic Systems Modeling

Version 0.1.0

Description Implements methods for building and analyzing models based on panel data as described in the paper by Moral-Benito (2013, <doi:10.1080/07350015.2013.818003>). The package provides functions to estimate dynamic panel data models and analyze the results of the estimation.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Suggests spelling, testthat (>= 3.0.0)

Config/testthat/edition 3

Imports dplyr, magrittr, optimbase, parallel, rje, rlang, rootSolve, stats, tidyr, tidyselect

Depends R (>= 2.10)

Language en-US

NeedsCompilation no

Author Mateusz Wyszynski [aut], Marcin Dubel [ctb, cre], Krzysztof Beck [ctb]

Maintainer Marcin Dubel <marcindubel@gmail.com>

Date/Publication 2025-01-21 15:30:02 UTC

Config/pak/sysreqs libicu-dev

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

RemoteUrl https://github.com/cran/bdsm

RemoteRef HEAD

RemoteSha 39952a582364d233408e454b5c5ec4066689c2e3

Contents

`bma_summary` *Summary of a model space*

Description

A summary of a given model space is prepared. This include things such as posterior inclusion probability (PIP), posterior mean and so on. This is the core function of the package, because it allows to make assessments and decisions about the parameters and models.

Usage

```
bma_summary(
  df,
  dep_var_col,
  timestamp_col,
  entity_col,
  model_space,
  exact_value = TRUE,
```

```
  model_prior = "uniform",
  run_parallel = FALSE
)
```

Arguments

df	Data frame with data for the SEM analysis.
dep_var_col	Column with the dependent variable
timestamp_col	The name of the column with timestamps
entity_col	Column with entities (e.g. countries)
model_space	A matrix (with named rows) with each column corresponding to a model. Each column specifies model parameters. Compare with optimal_model_space
exact_value	Whether the exact value of the likelihood should be computed (`TRUE`) or just the proportional part (`FALSE`). Check SEM_likelihood for details.
model_prior	Which model prior to use. For now there are two options: `'uniform'` and `'binomial-beta'`. Default is `'uniform'`.
run_parallel	If `TRUE` the optimization is run in parallel using the parApply function. If `FALSE` (default value) the base apply function is used. Note that using the parallel computing requires setting the default cluster. See README.

Value

List of parameters describing analyzed models

Examples

```
library(magrittr)

data_prepared <- economic_growth[,1:7] %>%
   feature_standardization(timestamp_col = year, entity_col = country) %>%
   feature_standardization(timestamp_col = year, entity_col = country,
                           cross_sectional = TRUE, scale = FALSE)


bma_result <- bma_summary(df = data_prepared, dep_var_col = gdp,
                          timestamp_col = year, entity_col = country,
                          model_space = economic_growth_ms)
```

`economic_growth` *Economic Growth Data*

Description

Data used in Growth Empirics in Panel Data under Model Uncertainty and Weak Exogeneity (Moral-Benito, 2016, Journal of Applied Econometrics).

Usage

```
economic_growth
```

Format

`economic_growth`:
A data frame with 365 rows and 12 columns:

year Year
country Country ID
gdp Logarithm of GDP per capita (2000 US dollars at PP)
ish Ratio of real domestic investment to GDP
sed Stock of years of secondary education in the total population
pgrw Average growth rate of population
pop Population in millions of people
ipr Purchasing-power-parity numbers for investment goods
opem Exports plus imports as a share of GDP
gsh Ratio of government consumption to GDP
lnlex Logarithm of the life expectancy at birth
polity Composite index given by the democracy score minus the autocracy score

Source

http://qed.econ.queensu.ca/jae/datasets/moral-benito001/

---

economic_growth_bma_params *Example Approximate Summary of Parameters of Interest Based on Model Space*

Description

A matrix representing the summary of parameters computed with `parameters_summary` based on the `economic_growth_ms` model space. TODO: describe the matrix properly after cleaning up the code of the function `parameters_summary`.

