

Package ‘badp’

September 22, 2026

Title Bayesian Averaging for Dynamic Panels

Version 0.8.0

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](https://doi.org/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 knitr, pkgdown, rmarkdown, spelling, testthat (>= 3.0.0)

Config/testthat/edition 3

Imports dplyr, ggplot2, grid, gridExtra, magrittr, parallel, patchwork (>= 1.1.0), pbapply, rje, rlang, RTMB (>= 1.6), stats, tibble, tidy, 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],
badp authors [cph]

Maintainer Marcin Dubel <marcindubel@gmail.com>

Repository CRAN

Date/Publication 2026-09-22 20:50:02 UTC

Contents

best_models	3
bma	4
bma_table	8
coef.badp_bma	9
coef.badp_model	11
coef.badp_model_space	12
coef_hist	13
compute_model_space_stats	14
convergence	16
economic_growth	17
feature_standardization	18
full_bma_results	19
full_model_space	19
jointness	20
join_lagged_col	21
logLik.badp	22
migration_data	24
migration_model_space	24
migration_model_space_nonnested	25
model_pmp	26
model_sizes	27
model_size_table	28
model_space_nonnested	29
model_stats	30
model_table	31
n_models	33
optim_model_space	34
optim_model_space_params	37
original_economic_growth	40
pip	41
plot.badp_best_models	42
plot.badp_bma	43
pmp	44
posterior_dens	45
print.badp_best_models	46
print.badp_bma	47
print.badp_bma_coef	48
print.badp_model	49
print.badp_model_space	49
print.badp_plots	50
print.summary.badp_best_models	51
print.summary.badp_bma	52

print.summary.badp_model	53
print.summary.badp_model_space	53
regressors	54
sem_likelihood	55
small_model_space	57
summary.badp_best_models	57
summary.badp_bma	58
summary.badp_model	59
summary.badp_model_space	60
weighting	61
[.badp_best_models	62

Index	64
-------	----

best_models	<i>Best Models by Posterior Model Probability</i>
-------------	---

Description

Ranks the models in a fitted model space by their posterior model probability and returns the highest ranked ones as a list of individual models, each carrying its own estimates and diagnostics.

Usage

```
best_models(x, prior = "binomial", best = 5, round = 3)
```

Arguments

x	An object of class badp_bma, typically the result of bma .
prior	Model prior used to rank the models: "binomial" (the default) or "beta" for the binomial-beta prior.
best	Number of models to return. If it exceeds the size of the model space, every model is returned and a message is issued.
round	Number of decimal places used when the estimates are formatted for display. The stored values are not rounded.

Details

The object returned is a list of badp_model objects, one per selected model, ordered from the highest to the lowest posterior model probability. Individual models are reached by position, for example best_models(bma_results)[[1]] for the best one, and each has its own print, summary and coef methods. The container has print, summary and plot methods that display the whole selection.

Value

An object of class badp_best_models: a list of badp_model objects with attributes prior, n_models (the size of the model space), reg_names and digits.

See Also

[bma](#), [summary.badp_best_models](#), [plot.badp_best_models](#), [print.badp_model](#)

Examples

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

best <- best_models(results, best = 5)
best           # the selection
best[[1]]      # the single best model
coef(best[[1]]) # its coefficients
summary(best, robust = TRUE)
```

bma

Bayesian Model Averaging over a Model Space

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 <code>badp_model_space</code> , 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 <code>dilution = 1</code> . 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 maximized 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 factorizes 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. $\text{eta} = 1$ reproduces "mb2012" and $\text{eta} = 1/N$ 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 $\eta A_j = A_j/N \rightarrow -c_j$, 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 $\eta = \text{rank}(J)/\text{tr}(H^{-1}J)$, 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 errors and robust standard errors, used by 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 binomial and binomial-beta model priors distributed over model sizes, used by 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.

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`.

weighting The approximation to the marginal likelihood used to weight the models.

eta The learning rate the weighting implies, or NA for "uip", which alters the penalty rather than the rate.

loglik The maximized log-likelihood of every model, in the order of the model space.

n_params The number of estimated parameters of every model, in the order of the model space.

nobs The number of observations used in the estimation.

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 centering 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

The quantities of interest are returned by accessor functions, so that no analysis needs to depend on the internal layout of the object:

- [bma_table](#) - Statistics under either model prior
- [pip](#) - Posterior inclusion probabilities
- [pmp](#) - Posterior model probabilities
- [model_size_table](#) - Prior and posterior model sizes
- [regressors, n_models](#) - Size of the problem
- [weighting, learning_rate](#) - Marginal likelihood used

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
)
```

