Package 'badp'

August 20, 2026

Title Bayesian Averaging for Dynamic Panels

Version 0.6.0

Description Implements Bayesian model averaging for dynamic panels with weakly exogenous regressors 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

URL https://badp-project.github.io/badp/, https://github.com/badp-project/badp

BugReports https://github.com/badp-project/badp/issues

Encoding UTF-8

LazyData true

Suggests pkgdown, rmarkdown, spelling, testthat (>= 3.0.0)

Config/testthat/edition 3

Imports dplyr, ggplot2, grid, gridExtra, knitr, magrittr, parallel, patchwork (>= 1.1.0), pbapply, rje, rlang, RTMB (>= 1.6), stats, tibble, tidyr, tidyselect

VignetteBuilder knitr

Depends R (>= 4.4)

Language en-US

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Krzysztof Beck [aut], Piotr Cukier [aut], Marcin Dubel [aut, cre], Mariusz Szczepanczyk [aut], Mateusz Wyszynski [aut]

Maintainer Marcin Dubel <marcindubel@gmail.com>

Repository CRAN

Date/Publication 2026-08-20 03:40:03 UTC

Contents

`best_models` *Table with the best models according to one of the posterior criteria*

Description

This function ranks the best models according to posterior model probabilities calculated using one of the available model priors: binomial or beta-binomial. It returns three types of tables in three different formats: an inclusion table, where 1 indicates that a regressor is included in the model and 0 indicates that it is excluded; an estimation results table, which displays the best models and their estimation output, including point estimates, standard errors, and significance levels; and an estimation results table with robust standard errors.

Usage

```
best_models(
  bma_list,
  prior = "binomial",
  best = 5,
  round = 3,
  estimate = TRUE,
  robust = TRUE
)
```

Arguments

bma_list	An object of class badp_bma, typically returned by bma.
prior	Character string specifying the model prior used for the ranking. Options are "binomial" (default) or "beta" (binomial-beta). Models are ranked by the posterior model probability computed under the chosen prior.
best	Integer. The number of best models to display (default: 5).
round	Integer indicating the decimal place to which numbers in the tables should be rounded (default: 3).
estimate	A parameter with values TRUE or FALSE indicating which table should be displayed when TRUE - table with the estimation results FALSE - table with the inclusion of regressors in the best models
robust	A parameter with values TRUE or FALSE indicating which type of standard errors should be displayed when the function finishes calculations. Works only if estimate = TRUE. Works well when best is small. TRUE - robust standard errors FALSE - regular standard errors

Value

A list with best_models objects:

1. matrix with inclusion of the regressors in the best models
2. matrix with estimation output in the best models with regular standard errors
3. matrix with estimation output in the best models with robust standard errors
4. knitr_kable table with inclusion of the regressors in the best models (the best for the display on the console - up to 11 models)
5. knitr_kable table with estimation output in the best models with regular standard errors (the best for the display on the console - up to 6 models)

6. knitr_kable table with estimation output in the best models with robust standard errors (the best for the display on the console - up to 6 models)

7. gTree table with inclusion of the regressors in the best models (displayed as a plot). Use grid::grid.draw() to display.

8. gTree table with estimation output in the best models with regular standard errors (displayed as a plot). Use grid::grid.draw() to display.
9. gTree table with estimation output in the best models with robust standard errors (displayed as a plot). Use grid::grid.draw() to display.

Examples

```
library(magrittr)

data_prepared <- badp::economic_growth[, 1:6] %>%
  badp::feature_standardization(
    excluded_cols = c(country, year, gdp)
  ) %>%
  badp::feature_standardization(
    group_by_col  = year,
    excluded_cols = country,
    scale         = FALSE
  )

bma_results <- bma(
  model_space = badp::small_model_space,
  round       = 3,
  dilution    = 0
)

best_5_models <- best_models(bma_results, prior = "binomial", best = 5, estimate = TRUE)
```

---

bma *Calculation of the bma object*

Description

This function calculates BMA statistics based on the provided model space. Other objects for further analysis are also returned.

Usage

```
bma(
  model_space,
  round = 4,
  EMS = NULL,
```

```
  dilution = 0,
  omega = 0.5,
  weighting = c("mb2016", "mb2012", "nt", "uip"),
  eta = NULL
)
```

Arguments

`model_space`	An object of class `badp_model_space`, typically returned by `optim_model_space`.
`round`	Integer indicating the decimal place to which numbers in the BMA tables and prior and posterior model sizes should be rounded (default: 4).
`EMS`	Numeric. Expected model size for binomial and binomial-beta model prior (default: R/2, where R is the number of regressors).
`dilution`	Integer. Use 0 for no dilution prior (default), or 1 to apply a dilution prior (George 2010).
`omega`	Numeric. The exponent of the determinant for the dilution prior (George 2010). Only used when `dilution = 1`. Default: 0.5.
`weighting`	Character string selecting the approximation to the marginal likelihood used to weight the models. One of: `"mb2016"` (default) - the approximation computed by the implementation accompanying Moral-Benito (2016), namely $\exp\{(\ell_j - (k_j/2)\log(NT))/N\}$, where ℓ_j is the maximised log-likelihood of model j, k_j its number of linear parameters, N the number of entities and T the number of periods. This is the Schwarz approximation raised to the power $1/N$ and is the option that reproduces the results published in Moral-Benito (2016); `"mb2012"` - the Schwarz criterion as stated in equations (24)-(30) of Moral-Benito (2012), $\exp\{\ell_j - (k_j/2)\log(NT)\}$, treating the NT entity-period observations as the sample; `"nt"` - as `"mb2016"` but averaging over entity-periods rather than entities, $\exp\{(\ell_j - (k_j/2)\log(NT))/(NT)\}$. This is the scaling that would be internally consistent with the $\log(NT)$ penalty, and it tempers T times more strongly than `"mb2016"`; `"uip"` - the Schwarz criterion with the entity as the unit of information, $\exp\{\ell_j - (k_j/2)\log(N)\}$. The unit information prior underlying the approximation (Kass and Wasserman 1995) is defined through the Fisher information for a single observation, and the likelihood factorises over entities, so the entity is the natural unit.
`eta`	Optional single positive number giving the learning rate directly, as in $f(y\|M_j) \propto \exp(\eta A_j)$ with $A_j = \ell_j - (k_j/2)\log(NT)$. Supplying it overrides `weighting`, which is the intended way to examine how sensitive conclusions are to the rate. `eta = 1` reproduces `"mb2012"` and `eta = 1/N` reproduces `"mb2016"`. Default `NULL`, meaning the rate implied by `weighting`.