Usage

```
economic_growth_bma_params
```

Format

`economic_growth_bma_params`:
A double matrix with 5 rows and 8 columns

economic_growth_liks *Example Approximate Likelihoods Summary based on Model Space*

Description

A matrix representing the summary of likelihoods computed with likelihoods_summary based on the economic_growth_ms model space. The matrix contains likelihoods, standard deviations and robust standard deviations

Usage

```
economic_growth_liks
```

Format

economic_growth_stds**:**

A double matrix with 11 rows and 16 columns.

first row Likelihoods for the models

second row Almost 1/2 * BIC_k as in Raftery's Bayesian Model Selection in Social Research eq. 19.

rows 3-7 Standard deviations

rows 8-12 Robust standard deviations

economic_growth_ms *Example Model Space*

Description

A matrix representing the model space built using subset of regressors from the economic_growth dataset. The included regressors are ish, sed, pgrw and pop. Therefore the model space contains 2^4 = 16 models (columns).

Usage

```
economic_growth_ms
```

Format

economic_growth_ms**:**

A double matrix with 51 rows and 16 columns.

economic_growth_ms_full_proj_const

Full Model Space with Constant Projection Matrix

Description

A matrix representing the model space built using all regressors from the `economic_growth` dataset. Therefore the model space contains `2^9 = 512` models (columns). The same projection matrix is used for each model.

Usage

```
economic_growth_ms_full_proj_const
```

Format

`economic_growth_ms_full_proj_const`:
A double matrix with 106 rows and 512 columns.

Details

TODO: to avoid NaNs when computing estimates of standard deviations, the step size in the hessian function has to be increased to 1e-2. This is most likely cause by the fact that the likelihood values are much closer to each other after the correction for the projection matrix is introduced. Hence we have to either increase the relative tolerance of the optimization algorithm or loosen the precision when computing approximate hessian.

economic_growth_ms_full_proj_var

Full Model Space with Varying Projection Matrix

Description

A matrix representing the model space built using all regressors from the `economic_growth` dataset. Therefore the model space contains `2^9 = 512` models (columns). This model space generates Posterior Inclusion Probabilities which are consistent with the results presented by Moral-Benito. The original results were approximated up to the 4th decimal place. The results obtained using this model space lead to exactly the same approximations. A different projection matrix is used for each model.

Usage

```
economic_growth_ms_full_proj_var
```

Format

`economic_growth_ms_full_proj_var`:

A double matrix with 106 rows and 512 columns.

---

exogenous_matrix — *Matrix with exogenous variables for SEM representation*

Description

Create matrix which contains exogenous variables used in the Simultaneous Equations Model (SEM) representation. Currently these are: dependent variable from the lowest time stamp and regressors from the second lowest time stamp. The matrix is then used to compute likelihood for SEM analysis.

Usage

```
exogenous_matrix(df, timestamp_col, entity_col, dep_var_col)
```

Arguments

`df`	Data frame with data for the SEM analysis.
`timestamp_col`	Column which determines time periods. For now only natural numbers can be used as timestamps
`entity_col`	Column which determines entities (e.g. countries, people)
`dep_var_col`	Column with dependent variable

Value

Matrix of size N x k+1 where N is the number of entities considered and k is the number of chosen regressors

Examples

```
set.seed(1)
df <- data.frame(
  entities = rep(1:4, 5),
  times = rep(seq(1960, 2000, 10), each = 4),
  dep_var = stats::rnorm(20), a = stats::rnorm(20), b = stats::rnorm(20)
)
exogenous_matrix(df, times, entities, dep_var)
```

`feature_standardization` *Perform feature standarization*

Description

This function performs feature standarization (also known as z-score normalization), i.e. the features are centered around the mean and scaled with standard deviation.