bma_table	<i>Extract the BMA Statistics Table</i>
-----------	---

Description

Returns the table of Bayesian model averaging statistics computed under one of the two model priors, without reaching into the internal structure of the object.

Usage

```
bma_table(x, ...)
```

```
## S3 method for class 'badp_bma'
bma_table(x, prior = c("binomial", "beta"), ...)
```

Arguments

- x An object of class badp_bma, typically the result of [bma](#).
- ... Arguments passed to methods.
- prior Model prior: "binomial" (the default) or "beta" for the binomial-beta prior.

Details

The columns are PIP, the posterior inclusion probability; PM and PSD, the posterior mean and posterior standard deviation; PSDR, the posterior standard deviation built from robust standard errors; PMcon, PSDcon and PSDRcon, the same three quantities conditional on the regressor being included; and %(+), the percentage of models in which the coefficient is positive. The first row is the lagged dependent variable, which enters every model by construction.

Value

A numeric matrix with one row per parameter and eight columns.

See Also

[bma](#), [pip](#), [coef.badp_bma](#), [summary.badp_bma](#)

Examples

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

bma_table(results)
bma_table(results, prior = "beta")
```

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) <i>conditional on inclusion</i> 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 behavior)
coef(results, prior = "binomial", PIP = FALSE)
```

coef.badp_model *Coefficients of a Single Model*

Description

Extracts the maximum likelihood estimates of one of the best models.

Usage

```
## S3 method for class 'badp_model'
coef(object, se = FALSE, include_absent = FALSE, digits = NULL, ...)
```

Arguments

object	An object of class <code>badp_model</code> , an element of the list returned by best_models .
se	Logical. If TRUE, a matrix with the estimates, their standard errors, robust standard errors and the corresponding p-values is returned instead of the estimates alone. The p-values are Wald p-values referred to the standard normal distribution.
include_absent	Logical. If FALSE (the default), parameters excluded from the model are dropped rather than returned as NA.
digits	Number of decimal places to round the result to. By default the values are returned exactly as estimated; supplying digits is a convenience for displaying them, equivalent to wrapping the call in round . The print and summary methods round to the number of digits given to best_models .
...	Additional arguments (currently unused).

Value

A named numeric vector, or a matrix when `se = TRUE`.

See Also

[best_models](#), [summary.badp_model](#)

Examples

```
data(small_model_space)
best <- best_models(bma(small_model_space), best = 3)

coef(best[[1]])
coef(best[[1]], se = TRUE)
coef(best[[1]], se = TRUE, digits = 3)
coef(best[[1]], include_absent = TRUE)
```

coef.badp_model_space *Estimated Parameters of a Model Space*

Description

Returns the maximum likelihood estimates of the parameters of every model in the model space, or of a single model.

Usage

```
## S3 method for class 'badp_model_space'
coef(object, model = NULL, ...)
```

Arguments

object	An object of class badp_model_space.
model	Optional position of a single model, as reported in the model column of model_table . By default all models are returned.
...	Further arguments, currently unused.

Details

Parameters not present in a model are NA. The rows are labeled as follows: alpha is the coefficient on the lagged dependent variable, beta_<name> the coefficient on regressor <name>; phi_0, err_var and dep_var_<t> parameterize the variance of the dependent variable and its covariance with the initial condition; phi_1_<name> the covariance between the initial condition and the regressors; and phis_<i> and psis_<i> are the remaining covariance parameters of the regressors, numbered in the order in which [sem_sigma_matrix](#) consumes them.

Value

Without model, a numeric matrix with one row per parameter and one column per model, the columns named model_1, model_2, and so on. With model, a named numeric vector holding the parameters estimated in that model.