Details

Writing $A_j = \ell_j - (k_j/2)\log(NT)$, three of the four options are the same construction with different learning rates η: $\log w_j = \eta A_j$, with $\eta = 1$ for `"mb2012"`, $1/N$ for `"mb2016"` and $1/(NT)$

for "nt". Only "uip" alters the penalty instead of the rate. The eta argument supplies any other rate.

Any $\eta \neq 1$ produces a tempered (power) posterior over models, proportional to $f(y|M_j)^\eta$, which lies outside the approximation that motivates the criterion. What follows from that depends on how the rate is set. A rate held fixed as the sample grows rescales every log weight by the same factor, so differences between models still diverge with N and the posterior continues to concentrate, more slowly when $\eta < 1$ and faster when $\eta > 1$.

A rate that shrinks with the sample behaves differently. Under $\eta = 1/N$, a model at Kullback-Leibler distance c_j per entity has $\eta A_j = A_j/N \to -c_j$, so the weights converge to the fixed positive constants $\exp(-c_j)$ and the posterior never concentrates, however large N becomes. The complexity penalty $(k_j/2)\log(NT)/N$ vanishes in the same limit, so overfitted models are not eventually rejected. Conversely $\eta = 1$ concentrates the posterior sharply, which on likelihoods of this kind can place essentially all mass on a single model.

No option estimates η from the data. The magnitude ("omnibus") adjustment for misspecified and composite likelihoods (Chandler and Bate 2007; Ribatet, Cooley and Davison 2012) would set $\eta = \mathrm{rank}(J)/\mathrm{tr}(H^{-1}J)$, matching the mean of the adjusted log-likelihood ratio to its limiting distribution. It is not offered here because J is substantially rank deficient for this likelihood: parameters entering it only through terms common to every entity contribute nothing, and in the model spaces bundled with the package the entity-level scores span between 8\ directions (see score_rank). The adjustment would then be calibrated on a small part of the parameter space, with no way to assess what the remainder contributes. The ingredients are nonetheless stored with every fitted model space, so the rate can be computed and inspected directly if wanted.

The default is retained for compatibility with the published literature; users are encouraged to check the sensitivity of their conclusions to this choice, which is what eta is for. Switching between the options requires no re-estimation, since all are recovered from the same fitted model space.

Value

An object of class badp_bma, which is a list containing:

uniform_table A table containing the results based on the binomial model prior.

random_table A table containing the results based on the binomial-beta model prior.

reg_names A vector containing the names of the regressors, used by the functions.

R The total number of regressors.

num_of_models The number of models present in the model space.

jointness_data A table containing model IDs and posterior model probabilities (PMPs) for the jointness function.

best_models_data A table containing model IDs, PMPs, coefficients, standard deviations, and standardized regression coefficients (stdRs) for the best_models function.

EMS The expected model size for the binomial and binomial-beta model priors, as specified by the user (default is EMS = R/2).

size_priors A table of uniform and random model priors distributed over model sizes for the model_sizes function.

PMPs A table containing the posterior model probabilities for use in the model_sizes function.

model_priors A table containing the model priors, used by the model_pmp function.

dilution A parameter indicating whether the priors were diluted, used in the model_sizes function.

alphas A matrix of coefficients for the lagged dependent variable across all models, used in the coef_hist function.

betas_nonzero A matrix of nonzero coefficients for the regressors, used in the coef_hist function.

df_free A table containing the degrees of freedom for the estimated models in the best_models function.

PMS_table A table containing the prior and posterior expected model sizes for the binomial and binomial-beta model priors.

omega The dilution parameter used (the exponent of the determinant). Relevant only when `dilution = 1`.

Identification of the robust standard deviations

The columns `PSDR` and `PSDRcon` are derived from the sandwich covariance $H^{-1}JH^{-1}$, where $J = \sum_{i=1}^{N} s_i s_i'$ is the outer product of the entity-level score vectors. The per-entity scores share a component that is identical across entities, and at the maximum they sum to zero, so J is unchanged by centring them: only the variation of the scores across entities contributes. Parameters entering the log-likelihood solely through terms common to every entity have no such variation, and J is consequently rank deficient however many entities are observed. See `score_rank`.

The directions J does span include the lagged dependent variable and the regressors, so the reported `PSDR` and `PSDRcon` are well defined: because J vanishes outside that block, the sandwich restricted to it equals the profile sandwich obtained by profiling the remaining parameters out. What the construction does discard is the score covariance involving those remaining directions, which is set to zero rather than estimated.

The deficiency is the normal state of this likelihood rather than a symptom of a poor fit, so it is not warned about; `summary()` of a model space reports the fraction of parameter directions the scores span. The likelihood, the posterior means (`PM`, `PMcon`) and the posterior inclusion probabilities (`PIP`) do not involve J and are unaffected, as are the Hessian-based standard deviations `PSD` and `PSDcon`, which rest on correct specification rather than on J and are the more defensible choice for this likelihood.

Methods

Objects of class `badp_bma` have the following methods available:

- `print.badp_bma` - Display results
- `summary.badp_bma` - Detailed statistical summary
- `coef.badp_bma` - Extract coefficients
- `plot.badp_bma` - Visualize results

Examples

```
library(magrittr)

data_prepared <- badp::economic_growth[, 1:6] %>%
  badp::feature_standardization(
    excluded_cols = c(country, year, gdp)
```

```
  ) %>%
  badp::feature_standardization(
    group_by_col  = year,
    excluded_cols = country,
    scale         = FALSE
  )

bma_results <- bma(
  model_space = badp::small_model_space,
  round       = 3,
  dilution    = 0
)
```

---

coef.badp_bma *Extract posterior statistics from Bayesian Model Averaging Results*

Description

Coefficient extraction method for objects of class badp_bma.

Usage

```
## S3 method for class 'badp_bma'
coef(
  object,
  prior = "both",
  conditional = FALSE,
  se = FALSE,
  robustSE = FALSE,
  PIP = TRUE,
  ...
)
```

Arguments

object	An object of class badp_bma, typically the result of bma.
prior	Character string specifying which prior to use. Options are "both" (default), "binomial", or "beta". With "both" the result reports estimates under the binomial and the binomial-beta priors and is printed via print.badp_bma_coef.
conditional	Logical. If TRUE, returns posterior means (and posterior standard deviations when se = TRUE) *conditional on inclusion* of the variable in a model - i.e. the columns whose names end in "con" (PMcon, PSDcon, PSDRcon). If FALSE (default), returns the unconditional posterior mean and standard error.
se	Logical. If TRUE, includes a posterior standard deviation column alongside each posterior mean. Defaults to FALSE.

robustSE	Logical. Only meaningful when se = TRUE. If TRUE, uses the robust posterior standard deviation (PSDR / PSDRcon); if FALSE (default), uses the non-robust version (PSD / PSDcon). Ignored with a warning when se = FALSE.
PIP	Logical. If TRUE (default), includes a posterior inclusion probability column for each prior. Set to FALSE to suppress.
...	Additional arguments (currently unused).