Usage

```
feature_standardization(
  df,
  timestamp_col,
  entity_col,
  cross_sectional = FALSE,
  scale = TRUE
)
```

Arguments

`df`	Dataframe with data that should be prepared for LIML estimation
`timestamp_col`	Column with timestamps (e.g. years)
`entity_col`	Column with entities (e.g. countries)
`cross_sectional`	Whether to perform feature standardization within cross sections
`scale`	Whether to divide by the standard deviation `TRUE` or not `FALSE`. Default is `TRUE`.

Value

A dataframe with standardized features

Examples

```
df <- data.frame(
  year = c(2000, 2001, 2002, 2003, 2004),
  country = c("A", "A", "B", "B", "C"),
  gdp = c(1, 2, 3, 4, 5),
  ish = c(2, 3, 4, 5, 6),
  sed = c(3, 4, 5, 6, 7)
)

feature_standardization(df, year, country)
```

hessian *Hessian matrix*

Description

Creates the hessian matrix for a given likelihood function.

Usage

```
hessian(lik, theta, ...)
```

Arguments

`lik`	function
`theta`	kx1 matrix
`...`	other parameters passed to `lik` function.

Value

Hessian kxk matrix where k is the number of parameters included in the theta matrix

Examples

```
lik <- function(theta) {
 return(theta[1]^2 + theta[2]^2)
}

hessian(lik, c(1, 1))
```

initialize_model_space *Initialize model space matrix*

Description

This function builds a representation of the model space, by creating a dataframe where each column represents values of the parameters for a given model. Real value means that the parameter is included in the model. A parameter not present in the model is marked as `NA`.

Usage

```
initialize_model_space(
  df,
  timestamp_col,
  entity_col,
  dep_var_col,
  init_value = 1
)
```

Arguments

`df`	Data frame with data for the SEM analysis.
`timestamp_col`	Column which determines time periods. For now only natural numbers can be used as timestamps
`entity_col`	Column which determines entities (e.g. countries, people)
`dep_var_col`	Column with dependent variable
`init_value`	Initial value for parameters present in the model. Default is `1`.

Details

Currently the set of features is assumed to be all columns which remain after excluding `timestamp_col`, `entity_col` and `dep_var_col`.

A power set of all possible exclusions of linear dependence on the given feature is created, i.e. if there are 4 features we end up with 2^4 possible models (for each model we independently decide whether to include or not a feature).

Value

matrix of model parameters

Examples

```
library(magrittr)

data_prepared <- economic_growth[,1:7] %>%
  feature_standardization(timestamp_col = year, entity_col = country) %>%
  feature_standardization(timestamp_col = year, entity_col = country,
                          cross_sectional = TRUE, scale = FALSE)

initialize_model_space(data_prepared, year, country, gdp)
```

join_lagged_col — *Dataframe with no lagged column*

Description

This function allows to turn data in the format with lagged values for a chosen column (i.e. there are two columns with the same quantity, but one column is lagged in time) into the format with just one column

Usage

```
join_lagged_col(
  df,
  col,
  col_lagged,
  timestamp_col,
  entity_col,
  timestep = NULL
)
```

Arguments

`df`	Dataframe with data with a column with lagged values
`col`	Column with quantity not lagged
`col_lagged`	Column with the same quantity as `col`, but the values are lagged in time
`timestamp_col`	Column with timestamps (e.g. years)
`entity_col`	Column with entities (e.g. countries)
`timestep`	Difference between timestamps (e.g. 10)

Value

A dataframe with two columns merged, i.e. just one column with the desired quantity is left.

Examples

```
df <- data.frame(
  year = c(2000, 2001, 2002, 2003, 2004),
  country = c("A", "A", "B", "B", "C"),
  gdp = c(1, 2, 3, 4, 5),
  gdp_lagged = c(NA, 1, 2, 3, 4)
)

join_lagged_col(df, gdp, gdp_lagged, year, country, 1)
```

likelihoods_summary — *Approximate standard deviations for the models*

Description

Approximate standard deviations are computed for the models in the given model space. Two versions are computed.