See Also

[model_stats](#), [model_table](#), [optim_model_space](#)

Examples

```
data(small_model_space)

coef(small_model_space)[1:10, 1:4]
coef(small_model_space, model = n_models(small_model_space))
```

coef_hist

*Histograms 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(
  x,
  weight = NULL,
  bin_method = c("FD", "SC", "vec"),
  bin_widths = NULL,
  use_bin_count = 0,
  bin_counts = NULL,
  use_kernel = 0
)
```

Arguments

x	An object of class <code>badp_bma</code> , typically returned by bma .
weight	Parameter indicating whether the coefficients should be weighted by posterior model probabilities: 1. <code>NULL</code> - no weighting (default option) 2. <code>"binomial"</code> - using posterior model probabilities based on binomial model prior 3. <code>"beta"</code> - using posterior model probabilities based on binomial-beta model prior
bin_method	Character string specifying the method for bin widths (default: <code>"FD"</code>). One of: <code>"FD"</code> - Freedman-Diaconis method; <code>"SC"</code> - Scott method; <code>"vec"</code> - user specified bin widths provided through a vector (parameter: <code>bin_widths</code>).
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 <code>bin_method = "vec"</code> .
use_bin_count	Parameter taking the values (default: <code>use_bin_count = 0</code>): 1 - the histogram will be built based on the number of bins specified by the user through parameter <code>bin_counts</code>. If <code>use_bin_count = 1</code>, the function ignores parameter <code>bin_method</code>. 0 - the histogram will be built in line with parameter <code>bin_method</code>.

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

Statistics and Standard Deviations for the Model Space

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

<code>df</code>	Data frame with data for the SEM analysis.
<code>dep_var_col</code>	Column with the dependent variable
<code>timestamp_col</code>	The name of the column with timestamps
<code>entity_col</code>	Column with entities (e.g. countries)
<code>params</code>	A matrix (with named rows) with each column corresponding to a model. Each column specifies model parameters. Compare with optim_model_space_params
<code>nested</code>	Logical. If TRUE (default), compute approximate standard deviations using the nested-model approach via <code>nested_std_dev_from_params()</code> . If FALSE, use the non-nested approach via <code>non_nested_std_dev_from_params()</code> . The choice affects which approximation routine is used for each model in <code>params</code> .
<code>exact_value</code>	Whether the exact value of the likelihood should be computed (TRUE) or just the proportional part (FALSE). Check sem_likelihood for details.
<code>model_prior</code>	Which model prior to use. For now there are two options: 'uniform' and 'binomial-beta'. Default is 'uniform'.
<code>cl</code>	An optional cluster object. If supplied, the function will use this cluster for parallel processing. If NULL (the default), <code>pbapply::pblapply</code> 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 $\exp((\log \text{lik} - (k/2) * \log(N * T)) / N)$; 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 $\text{tr}(H^{-1} J)$, 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 $\text{rank}(J) / \text{tr}(H^{-1} J)$; 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
)
```

convergence

Per-Model Convergence Diagnostics

Description

Returns the diagnostics recorded for the numerical optimization of every model in the model space: whether it converged, the code returned by [optim](#), the number of restarts and of initial draws, and the largest absolute gradient at the reported optimum.

Usage

```
convergence(x, ...)
```

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

Arguments

x An object of class `badp_model_space`.

... Arguments passed to methods.

Details

Models that failed to converge are retained in the model space rather than dropped, so that they can be inspected. See [optim_model_space](#) for what the individual diagnostics mean.

Value

A numeric matrix with one column per model.

See Also

[optim_model_space](#), [summary.badp_model_space](#)

Examples

```
data(full_model_space)
convergence(full_model_space)[, 1:5]
```

economic_growth	<i>Economic Growth Data</i>
-----------------	-----------------------------

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

<code>df</code>	Data frame with the data.
<code>excluded_cols</code>	Unquoted column names to exclude from standardization. If missing, all columns are standardized.
<code>group_by_col</code>	Unquoted column names to group the data by before applying standardization. If missing, no grouping is performed.
<code>scale</code>	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	<i>Example Bayesian Model Averaging Results</i>
------------------	---