Details

This function extracts coefficient estimates from Bayesian Model Averaging results. By default both priors are reported so the user can compare them at a glance; set prior = "binomial" or prior = "beta" to obtain the legacy single-prior return values (useful when feeding coefficients into downstream code).

Value

The shape of the return value depends on prior, conditional, se, robustSE and PIP:

- prior = "both": always a badp_bma_coef data frame. Columns are grouped by prior; for each prior the columns binom_PM / beta_PM (or binom_PMcon / beta_PMcon when conditional = TRUE), standard error columns binom_PSD / beta_PSD (or binom_PSDR / beta_PSDR when robustSE = TRUE, or their con variants when conditional = TRUE) when se = TRUE, and binom_PIP / beta_PIP when PIP = TRUE.
- prior = "binomial" or "beta": a named numeric vector of estimates when se = FALSE and PIP = FALSE; otherwise a data frame with columns PM (or PMcon when conditional = TRUE), posterior standard deviation column PSD / PSDR (or PSDcon / PSDRcon) when se = TRUE, and PIP when PIP = TRUE.

See Also

bma, summary.badp_bma, print.badp_bma_coef

Examples

```
data(full_model_space)
results <- bma(full_model_space)

# Posterior means under both priors with PIP
coef(results)

# With standard errors
coef(results, se = TRUE)

# With robust standard errors
coef(results, se = TRUE, robustSE = TRUE)

# Conditional posterior means and SEs
coef(results, conditional = TRUE, se = TRUE)

# Suppress PIP column
```

```
coef(results, PIP = FALSE)

# Single-prior numeric vector (legacy behaviour)
coef(results, prior = "binomial", PIP = FALSE)
```

`coef_hist` *Graphs of the distribution of the coefficients over the model space*

Description

This function draws graphs of the distribution (in the form of histogram or kernel density) of the coefficients for all the considered regressors over the part of the model space that includes these regressors (half of the model space).

Usage

```
coef_hist(
  bma_list,
  weight = NULL,
  bin_method = c("FD", "SC", "vec"),
  bin_widths = NULL,
  use_bin_count = 0,
  bin_counts = NULL,
  use_kernel = 0
)
```

Arguments

`bma_list`	An object of class `badp_bma`, typically returned by `bma`.
`weight`	Parameter indicating whether the coefficients should be weighted by posterior model probabilities: 1. NULL - no weighting (default option) 2. "binomial" - using posterior model probabilities based on binomial model prior 3. "beta" - using posterior model probabilities based on binomial-beta model prior
`bin_method`	Character string specifying the method for bin widths (default: `"FD"`). One of: `"FD"` - Freedman-Diaconis method; `"SC"` - Scott method; `"vec"` - user specified bin widths provided through a vector (parameter: `bin_widths`).

`bin_widths` A vector with bin widths to be used to construct histograms for the regressors. The vector must be of the size equal to total number of regressors plus one for the lagged dependent variable. The vector with bin widths is used only if parameter `bin_method = "vec"`.

`use_bin_count` Parameter taking the values (default: `use_bin_count = 0`):
1 - the histogram will be built based on the number of bins specified by the user through parameter `bin_counts`. If `use_bin_count = 1`, the function ignores parameter `bin_method`.
0 - the histogram will be built in line with parameter `bin_method`.

`bin_counts` A vector with the numbers of bins to be used to construct histograms for the regressors. The vector must be of the size equal to total number of regressors plus one for the lagged dependent variable. The vector with bin counts is used only if parameter `use_bin_count = 1`.

`use_kernel` A parameter taking the values (default: `use_kernel = 0`):
1 - the function will build graphs using kernel density for the distribution of coefficients (with `use_kernel = 1`, the function ignores parameters `bin_method` and `use_bin_count`)
0 - the function will build regular histogram density for the distribution of coefficients

Value

A list with the graphs of the distribution of coefficients for all the considered regressors and the lagged dependent variable.

Examples

```
library(magrittr)

data_prepared <- badp::economic_growth[, 1:6] %>%
  badp::feature_standardization(
    excluded_cols = c(country, year, gdp)
  ) %>%
  badp::feature_standardization(
    group_by_col  = year,
    excluded_cols = country,
    scale         = FALSE
  )

bma_results <- bma(
  model_space = badp::small_model_space,
  round       = 3,
  dilution    = 0
)

coef_plots <- coef_hist(bma_results, use_kernel = 1)
```

`compute_model_space_stats` *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

```
compute_model_space_stats(
  df,
  dep_var_col,
  timestamp_col,
  entity_col,
  params,
  nested = TRUE,
  exact_value = FALSE,
  model_prior = "uniform",
  cl = NULL
)
```

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)
`params`	A matrix (with named rows) with each column corresponding to a model. Each column specifies model parameters. Compare with optim_model_space_params
`nested`	Logical. If `TRUE` (default), compute approximate standard deviations using the nested-model approach via `nested_std_dev_from_params()`. If `FALSE`, use the non-nested approach via `non_nested_std_dev_from_params()`. The choice affects which approximation routine is used for each model in `params`.
`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'`.
`cl`	An optional cluster object. If supplied, the function will use this cluster for parallel processing. If `NULL` (the default), `pbapply::pblapply` will run sequentially.

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 the marginal likelihood approximation used to weight the models, namely `exp((loglik - (k/2) * log(N * T)) / N)`; note that it is not a BIC. Then there are rows with standard deviations for each parameter, followed by rows with robust standard deviations. The last three rows hold `tr(H^-1 J)`, where `H` is the observed information and `J` the outer product of the entity-level scores, the dimension of the parameter vector, and the numerical rank of `J`. These are the ingredients of the magnitude adjustment for misspecified likelihoods, whose rate is `rank(J) / tr(H^-1 J)`; see score_rank for why `J` is typically rank deficient, and bma for why no weighting option is built on it.