Usage

```
likelihoods_summary(
  df,
  dep_var_col,
  timestamp_col,
  entity_col,
  model_space,
  exact_value = TRUE,
  model_prior = "uniform",
  run_parallel = FALSE
)
```

Arguments

`df`	Data frame with data for the SEM analysis.
`dep_var_col`	Column with the dependent variable
`timestamp_col`	The name of the column with timestamps
`entity_col`	Column with entities (e.g. countries)
`model_space`	A matrix (with named rows) with each column corresponding to a model. Each column specifies model parameters. Compare with optimal_model_space
`exact_value`	Whether the exact value of the likelihood should be computed (`TRUE`) or just the proportional part (`FALSE`). Check SEM_likelihood for details.
`model_prior`	Which model prior to use. For now there are two options: `'uniform'` and `'binomial-beta'`. Default is `'uniform'`.
`run_parallel`	If `TRUE` the optimization is run in parallel using the parApply function. If `FALSE` (default value) the base apply function is used. Note that using the parallel computing requires setting the default cluster. See README.

Value

Matrix with columns describing likelihood and standard deviations for each model. The first row is the likelihood for the model (computed using the parameters in the provided model space). The second row is almost 1/2 * BIC_k as in Raftery's Bayesian Model Selection in Social Research eq. 19 (see TODO in the code below). The third row is model posterior probability. Then there are rows with standard deviations for each parameter. After that we have rows with robust standard deviation (not sure yet what exactly "robust" means).

Examples

```
data_centered_scaled <-
  feature_standardization(df = bdsm::economic_growth[,1:7],
                          timestamp_col = year, entity_col = country)
data_cross_sectional_standarized <-
  feature_standardization(df = data_centered_scaled, timestamp_col = year,
                          entity_col = country, cross_sectional = TRUE,
                          scale = FALSE)

  likelihoods_summary(df = data_cross_sectional_standarized,
                      dep_var_col = gdp, timestamp_col = year,
                      entity_col = country, model_space = economic_growth_ms)
```

---

matrices_from_df *List of matrices for SEM model*

Description

List of matrices for SEM model

Usage

```
matrices_from_df(
  df,
  timestamp_col,
  entity_col,
  dep_var_col,
  lin_related_regressors = NULL,
  which_matrices = c("Y1", "Y2", "Z", "cur_Y2", "cur_Z", "res_maker_matrix")
)
```

Arguments

df	Dataframe with data for the likelihood computations.
timestamp_col	Column which determines time stamps. For now only natural numbers can be used.
entity_col	Column which determines entities (e.g. countries, people)
dep_var_col	Column with dependent variable
lin_related_regressors	Vector of strings of column names. Which subset of regressors is in non trivial linear relation with the dependent variable (dep_var_col). In other words regressors with non-zero beta parameters.
which_matrices	character vector with names of matrices which should be computed. Possible matrices are "Y1", "Y2", "Z", "cur_Y2", "cur_Z", "res_maker_matrix". Default is c("Y1", "Y2", "Z", "cur_Y2","cur_Z", "res_maker_matrix") in which case all possible matrices are generated

Value

Named list with matrices as its elements

Examples

```
matrices_from_df(economic_growth, year, country, gdp, c("pop", "sed"),
                 c("Y1", "Y2"))
```

---

optimal_model_space   *Finds MLE parameters for each model in the given model space*

Description

Given a dataset and an initial value for parameters, initializes a model space with parameters equal to initial value for each model. Then for each model performs a numerical optimization and finds parameters which maximize the likelihood.