Description

A `badp_bma` object summarising the BMA analysis

Usage

```
full_bma_results
```

Format

An object of class `badp_bma`

full_model_space	<i>Example Model Space</i>
------------------	----------------------------

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 $\text{tr}(H^{-1} J)$, 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`.

jointness

*Bivariate 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(x, measure = c("HCGHM", "LS", "DW", "PPI"), rho = 0.5, round = 3)
```

Arguments

x	An object of class <code>badp_bma</code> , 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)
```

join_lagged_col

Merge a Lagged Column into a Single Column

Description

Turns data in which a quantity is stored twice, once in levels and once lagged in time, into the single-column form the rest of the package expects.

Usage

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

Arguments

<code>df</code>	Data frame with a column holding lagged values
<code>col</code>	Column with quantity not lagged
<code>col_lagged</code>	Column with the same quantity as <code>col</code> , but the values are lagged in time
<code>timestamp_col</code>	Column with timestamps (e.g. years)
<code>entity_col</code>	Column with entities (e.g. countries)
<code>timestep</code>	Difference between timestamps (e.g. 10)

Details

Some panel data sets ship the dependent variable in two columns, the value at time t and the value at time $t - 1$, held side by side in the same row. The functions in **badp** build the lags themselves from the panel structure, so they need the quantity only once. This function performs that reduction: the two columns are merged into one and the rows are re-indexed so that no observation is lost.

The step is conditional. Data that already store the quantity once, as [economic_growth](#) and [migration_data](#) do, pass straight to [feature_standardization](#) and do not need this function at all.

Value

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

See Also

[feature_standardization](#)

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)
```

Description

Returns the maximized log-likelihood of a model as an object of class "logLik", so that the generic tools of the **stats** package, such as [AIC](#) and [BIC](#), apply to it.

Usage

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

## S3 method for class 'badp_model'
logLik(object, ...)
```

Arguments

object	An object of class <code>badp_model_space</code> , or a single model of class <code>badp_model</code> as returned within best_models .
model	For a model space, the position of the model, as reported in the <code>model</code> column of model_table . A single integer.
...	Further arguments, currently unused.

Details

The value is the exact log-likelihood of the model evaluated at the estimates, including all constants, whatever `exact_value` was used during the optimization. The `df` attribute counts every estimated parameter of the model, the regression coefficients as well as the parameters of the covariance structure, and `nobs` is the number of observations used in the estimation.

Note that [BIC](#) applied to these objects is the textbook criterion, which penalizes all `df` parameters. The approximation to the marginal likelihood that [bma](#) uses to weight the models penalizes only the regression coefficients, so the two need not rank the models identically.

Value

An object of class "logLik": a number with attributes `df` and `nobs`.

See Also

[model_table](#), [best_models](#), [logLik](#)

Examples

```
data(small_model_space)

# the model with all regressors
l1 <- logLik(small_model_space, model = n_models(small_model_space))
l1
AIC(l1)
BIC(l1)

results <- bma(small_model_space)
best <- best_models(results, best = 2)
logLik(best[[1]])
```

migration_data	<i>Migration Data in the Original Format</i>
----------------	--

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](https://doi.org/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](https://doi.org/10.7910/DVN/GTOFJB)

migration_model_space	<i>Example Model Space for the Migration Data</i>
-----------------------	---

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 $\text{tr}(H^{-1} J)$, 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.

convergence A matrix of per-model convergence diagnostics; see [optim_model_space](#).

migration_model_space_nonnested

Example Non-Nested Model Space for the Migration Data

Description

A `badp_model_space` object created with [optim_model_space](#) using the `migration_data` dataset with the non-nested 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 $\text{tr}(H^{-1} J)$, 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.

convergence A matrix of per-model convergence diagnostics; see [optim_model_space](#).