Examples

```
library(magrittr)
data_prepared <- badp::economic_growth[, 1:6] %>%
  badp::feature_standardization(
    excluded_cols = c(country, year, gdp)
  ) %>%
  badp::feature_standardization(
    group_by_col  = year,
    excluded_cols = country,
    scale         = FALSE
  )

compute_model_space_stats(
  df            = data_prepared,
  dep_var_col   = gdp,
  timestamp_col = year,
  entity_col    = country,
  params        = small_model_space$params
)
```

`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 (73 countries and 4 periods + extra one for lagged dependent variable):

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/

`feature_standardization` *Perform feature standardization*

Description

This function performs feature standardization (also known as z-score normalization) by centering the features around their mean and scaling by their standard deviation.

Usage

```
feature_standardization(df, excluded_cols, group_by_col, scale = TRUE)
```

Arguments

`df`	Data frame with the data.
`excluded_cols`	Unquoted column names to exclude from standardization. If missing, all columns are standardized.
`group_by_col`	Unquoted column names to group the data by before applying standardization. If missing, no grouping is performed.
`scale`	Logical. If `TRUE` (default) scales by the standard deviation.

Value

A data frame 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)
)

# Standardize every column
df_with_only_numeric_values <- df[, setdiff(names(df), "country")]
feature_standardization(df_with_only_numeric_values)

# Standardize all columns except 'country'
feature_standardization(df, excluded_cols = country)

# Standardize across countries (grouped by 'country')
feature_standardization(df, group_by_col = country)

# Standardize, excluding 'country' and group-wise by 'year'
feature_standardization(df, excluded_cols = country, group_by_col = year)
```

---

full_bma_results *Example output of the bma function*

Description

A badp_bma object summarising the BMA analysis

Usage

```
full_bma_results
```

Format

An object of class badp_bma

full_model_space *Example output of* optim_model_space

Description

A badp_model_space object created with optim_model_space using the economic_growth dataset.

Usage

```
full_model_space
```

Format

An object of class badp_model_space:

params A numeric matrix with 106 rows and 512 columns (corresponding to $2^9 = 512$ models), containing parameter values for the full model space. Each column represents a different model. Entries are NA for parameters absent from a given model.

stats A numeric matrix of statistics computed by compute_model_space_stats based on params, with one column per model. Row 1 holds the maximized log-likelihood. Row 2 holds the marginal likelihood approximation used to weight the models, exp((loglik - (k/2) * log(N * T)) / N); note that it is not a BIC. There follow K rows of standard deviations and K rows of robust standard deviations, where K is the number of regressors including the lagged dependent variable. The final three rows hold tr(H^-1 J), the dimension of the parameter vector and the numerical rank of J; see score_rank.

reg_names A character vector with the names of the variables.

observations_num The total number of observations in the panel (292).

df The data frame used in the analysis.

is_nested A logical indicating whether the model space uses nested specifications.

jointness *Calculation of the jointness measures*

Description

This function calculates four types of the jointness measures based on the posterior model probabilities calculated using binomial and binomial-beta model prior. The four measures are:

1. HCGHM - for Hofmarcher et al. (2018) measure;

2. LS - for Ley & Steel (2007) measure;

3. DW - for Doppelhofer & Weeks (2009) measure;

4. PPI - for posterior probability of including both variables.
   The measures under binomial model prior will appear in a table above the diagonal, and the measure calculated under binomial-beta model prior below the diagonal.

   REFERENCES
   Doppelhofer G, Weeks M (2009) Jointness of growth determinants. Journal of Applied Econometrics., 24(2), 209-244. doi: 10.1002/jae.1046
   Hofmarcher P, Crespo Cuaresma J, Grün B, Humer S, Moser M (2018) Bivariate jointness measures in Bayesian Model Averaging: Solving the conundrum. Journal of Macroeconomics, 57, 150-165. doi: 10.1016/j.jmacro.2018.05.005
   Ley E, Steel M (2007) Jointness in Bayesian variable selection with applications to growth regression. Journal of Macroeconomics, 29(3), 476-493. doi: 10.1016/j.jmacro.2006.12.002

Usage

```
jointness(
  bma_list,
  measure = c("HCGHM", "LS", "DW", "PPI"),
  rho = 0.5,
  round = 3
)
```

Arguments

bma_list	An object of class badp_bma, typically returned by bma.
measure	Character string specifying the measure of jointness. One of: "HCGHM" - Hofmarcher et al. (2018) measure (default); "LS" - Ley & Steel (2007) measure; "DW" - Doppelhofer & Weeks (2009) measure; "PPI" - posterior probability of including both variables.
rho	The parameter "rho" (ρ) to be used in HCGHM jointness measure (default rho = 0.5). Works only if HCGHM measure is chosen (Hofmarcher et al. 2018).
round	Parameter indicating the decimal place to which the jointness measures should be rounded (default round = 3).

Value

A table with jointness measures for all the pairs of regressors used in the analysis. The results obtained with the binomial model prior are above the diagonal, while the ones obtained with the binomial-beta prior are below.

Examples

```
library(magrittr)

data_prepared <- badp::economic_growth[, 1:6] %>%
```

```
  badp::feature_standardization(
    excluded_cols = c(country, year, gdp)
  ) %>%
  badp::feature_standardization(
    group_by_col  = year,
    excluded_cols = country,
    scale         = FALSE
  )

bma_results <- bma(
  model_space = badp::small_model_space,
  round       = 3,
  dilution    = 0
)

jointness_table <- jointness(bma_results, measure = "HCGHM", rho = 0.5, round = 3)
```

`migration_data` *Migration data in the original format*

Description

Data used in the manuscript Afonso, A., Alves, J., & Beck, K. (2025). Drivers of migration flows in the European Union: Earnings or unemployment? International Labour Review, 164(2), 1-23. doi:10.16995/ilr.18845

Usage

```
migration_data
```

Format

`migration_data`**:**

A data frame with 1012 rows and 8 columns (253 country pairs and 4 periods + one additional observation for the lagged dependent variable):

Time Year

Pair Country pair ID

Mig Net migration between two countries

Mig_lag Lagged net migration between two countries

Earn Difference in average real after-tax earnings in PPP

Unemp Difference in the unemployment rate

Social Difference in average social benefits in PPP

Tax Difference in average tax rate

Source

doi:10.7910/DVN/GTOFJB

migration_model_space *Example output of* optim_model_space *in the case of migration data*

Description

A badp_model_space object created with optim_model_space using the migration_data dataset.

Usage