Usage

```
optimal_model_space(
  df,
  timestamp_col,
  entity_col,
  dep_var_col,
  init_value,
  exact_value = TRUE,
  run_parallel = FALSE,
 control = list(trace = 2, maxit = 10000, fnscale = -1, REPORT = 100, scale = 0.05)
)
```

Arguments

`df`	Data frame with data for the SEM analysis.
`timestamp_col`	The name of the column with time stamps
`entity_col`	Column with entities (e.g. countries)
`dep_var_col`	Column with the dependent variable
`init_value`	The value with which the model space will be initialized. This will be the starting point for the numerical optimization.
`exact_value`	Whether the exact value of the likelihood should be computed (`TRUE`) or just the proportional part (`FALSE`). Check SEM_likelihood for details.
`run_parallel`	If `TRUE` the optimization is run in parallel using the parApply function. If `FALSE` (default value) the base apply function is used. Note that using the parallel computing requires setting the default cluster. See README.
`control`	a list of control parameters for the optimization which are passed to optim. Default is `list(trace = 2, maxit = 10000, fnscale = -1, REPORT = 100, scale = 0.05)`, but note that `scale` is used only for adjusting the `parscale` element added later in the function code.

Value

List of parameters describing analyzed models

Examples

```
library(magrittr)

data_prepared <- economic_growth[,1:7] %>%
   feature_standardization(timestamp_col = year, entity_col = country) %>%
   feature_standardization(timestamp_col = year, entity_col = country,
                           cross_sectional = TRUE, scale = FALSE)

model_space <- optimal_model_space(df = data_prepared, dep_var_col = gdp,
                                   timestamp_col = year, entity_col = country,
                                   init_value = 0.5)
```

`parameters_summary` *BMA summary for parameters of interest*

Description

TODO This is just the code previously present in the morel-benito.R script wrapped as a function (to get rid of the script). Well written code and docs are still needed

Usage

```
parameters_summary(
  regressors,
  bet,
  pvarh,
  pvarr,
  fy,
  fyt,
  ppmsize,
  cout,
  nts,
  pts,
  variables_n
)
```

Arguments

`regressors`	TODO
`bet`	TODO
`pvarh`	TODO

`pvarr`	TODO
`fy`	TODO
`fyt`	TODO
`ppmsize`	TODO
`cout`	TODO
`nts`	TODO (negatives)
`pts`	TODO (positives)
`variables_n`	TODO

Value

TODO dataframe with results

`regressor_names_from_params_vector`
Helper function to extract names from a vector defining a model

---

Description

For now it is assumed that we can only exclude linear relationships between regressors and the dependent variable.

Usage

```
regressor_names_from_params_vector(params)
```

Arguments

`params`	a vector with parameters describing the model

Details

The vector needs to have named rows, i.e. it is assumed it comes from a model space (see initialize_model_space for details).

Value

Names of regressors which are assumed to be linearly connected with dependent variable within the model described by the `params` vector.

Examples

```
params <- c(alpha = 1, beta_gdp = 1, beta_gdp_lagged = 1, phi_0 = 1, err_var = 1)
regressor_names_from_params_vector(params)
```

residual_maker_matrix *Residual Maker Matrix*

Description

Create residual maker matrix from a given matrix m. See article about projection matrix on the Wikipedia.

Usage

```
residual_maker_matrix(m)
```

Arguments

m	Matrix

Value

M x M matrix where M is the number of rows in the m matrix.

Examples

```
residual_maker_matrix(matrix(c(1,2,3,4), nrow = 2))
```

SEM_B_matrix *Coefficients matrix for SEM representation*

Description

Create coefficients matrix for Simultaneous Equations Model (SEM) representation.

Usage

```
SEM_B_matrix(alpha, periods_n, beta = c())
```

Arguments

alpha	numeric
periods_n	integer
beta	numeric vector. Default is c() for no regressors case.

Value

List with two matrices B11 and B12

Examples

```
SEM_B_matrix(3, 4, 4:6)
```

SEM_C_matrix — *Coefficients matrix for initial conditions*

Description

Create matrix for Simultaneous Equations Model (SEM) representation with coefficients placed next to initial values of regressors, dependent variable and country-specific time-invariant variables.