model_pmp

Graphs of the Prior and Posterior Probabilities of 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(x, top = NULL, type = c("line", "histogram"))
```

Arguments

- | | |
|------|--|
| x | An object of class <code>badp_bma</code> , typically returned by bma . |
| top | The number of the best model to be placed on the graphs |
| type | Character, either "line" (the default) for the prior and posterior drawn as lines against the model ranking, or "histogram" for them drawn as side-by-side bars. The bar form suits a small top, where individual models can still be told apart; with a large top the lines are easier to read. |

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)

# bars instead of lines
model_pmp(bma_results, top = 5, type = "histogram")
```

model_sizes

Graphs of the Prior and Posterior Probabilities of 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(x, type = c("line", "histogram"))
```

Arguments

x	An object of class <code>badp_bma</code> , typically returned by bma .
type	Character, either "line" (the default) for the prior and posterior drawn as lines against model size, or "histogram" for them drawn as side-by-side bars. Bars read well when there are few model sizes; with many regressors the line form stays legible where the bars become crowded.

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)

# bars instead of lines
model_sizes(bma_results, type = "histogram")
```

model_size_table

Extract the Prior and Posterior Model Sizes

Description

Returns the table of prior and posterior expected model sizes under the binomial and binomial-beta model priors. Model size counts regressors and excludes the lagged dependent variable, which is present in every model.

Usage

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

Arguments

x	An object of class badp_bma.
...	Arguments passed to methods.

Value

A numeric matrix with two rows, one per model prior, and two columns holding the prior and the posterior expected model size.

See Also

[bma](#), [model_sizes](#)

Examples

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

model_space_nonnested *Example 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 with 106 rows and 512 columns (corresponding to $2^9 = 512$ models), containing parameter values for the 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 $\text{tr}(H^{-1} J)$, 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`.

model_stats

Statistics of the Models in a Model Space

Description

Returns the statistics computed for every model of the model space after the estimation: the maximized log-likelihood, the approximation to the marginal likelihood used to weight the models, and the conventional and robust standard errors of the coefficients, together with the diagnostics of the robust covariance.

Usage

```
model_stats(x, ...)

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

Arguments

x An object of class `badp_model_space`.

... Arguments passed to methods.

Details

The rows are

`loglik` The maximized log-likelihood; see also [logLik.badp_model_space](#).

`marg_lik` The approximation to the marginal likelihood used by [bma](#) to weight the models, $\exp((\loglik - (k/2) * \log(N * T)) / N)$. It is not a BIC.

`se_<name>` The standard error of the coefficient on `<name>`, from the observed information, for the lagged dependent variable and each regressor. Zero when the regressor is not in the model.

`robust_se_<name>` The corresponding robust (sandwich) standard error.

`trace_HinvJ`, `n_params`, `rank_J` The trace of $H^{-1}J$, the number of estimated parameters and the numerical rank of J ; see [optim_model_space](#). Absent from model spaces estimated before these were stored.

Value

A numeric matrix with the rows described above and one column per model, the columns named `model_1`, `model_2`, and so on.

See Also

[coef.badp_model_space](#), [model_table](#), [logLik.badp_model_space](#)

Examples

```
data(small_model_space)

model_stats(small_model_space)[, 1:4]
model_stats(small_model_space)["loglik", ]
```

model_table	<i>Table of the Models in a Model Space</i>
-------------	---

Description

Returns one row per model of the model space, describing which regressors the model includes and how well it fits. This is the direct way to rank or filter the estimated models, for example by their maximized log-likelihood, without reaching into the parameter and statistics matrices.

Usage

```
model_table(x, ...)

## S3 method for class 'badp_model_space'
model_table(x, sort_by = c("none", "loglik", "size"), top = NULL, ...)
```

Arguments

<code>x</code>	An object of class <code>badp_model_space</code> .
<code>...</code>	Arguments passed to methods.
<code>sort_by</code>	Ordering of the rows: "none" (the default) keeps the order in which the models are held in the model space, "loglik" sorts by decreasing log-likelihood, and "size" by increasing number of regressors, ties broken by decreasing log-likelihood.
<code>top</code>	Optional number of rows to return, taken after sorting.

Details

A regressor counts as included in a model when its coefficient `beta_<name>` was estimated in that model, that is, when it is not NA; a coefficient estimated at exactly zero is an estimate like any other and counts as included. The lagged dependent variable enters every model and is not counted in size.

The log-likelihood is the exact log-likelihood of the model evaluated at the estimates, including all constants, whatever `exact_value` was used during the optimization. It is also available as a "logLik" object from [logLik.badp_model_space](#).

Value

A data frame with one row per model and columns

`model` Position of the model in the model space, the index of its column in the parameter matrix.

`size` Number of regressors included.

`regressors` Names of the included regressors, separated by commas, or "(none)" for the model with the lagged dependent variable only.

`loglik` Maximized log-likelihood.

`converged` Logical, whether the optimization converged. Only present when the model space records convergence diagnostics; see [convergence](#).

See Also

[optim_model_space](#), [convergence](#), [regressors](#), [n_models](#)

Examples