```
migration_model_space
```

Format

An object of class badp_model_space:

params A numeric matrix with 51 rows and 16 columns, containing parameter values for the full model space. Each column represents a different model.

stats A numeric matrix of statistics computed by compute_model_space_stats based on params, with one column per model. Row 1 holds the maximized log-likelihood. Row 2 holds the marginal likelihood approximation used to weight the models, exp((loglik - (k/2) * log(N * T)) / N); note that it is not a BIC. There follow K rows of standard deviations and K rows of robust standard deviations, where K is the number of regressors including the lagged dependent variable. The final three rows hold tr(H^-1 J), the dimension of the parameter vector and the numerical rank of J; see score_rank.

reg_names A character vector with the names of the variables.

observations_num The total number of observations in the panel (1012).

df The data frame used in the analysis.

is_nested A logical indicating whether the model space uses nested specifications.

migration_model_space_nonnested *Example output of* optim_model_space *in the case of migration data obtained with nonnested approach.*

Description

A badp_model_space object created with optim_model_space using the migration_data dataset with nonnested approach.

Usage

```
migration_model_space_nonnested
```

Format

An object of class `badp_model_space`:

params A numeric matrix with 51 rows and 16 columns, containing parameter values for the full model space. Each column represents a different model.

stats A numeric matrix of statistics computed by `compute_model_space_stats` based on `params`, with one column per model. Row 1 holds the maximized log-likelihood. Row 2 holds the marginal likelihood approximation used to weight the models, `exp((loglik - (k/2) * log(N * T)) / N)`; note that it is not a BIC. There follow `K` rows of standard deviations and `K` rows of robust standard deviations, where `K` is the number of regressors including the lagged dependent variable. The final three rows hold `tr(H^-1 J)`, the dimension of the parameter vector and the numerical rank of `J`; see `score_rank`.

reg_names A character vector with the names of the variables.

observations_num The total number of observations in the panel (1012).

df The data frame used in the analysis.

is_nested A logical indicating whether the model space uses nested specifications.

`model_pmp` *Graphs of the prior and posterior model probabilities for the best individual models*

Description

This function draws four graphs of prior and posterior model probabilities for the best individual models:
a) The results with binomial model prior (based on PMP - posterior model probability)
b) The results with binomial-beta model prior (based on PMP - posterior model probability)
Models on the graph are ordered according to their posterior model probability.

Usage

```
model_pmp(bma_list, top = NULL)
```

Arguments

`bma_list`	bma_list object (the result of the bma function)
`top`	The number of the best model to be placed on the graphs

Value

A list with three graphs with prior and posterior model probabilities for individual models:

1. The results with binomial model prior (based on PMP - posterior model probability)
2. The results with binomial-beta model prior (based on PMP - posterior model probability)
3. One graph combining the aforementioned graphs

Examples

```
library(magrittr)

data_prepared <- badp::economic_growth[, 1:6] %>%
  badp::feature_standardization(
    excluded_cols = c(country, year, gdp)
  ) %>%
  badp::feature_standardization(
    group_by_col  = year,
    excluded_cols = country,
    scale         = FALSE
  )

bma_results <- bma(
  model_space = badp::small_model_space,
  round       = 3,
  dilution    = 0
)

model_graphs <- model_pmp(bma_results, top = 16)
```

`model_sizes` *Graphs of the prior and posterior model probabilities of the model sizes*

Description

This function draws two graphs of prior and posterior model probabilities:
a) The results with binomial model prior
b) The results with binomial-beta model prior
c) One graph combining all the aforementioned graphs

Usage

```
model_sizes(bma_list)
```

Arguments

`bma_list`	bma_list object (the result of the bma function)

Value

A list with three graphs with prior and posterior model probabilities for model sizes:

1. The results with binomial model prior

2. The results with binomial-beta model prior

3. One graph combining all the aforementioned graphs

Examples

```
library(magrittr)

data_prepared <- badp::economic_growth[, 1:6] %>%
  badp::feature_standardization(
    excluded_cols = c(country, year, gdp)
  ) %>%
  badp::feature_standardization(
    group_by_col  = year,
    excluded_cols = country,
    scale         = FALSE
  )

bma_results <- bma(
  model_space = badp::small_model_space,
  round       = 3,
  dilution    = 0
)

size_graphs <- model_sizes(bma_results)
```

---

model_space_nonnested *Example output of* optim_model_space *for non-nested models*

Description

A badp_model_space object created with optim_model_space using the economic_growth dataset and nested = FALSE. Compare full_model_space, which is the same data under the nested approach.

Usage

```
model_space_nonnested
```

Format

An object of class badp_model_space:

params A numeric matrix of parameter values for the model space, one column per model. Entries are NA for parameters absent from a given model.

stats A numeric matrix of statistics computed by `compute_model_space_stats` based on `params`, with one column per model. Row 1 holds the maximized log-likelihood. Row 2 holds the marginal likelihood approximation used to weight the models, `exp((loglik - (k/2) * log(N * T)) / N)`; note that it is not a BIC. There follow `K` rows of standard deviations and `K` rows of robust standard deviations, where `K` is the number of regressors including the lagged dependent variable. The final three rows hold `tr(H^-1 J)`, the dimension of the parameter vector and the numerical rank of `J`; see `score_rank`.

reg_names A character vector with the names of the variables.

observations_num The total number of observations in the panel (292).

df The data frame used in the analysis.

is_nested A logical indicating whether the model space uses nested specifications.

convergence A matrix of per-model convergence diagnostics; see `optim_model_space`.

`optim_model_space` *Calculation of the model_space object*

Description

This function calculates model space, values of the maximized likelihood function, BICs, and standard deviations of the parameters that will be used in Bayesian model averaging. Moreover, it provides a vector with the names of the variables for bma function and the number of observations.

Usage