Usage

```
SEM_C_matrix(alpha, phi_0, periods_n, beta = c(), phi_1 = c())
```

Arguments

alpha	numeric
phi_0	numeric
periods_n	numeric
beta	numeric vector. Default is c() for no regressors case.
phi_1	numeric vector. Default is c() for no regressors case.

Value

matrix

Examples

```
alpha <- 9
phi_0 <- 19
beta <- 11:15
phi_1 <- 21:25
periods_n <- 4
SEM_C_matrix(alpha, phi_0, periods_n, beta, phi_1)
```

SEM_dep_var_matrix — *Matrix with dependent variable data for SEM representation*

Description

Create matrix which contains dependent variable data used in the Simultaneous Equations Model (SEM) representation on the left hand side of the equations. The matrix contains the data for time periods greater than or equal to the second lowest time stamp. The matrix is then used to compute likelihood for SEM analysis.

Usage

```
SEM_dep_var_matrix(df, timestamp_col, entity_col, dep_var_col)
```

Arguments

df	Data frame with data for the SEM analysis.
timestamp_col	Column which determines time periods. For now only natural numbers can be used as timestamps
entity_col	Column which determines entities (e.g. countries, people)
dep_var_col	Column with dependent variable

Value

Matrix of size N x T where N is the number of entities considered and T is the number of periods greater than or equal to the second lowest time stamp.

Examples

```
set.seed(1)
df <- data.frame(
  entities = rep(1:4, 5),
  times = rep(seq(1960, 2000, 10), each = 4),
  dep_var = stats::rnorm(20), a = stats::rnorm(20), b = stats::rnorm(20)
)
SEM_dep_var_matrix(df, times, entities, dep_var)
```

SEM_likelihood *Likelihood for the SEM model*

Description

Likelihood for the SEM model

Usage

```
SEM_likelihood(
  params,
  data,
  timestamp_col,
  entity_col,
  dep_var_col,
  lin_related_regressors = NULL,
  per_entity = FALSE,
  exact_value = TRUE
)
```

Arguments

`params` Parameters describing the model. Can be either a vector or a list with named parameters. See 'Details'

`data` Data for the likelihood computations. Can be either a list of matrices or a dataframe. If the dataframe, additional parameters are required to build the matrices within the function.

`timestamp_col` Column which determines time stamps. For now only natural numbers can be used.

`entity_col` Column which determines entities (e.g. countries, people)

`dep_var_col` Column with dependent variable

`lin_related_regressors`
Which subset of columns should be used as regressors for the current model. In other words `regressors` are the total set of regressors and `lin_related_regressors` are the ones for which linear relation is not set to zero for a given model.

`per_entity` Whether to compute overall likelihood or a vector of likelihoods with per entity value

`exact_value` Whether the exact value of the likelihood should be computed (`TRUE`) or just the proportional part (`FALSE`). Currently `TRUE` adds: 1. a normalization constant coming from Gaussian distribution, 2. a term disappearing during likelihood simplification in Likelihood-based Estimation of Dynamic Panels with Predetermined Regressors by Moral-Benito (see Appendix A.1). The latter happens when transitioning from equation (47) to equation (48), in step 2: the term `trace(HG_22)` is dropped, because it can be assumed to be constant from Moral-Benito perspective. To get the exact value of the likelihood we have to take this term into account.

Details

The `params` argument is a list that should contain the following components:

`alpha` scalar value which determines linear dependence on lagged dependent variable

`phi_0` scalar value which determines linear dependence on the value of dependent variable at the lowest time stamp

`err_var` scalar value which determines classical error component (Sigma11 matrix, sigma_epsilon^2)

`dep_vars` double vector of length equal to the number of time stamps (i.e. time stamps greater than or equal to the second lowest time stamp)

`beta` double vector which determines the linear dependence on regressors different than the lagged dependent variable; The vector should have length equal to the number of regressors.