```
data(small_model_space)

model_table(small_model_space)

# the five best-fitting models
model_table(small_model_space, sort_by = "loglik", top = 5)
```

n_models	<i>Size of the Model Space</i>
----------	--------------------------------

Description

Returns the number of models over which the averaging is performed. The lagged dependent variable enters every model and is never averaged over, so the count is two to the power of `length(regressors(x))`, not of the total number of columns.

Usage

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

Arguments

x	An object of class <code>badp_bma</code> or <code>badp_model_space</code> .
...	Arguments passed to methods.

Value

A single integer.

See Also

[bma](#), [optim_model_space](#)

Examples

```
data(full_model_space)  
n_models(full_model_space)
```

optim_model_space	<i>Estimate the Model Space</i>
-------------------	---------------------------------

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. <code>function(n) runif(n, 0.1, 1)</code>). 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 <code>init_value = 0.5</code> is equivalent to <code>function(n) rep(0.5, n)</code> . 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 <code>nested_std_dev_from_params()</code> . If FALSE, use the non-nested approach via <code>non_nested_std_dev_from_params()</code> . The choice affects which approximation routine is used for each model in <code>params</code> .
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), <code>pbapply::pblapply</code> will run sequentially.
control	a list of control parameters for the optimization which are passed to <code>optim</code> . Default is <code>list(trace = 0, maxit = 10000, fnscale = -1, REPORT = 100, scale = 0.05)</code> , but note that <code>scale</code> is used only for adjusting the <code>parscale</code> 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 <code>init_value</code> , 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 <code>converged = 0</code> ; 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 <code>init_value</code> 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 <code>init_value</code> , and only after <code>max_init_attempts</code> unsuccessful draws is an error raised. The number of draws each model actually needed is reported in the <code>n_init_draws</code> 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
- [summary.badp_model_space](#) - Dimensions, variables and diagnostics of the model space
- [model_table](#) - One row per model: regressors, size, log-likelihood and convergence
- [coef.badp_model_space](#) - Estimated parameters
- [model_stats](#) - Log-likelihoods, standard errors and other statistics of every model
- [logLik.badp_model_space](#) - Log-likelihood of one model as a "logLik" object
- [regressors](#), [n_models](#), [convergence](#) - Further accessors

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

Maximum Likelihood Parameters for Every Model in the 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 = 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

<code>df</code>	Data frame with data for the analysis.
<code>timestamp_col</code>	The name of the column with time stamps.
<code>entity_col</code>	Column with entities (e.g. countries).
<code>dep_var_col</code>	Column with the dependent variable.
<code>init_value</code>	Function of one argument <code>n</code> returning <code>n</code> starting values for the numerical optimization (e.g. <code>function(n) runif(n, 0.1, 1)</code>). 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 <code>init_value = 0.5</code> is equivalent to <code>function(n) rep(0.5, n)</code> . Starting values must be non-zero, because zeros encode excluded parameters.
<code>nested</code>	Logical. If TRUE (default), compute approximate standard deviations using the nested-model approach via <code>nested_std_dev_from_params()</code> . If FALSE, use the non-nested approach via <code>non_nested_std_dev_from_params()</code> . The choice affects which approximation routine is used for each model in <code>params</code> .
<code>exact_value</code>	Whether the exact value of the likelihood should be computed (TRUE) or just the proportional part (FALSE). Check sem_likelihood for details.
<code>cl</code>	An optional cluster object. If supplied, the function will use this cluster for parallel processing. If NULL (the default), <code>pbapply::pblapply</code> will run sequentially.
<code>control</code>	a list of control parameters for the optimization which are passed to optim . Default is <code>list(trace = 0, maxit = 10000, fnscale = -1, REPORT = 100, scale = 0.05)</code> .
<code>max_restarts</code>	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.
<code>restart_tol</code>	Log-likelihood improvement between restarts below which the optimization is considered converged. Improvements of this size are immaterial for posterior model probabilities. Default is $1e-3$.
<code>max_reoptimizations</code>	Maximum number of times a model is re-optimized from a fresh starting point drawn from <code>init_value</code> , 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 <code>converged = 0</code> ; 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.

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
  )

# the parameter matrix only; optim_model_space() adds the statistics
params <- optim_model_space_params(
  df           = data_prepared,
  dep_var_col  = gdp,
  timestamp_col = year,
  entity_col   = country,
  init_value   = function(n) rep(0.5, n),
  nested      = TRUE
)
dim(params)
```