```
optim_model_space(
  df,
  timestamp_col,
  entity_col,
  dep_var_col,
  init_value,
  nested = TRUE,
  exact_value = FALSE,
  cl = NULL,
 control = list(trace = 0, maxit = 10000, fnscale = -1, REPORT = 100, scale = 0.05),
  max_restarts = 5,
  restart_tol = 0.001,
  max_reoptimizations = 5,
  max_init_attempts = 100
)
```

Arguments

`df`	Data frame with data for the 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
: Function of one argument n returning n starting values for the numerical optimization (e.g. function(n) runif(n, 0.1, 1)). It is called separately for every model, so every model gets its own starting point; with a random generator this turns the estimation into a randomized multi-start experiment. A single number is also accepted and is used as the starting value for every parameter, so that init_value = 0.5 is equivalent to function(n) rep(0.5, n). Starting values must be non-zero, because zeros encode excluded parameters.

nested
: Logical. If TRUE (default), compute approximate standard deviations using the nested-model approach via nested_std_dev_from_params(). If FALSE, use the non-nested approach via non_nested_std_dev_from_params(). The choice affects which approximation routine is used for each model in params.

exact_value
: Whether the exact value of the likelihood should be computed (TRUE) or just the proportional part (FALSE). Check sem_likelihood for details.

cl
: An optional cluster object. If supplied, the function will use this cluster for parallel processing. If NULL (the default), pbapply::pblapply will run sequentially.

control
: a list of control parameters for the optimization which are passed to optim. Default is list(trace = 0, 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.

max_restarts
: Maximum number of times the BFGS optimization is restarted from its previous solution for a single model. A restart resets the internal curvature approximation of BFGS, which often makes further progress on ill-conditioned likelihood ridges where a single run stalls. Default is 5.

restart_tol
: Log-likelihood improvement between restarts below which the optimization is considered converged. Improvements of this size are immaterial for posterior model probabilities. Default is 1e-3.

max_reoptimizations
: Maximum number of times a model is re-optimized from a fresh starting point drawn from init_value, when the solution reached turns out to be one no standard errors can be computed from. BFGS stops wherever the gradient vanishes, which need not be a maximum, and from a randomly drawn starting point a model occasionally ends up in a degenerate region instead - hundreds of log-likelihood units below its maximum, with an observed information matrix so rank-deficient that inverting it into standard errors fails with 'system is computationally singular'. Such a solution is therefore discarded and the whole optimization repeated from a new starting point. A model whose every attempt ends this way keeps the likeliest of them and is reported with converged = 0; no error is raised, so a single model cannot bring down the estimation of the whole model space. Default is 5.

max_init_attempts
: Maximum number of starting points drawn from init_value for a single model. Not every parameter vector is a usable starting point: at some of them the covariance matrix implied by the parameters is not positive definite, so the likelihood is undefined there and the optimization cannot even start (taping the likelihood

for automatic differentiation fails on the Cholesky factorization with an error such as 'the leading minor of order 13 is not positive'). Whether a starting point is usable depends on the model, so it can only be checked after the point has been drawn: unusable draws are discarded and replaced by fresh draws from `init_value`, and only after `max_init_attempts` unsuccessful draws is an error raised. The number of draws each model actually needed is reported in the `n_init_draws` row of the `convergence` element of the result. Default is `100`.

Value

An object of class `badp_model_space`, which is a list with the following elements:

1. params - table with parameters of all estimated models
2. stats - table with the value of maximized likelihood function, BIC, and standard errors for all estimated models
3. reg_names - vector with the names of the variables
4. observations_num - number of observations
5. df - data frame used in estimation
6. is_nested - logical indicating whether nested approach was used
7. convergence - matrix of per-model convergence diagnostics with rows `converged` (1 if the likelihood value stalled across restarts at a solution standard errors can be computed from, 0 if the restart budget was exhausted while the value was still improving or every re-optimization ended in a degenerate region), `optim_code` (the optim convergence code of the final run), `n_restarts`, `max_abs_gradient` and `n_init_draws` (the number of starting points drawn from `init_value` before one at which the likelihood is defined was found). A large final gradient with `converged = 1` indicates parameters on a degenerate (nearly collinear) likelihood ridge; the likelihood value is trustworthy but the standard errors of the affected coordinates are not.

Rank of the sandwich covariance

The robust standard errors stored in the statistics, and the columns `PSDR` and `PSDRcon` that bma derives from them, come from the sandwich $H^{-1}JH^{-1}$ with $J = \sum_{i=1}^{N} s_i s_i'$ formed from the entity-level score vectors. J depends only on the variation of those scores across entities, so parameters entering the log-likelihood solely through terms common to every entity contribute nothing and J is rank deficient regardless of N; see `score_rank`. The fraction of parameter directions the scores span is reported by `summary.badp_model_space`. The standard errors that are reported remain well defined, as the sandwich restricted to the spanned block is the profile sandwich on that block, but the score covariance involving the remaining directions is discarded. The likelihood, the parameter estimates and the Hessian-based standard errors are unaffected.

Methods

Objects of class badp_model_space have the following methods available:

- print.badp_model_space - Display model space information

Examples

```
library(magrittr)

data_prepared <- badp::economic_growth[, 1:5] %>%
  badp::feature_standardization(
    excluded_cols = c(country, year, gdp)
  ) %>%
  badp::feature_standardization(
    group_by_col  = year,
    excluded_cols = country,
    scale         = FALSE
  )

optim_model_space(
  df            = data_prepared,
  dep_var_col   = gdp,
  timestamp_col = year,
  entity_col    = country,
  init_value    = function(n) runif(n, 0.1, 1)
)
```

---

optim_model_space_params

Finds MLE parameters for each model in the given model space

Description

Given a dataset and a generator of starting values, initializes a model space by drawing a starting point for each model. Then for each model performs a numerical optimization and finds parameters which maximize the likelihood.

Usage

```
optim_model_space_params(
  df,
  timestamp_col,
  entity_col,
  dep_var_col,
  init_value,
  nested,
  exact_value = FALSE,
```