`phi_1` double vector which determines the linear dependence on initial values of regressors different than the lagged dependent variable; The vector should have length equal to the number of regressors.

`phis` double vector which together with `psis` determines upper right and bottom left part of the covariance matrix; The vector should have length equal to the number of regressors times number of time stamps minus 1, i.e. `regressors_n * (periods_n - 1)`

`psis` double vector which together with `psis` determines upper right and bottom left part of the covariance matrix; The vector should have length equal to the number of regressors times number of time stamps minus 1 times number of time stamps divided by 2, i.e. `regressors_n * (periods_n - 1) * periods_n / 2`

Value

The value of the likelihood for SEM model (or a part of interest of the likelihood)

Examples

```
set.seed(1)
df <- data.frame(
  entities = rep(1:4, 5),
  times = rep(seq(1960, 2000, 10), each = 4),
  dep_var = stats::rnorm(20), a = stats::rnorm(20), b = stats::rnorm(20)
)
df <-
  feature_standardization(df, timestamp_col = times, entity_col = entities)
SEM_likelihood(0.5, df, times, entities, dep_var)
```

SEM_psi_matrix   *Matrix with psi parameters for SEM representation*

Description

Matrix with psi parameters for SEM representation

Usage

```
SEM_psi_matrix(psis, timestamps_n, features_n)
```

Arguments

`psis`	double vector with psi parameter values
`timestamps_n`	number of time stamps (e.g. years)
`features_n`	number of features (e.g. population size, investment rate)

Value

A matrix with `timestamps_n` rows and `(timestamps_n - 1) * feature_n` columns. Psis are filled in row by row in a block manner, i.e. blocks of size `feature_n` are placed next to each other

Examples

```
SEM_psi_matrix(1:30, 4, 5)
```

`SEM_regressors_matrix` *Matrix with regressors data for SEM representation*

Description

Create matrix which contains regressors data used in the Simultaneous Equations Model (SEM) representation on the left hand side of the equations. The matrix contains regressors data for time periods greater than or equal to the second lowest time stamp. The matrix is then used to compute likelihood for SEM analysis.

Usage

```
SEM_regressors_matrix(df, timestamp_col, entity_col, dep_var_col)
```

Arguments

`df`	Data frame with data for the SEM analysis.
`timestamp_col`	Column which determines time periods. For now only natural numbers can be used as timestamps
`entity_col`	Column which determines entities (e.g. countries, people)
`dep_var_col`	Column with dependent variable

Value

Matrix of size N x (T-1)*k where N is the number of entities considered, T is the number of periods greater than or equal to the second lowest time stamp and k is the number of chosen regressors. If there are no regressors returns `NULL`.

Examples

```
set.seed(1)
df <- data.frame(
  entities = rep(1:4, 5),
  times = rep(seq(1960, 2000, 10), each = 4),
  dep_var = stats::rnorm(20), a = stats::rnorm(20), b = stats::rnorm(20)
)
SEM_regressors_matrix(df, times, entities, dep_var)
```

SEM_sigma_matrix *Covariance matrix for SEM representation*

Description

Create covariance matrix for Simultaneous Equations Model (SEM) representation. Only the part necessary to compute concentrated likelihood function is computed (cf. Appendix in the Moral-Benito paper)

Usage

```
SEM_sigma_matrix(err_var, dep_vars, phis = c(), psis = c())
```

Arguments

err_var	numeric
dep_vars	numeric vector
phis	numeric vector
psis	numeric vector

Value

List with two matrices Sigma11 and Sigma12

Examples

```
err_var <- 1
dep_vars <- c(2, 2, 2, 2)
phis <- c(10, 10, 20, 20, 30, 30)
psis <- c(101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112)
SEM_sigma_matrix(err_var, dep_vars, phis, psis)
```

Index