```
  cl = NULL,
 control = list(trace = 0, maxit = 10000, fnscale = -1, REPORT = 100, scale = 0.05),
  max_restarts = 5,
  restart_tol = 0.001,
  max_reoptimizations = 5,
  max_init_attempts = 100
)
```

Arguments

df	Data frame with data for the 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	Function of one argument n returning n starting values for the numerical optimization (e.g. function(n) runif(n, 0.1, 1)). It is called separately for every model, so every model gets its own starting point; with a random generator this turns the estimation into a randomized multi-start experiment. A single number is also accepted and is used as the starting value for every parameter, so that init_value = 0.5 is equivalent to function(n) rep(0.5, n). Starting values must be non-zero, because zeros encode excluded parameters.
nested	Logical. If TRUE (default), compute approximate standard deviations using the nested-model approach via nested_std_dev_from_params(). If FALSE, use the non-nested approach via non_nested_std_dev_from_params(). The choice affects which approximation routine is used for each model in params.
exact_value	Whether the exact value of the likelihood should be computed (TRUE) or just the proportional part (FALSE). Check sem_likelihood for details.
cl	An optional cluster object. If supplied, the function will use this cluster for parallel processing. If NULL (the default), pbapply::pblapply will run sequentially.
control	a list of control parameters for the optimization which are passed to optim. Default is list(trace = 0, maxit = 10000, fnscale = -1, REPORT = 100, scale = 0.05).
max_restarts	Maximum number of times the BFGS optimization is restarted from its previous solution for a single model. A restart resets the internal curvature approximation of BFGS, which often makes further progress on ill-conditioned likelihood ridges where a single run stalls. Default is 5.
restart_tol	Log-likelihood improvement between restarts below which the optimization is considered converged. Improvements of this size are immaterial for posterior model probabilities. Default is 1e-3.
max_reoptimizations	Maximum number of times a model is re-optimized from a fresh starting point drawn from init_value, when the solution reached turns out to be one no standard errors can be computed from. BFGS stops wherever the gradient vanishes, which need not be a maximum, and from a randomly drawn starting point a

model occasionally ends up in a degenerate region instead - hundreds of log-likelihood units below its maximum, with an observed information matrix so rank-deficient that inverting it into standard errors fails with 'system is computationally singular'. Such a solution is therefore discarded and the whole optimization repeated from a new starting point. A model whose every attempt ends this way keeps the likeliest of them and is reported with `converged = 0`; no error is raised, so a single model cannot bring down the estimation of the whole model space. Default is `5`.

`max_init_attempts`
: Maximum number of starting points drawn from `init_value` for a single model. Not every parameter vector is a usable starting point: at some of them the covariance matrix implied by the parameters is not positive definite, so the likelihood is undefined there and the optimization cannot even start (taping the likelihood for automatic differentiation fails on the Cholesky factorization with an error such as 'the leading minor of order 13 is not positive'). Whether a starting point is usable depends on the model, so it can only be checked after the point has been drawn: unusable draws are discarded and replaced by fresh draws from `init_value`, and only after `max_init_attempts` unsuccessful draws is an error raised. The number of draws each model actually needed is reported in the `n_init_draws` row of the convergence diagnostics. Default is `100`.

Value

List (or matrix) of parameters describing analyzed models. The returned matrix carries a `"convergence"` attribute: a matrix with one column per model and rows `converged` (1 if the likelihood value stalled at a solution standard errors can be computed from, 0 if the restart budget was exhausted while still improving or every re-optimization ended in a degenerate region), `optim_code` (the optim convergence code of the final run), `n_restarts`, `max_abs_gradient` and `n_init_draws` (the number of starting points drawn from `init_value` before one at which the likelihood is defined was found). A large final gradient with `converged = 1` indicates parameters on a degenerate (nearly collinear) likelihood ridge rather than a failed optimization.

`original_economic_growth`
Economic Growth Data in the original format

Description

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

Usage

```
original_economic_growth
```

Format

`original_economic_growth`:
A data frame with 292 rows and 13 columns (73 countries and 4 periods + extra one for lagged dependent variable):

year Year
country Country ID
gdp Logarithm of GDP per capita (2000 US dollars at PP)
gdp_lag Lagged 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/

---

plot.badp_bma *Plot Bayesian Model Averaging Results*

Description

Plot method for objects of class badp_bma.

Usage

```
## S3 method for class 'badp_bma'
plot(x, which = "model_sizes", ...)
```

Arguments

x	An object of class badp_bma, typically the result of bma.
which	Character string specifying which plot to create. Options are: • "model_sizes" - Model size distributions (default) • "best_models" - Best models • "jointness" - Jointness analysis • "coef_hist" - Coefficient histograms • "posterior_dens" - Posterior densities • "model_pmp" - Model posterior probabilities
...	Additional arguments passed to the underlying plot function.

Details

This function dispatches to the appropriate visualization function based on the `which` parameter. The default plot shows model size distributions, which provides a comprehensive overview of the prior and posterior distributions over model sizes.

Value

The object returned by the selected visualization helper. Depending on `which`, this may be a single plot object or a list containing plots and/or tables; some helpers may also print output as a side effect.

See Also

`bma`, `model_sizes`, `best_models`, `jointness`, `coef_hist`, `posterior_dens`, `model_pmp`

Examples

```
data(full_model_space)
results <- bma(full_model_space)

# Default plot (model sizes)
plot(results)

# Other plot types
plot(results, which = "best_models")
plot(results, which = "jointness")
```

`posterior_dens` *Graphs of the posterior densities of the coefficients*

Description

This function draws graphs of the posterior densities of all the coefficients of interest.

Usage

```
posterior_dens(bma_list, prior = "binomial", SE = "standard")
```

Arguments

`bma_list`	bma object (the result of the bma function)
`prior`	Parameter indicating which model prior should be used for calculations: 1. "binomial" - using binomial model prior (default option) 2. "beta" - using binomial-beta model prior

SE — Parameter indicating which standard errors should be used in calculation of posterior standard deviation:

1. "standard" - regular standard errors (default option)
2. "robust" - robust standard errors

Value

A list with the graphs of the posterior densities of coefficients for all the considered regressors.

Examples

```
library(magrittr)

data_prepared <- badp::economic_growth[, 1:6] %>%
  badp::feature_standardization(
    excluded_cols = c(country, year, gdp)
  ) %>%
  badp::feature_standardization(
    group_by_col  = year,
    excluded_cols = country,
    scale         = FALSE
  )

bma_results <- bma(
  model_space = badp::small_model_space,
  round       = 3,
  dilution    = 0
)

posterior_graphs <- posterior_dens(bma_results, prior = "binomial", SE = "robust")
```

---

print.badp_best_models

Print Best Models Tables

Description

Print method for objects of class badp_best_models returned by best_models. Draws the table chosen by the estimate and robust arguments of the original best_models call to the active graphics device.

Usage

```
## S3 method for class 'badp_best_models'
print(x, ...)
```

Arguments

x	An object of class badp_best_models.
...	Additional arguments (currently unused).

Details

Because R only auto-prints expressions evaluated at the top level, calling best_models(bma_list) without assignment triggers this method (and hence the table is drawn), while best <- best_models(bma_list) stays silent.

Value

Invisibly returns the input object x.

See Also

best_models

---

print.badp_bma *Print Bayesian Model Averaging Results*

Description

Print method for objects of class badp_bma.

Usage

```
## S3 method for class 'badp_bma'
print(x, ...)
```

Arguments

x	An object of class badp_bma, typically the result of bma.
...	Additional arguments (currently unused).

Details

This method is a thin wrapper that delegates to summary.badp_bma and then prints the resulting summary, so print(x) and print(summary(x)) produce identical output. The output displays BMA statistics for both the binomial and binomial-beta priors to allow direct comparison.

Value

Invisibly returns the input object x.

See Also

bma, summary.badp_bma, coef.badp_bma

Examples

```
data(full_model_space)
results <- bma(full_model_space)
print(results)
```

`print.badp_bma_coef` *Print Coefficient Tables from Bayesian Model Averaging*

Description

Print method for objects of class `badp_bma_coef` produced by `coef.badp_bma` when `prior = "both"`.

Usage

```
## S3 method for class 'badp_bma_coef'
print(x, digits = 4, ...)
```

Arguments

`x`	An object of class `badp_bma_coef`.
`digits`	Integer. Number of significant digits used when printing numeric columns. Defaults to 4.
`...`	Additional arguments passed to `print.data.frame`.

Details

When the result contains only point estimates (`se = FALSE`, `PIP = FALSE`), the two priors are printed side by side. Otherwise the output is split into two stacked panels - one per prior - each with the requested combination of `estimate`, `std.error` and `PIP` columns. The header reflects whether the estimates are unconditional or conditional on inclusion, and whether standard errors are robust.

Value

Invisibly returns the input object `x`.

See Also

`coef.badp_bma`

print.badp_drawable_grob

Print a Drawable Best-Models Grob

Description

Print method for the captured gTree objects stored in positions 7-9 of the list returned by best_models. Renders the grob to the active graphics device via grid.draw so that typing e.g. best[[9]] at the console displays the picture instead of a description string.

Usage

```
## S3 method for class 'badp_drawable_grob'
print(x, newpage = TRUE, ...)
```

Arguments

x	A badp_drawable_grob object (a gTree captured by grid.grabExpr).
newpage	Logical; if TRUE (default) a new graphics page is started before drawing.
...	Additional arguments (currently unused).

Value

Invisibly returns the input object x.

See Also

best_models, grid.draw

print.badp_model_space

Print Model Space Object

Description

Print method for objects of class badp_model_space.

Usage

```
## S3 method for class 'badp_model_space'
print(x, ...)
```

Arguments

x	An object of class badp_model_space, typically the result of optim_model_space.
...	Additional arguments forwarded to summary.badp_model_space.

Details

This method is a thin wrapper that delegates to summary.badp_model_space and then prints the resulting summary, so print(x) and print(summary(x)) produce identical output.

Value

Invisibly returns the input object x.

See Also

summary.badp_model_space, optim_model_space, bma

Examples

```
data(full_model_space)
print(full_model_space)
```

---

print.summary.badp_bma

Print Summary of Bayesian Model Averaging Results

Description

Print method for summary.badp_bma objects.

Usage

```
## S3 method for class 'summary.badp_bma'
print(x, ...)
```

Arguments

x	An object of class summary.badp_bma.
...	Additional arguments (currently unused).

Value

Invisibly returns the input object x.

See Also

summary.badp_bma

print.summary.badp_model_space *Print Summary of a Model Space Object*

Description

Print method for `summary.badp_model_space` objects.

Usage

```
## S3 method for class 'summary.badp_model_space'
print(x, ...)
```

Arguments

`x`	An object of class `summary.badp_model_space`.
`...`	Additional arguments (currently unused).

Value

Invisibly returns the input object `x`.

See Also

`summary.badp_model_space`

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. n_regressors * (n_periods - 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. `n_regressors * (n_periods - 1) * n_periods / 2`

Value

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

Examples

```
data(economic_growth)
eg <- feature_standardization(economic_growth, excluded_cols = c(year, country))
sem_likelihood(0.5, eg, year, country, gdp)
```

---

small_model_space *Example output of* optim_model_space *(small version)*

Description

A list created with `optim_model_space` using the `economic_growth` dataset and only three regressors: `ish`, `sed`, and `pgrw`.

Usage

```
small_model_space
```

Format

An object of class `badp_model_space`:

params A numeric matrix with 40 rows and 8 columns (corresponding to $2^3 = 8$ models), containing parameter values for the model space. Each column represents a different model.

stats A numeric matrix of statistics computed by `compute_model_space_stats` based on `params`, with one column per model. Row 1 holds the maximized log-likelihood. Row 2 holds the marginal likelihood approximation used to weight the models, `exp((loglik - (k/2) * log(N * T)) / N)`; note that it is not a BIC. There follow `K` rows of standard deviations and `K` rows of robust standard deviations, where `K` is the number of regressors including the lagged dependent variable. The final three rows hold `tr(H^-1 J)`, the dimension of the parameter vector and the numerical rank of `J`; see `score_rank`.

reg_names A character vector with the names of the regressors.

observations_num The total number of observations in the panel (292).

df The data frame used in the analysis.

is_nested A logical indicating whether the model space uses nested specifications.

summary.badp_bma *Summarize Bayesian Model Averaging Results*

Description

Summary method for objects of class badp_bma.

Usage

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

Arguments

object	An object of class badp_bma, typically the result of bma.
...	Additional arguments (currently unused).

Details

This function creates a comprehensive summary object that includes model space information, BMA statistics for both priors, and highlights variables with high posterior inclusion probabilities. The summary always displays results for both the binomial and binomial-beta priors to allow direct comparison.

Value

An object of class summary.badp_bma containing:

- model_space_size - Total number of models in the model space
- num_regressors - Number of regressors (excluding lagged dependent variable)
- expected_model_size - Expected model size
- dilution_applied - Logical indicating if dilution prior was used
- omega - Numeric value of the dilution parameter (omega); only relevant when dilution_applied is TRUE
- results_binomial - Coefficient table for binomial prior
- results_beta - Coefficient table for binomial-beta prior
- model_sizes - Prior and posterior model sizes table
- reg_names - Variable names

See Also

bma, print.badp_bma, coef.badp_bma

Examples

```
data(full_model_space)
results <- bma(full_model_space)
summary(results)
```

`summary.badp_model_space` *Summarize a Model Space Object*

Description

Summary method for objects of class `badp_model_space`. Replaces the default `summary()` output (which just lists the structure of the underlying list) with a structured object describing the dimensions of the model space, its variables, and a brief look at the per-model log-likelihoods stored in `object$stats`.

Usage

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

Arguments

`object`	An object of class `badp_model_space`, typically the result of `optim_model_space`.
`...`	Additional arguments (currently unused).

Value

An object of class `summary.badp_model_space` containing:

- `num_models` - Number of models in the model space (2^R).
- `num_regressors` - Number of regressors excluding the lagged dependent variable (R).
- `num_params` - Number of parameters in the full parameter vector (rows of `object$params`).
- `observations_num` - Number of observations used.
- `is_nested` - Logical, whether the model space is nested.
- `reg_names` - All variable names (lagged dependent first).
- `dep_var_name` - The (lagged) dependent variable name.
- `regressor_names` - The regressor names.
- `data_dim` - Dimensions of the source data frame, or `NULL` if not stored.
- `likelihoods` - Per-model log-likelihood values (row 1 of `object$stats`), or `NULL` if not available.
- `num_nonconverged` - Number of models whose optimization did not converge, or `NULL` if the model space carries no convergence diagnostics (e.g. objects created before badp 0.6.0).

See Also

print.badp_model_space, print.summary.badp_model_space, optim_model_space, bma

Examples

```
data(full_model_space)
summary(full_model_space)
```

Index