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/>

`pip`*Extract Posterior Inclusion Probabilities*

Description

Returns the posterior inclusion probability of each regressor.

Usage

```
pip(x, ...)  
  
## S3 method for class 'badp_bma'  
pip(x, prior = c("binomial", "beta"), include_lagged = FALSE, ...)
```

Arguments

<code>x</code>	An object of class <code>badp_bma</code> .
<code>...</code>	Arguments passed to methods.
<code>prior</code>	Model prior: "binomial" (the default) or "beta".
<code>include_lagged</code>	Logical. Include the lagged dependent variable, whose inclusion probability is undefined. Defaults to FALSE.

Details

The lagged dependent variable enters every model by construction and therefore carries no inclusion probability. It is omitted unless `include_lagged = TRUE`, in which case it appears first with value NA.

Value

A named numeric vector of posterior inclusion probabilities.

See Also

[bma_table](#), [pmp](#), [bma](#)

Examples

```
data(full_model_space)  
results <- bma(full_model_space)  
  
pip(results)  
sort(pip(results), decreasing = TRUE)
```

`plot.badp_best_models` *Plot a Best-Models Selection*

Description

Draws the inclusion table or the table of estimates for the selected models to the active graphics device.

Usage

```
## S3 method for class 'badp_best_models'
plot(x, which = c("estimates", "inclusion"), robust = FALSE, ...)
```

Arguments

<code>x</code>	An object of class <code>badp_best_models</code> .
<code>which</code>	Which table to draw: "estimates" (the default) or "inclusion".
<code>robust</code>	Logical. If TRUE, robust standard errors are used in the table of estimates.
<code>...</code>	Additional arguments (currently unused).

Value

Invisibly returns the drawn table.

See Also

[best_models](#), [print.badp_best_models](#)

Examples

```
data(small_model_space)
best <- best_models(bma(small_model_space), best = 3)

plot(best)
plot(best, which = "inclusion")
```

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 = c("model_sizes", "model_pmp", "best_models", "jointness", "coef_hist",
    "posterior_dens"),
  ...
)
```

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.

With which = "best_models" two functions are involved, [best_models](#) to select the models and [plot.badp_best_models](#) to draw them, and the arguments in ... are split between them by name: prior, best and round are passed to the first, robust to the second. The table drawn is the one of estimates; for the inclusion table call plot(best_models(x), which = "inclusion") directly, since which is taken here by the choice of plot.

Value

Invisibly returns the object produced by the selected helper. The figure or table is drawn to the active device as a side effect, so a single call draws exactly one graphic.

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")
```

pmp

Extract Posterior Model Probabilities

Description

Returns the posterior probability of each model in the model space.

Usage

```
pmp(x, ...)
```

```
## S3 method for class 'badp_bma'
pmp(x, prior = c("binomial", "beta"), top = NULL, ...)
```

Arguments

x	An object of class <code>badp_bma</code> .
...	Arguments passed to methods.
prior	Model prior: "binomial" (the default) or "beta".
top	Optional number of models to return, ordered by decreasing posterior probability. By default every model is returned, unsorted.

Value

A numeric vector of posterior model probabilities, one per model, in the order in which the models are held in the model space unless `top` is given.

See Also

[pip](#), [best_models](#), [model_pmp](#)

Examples

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

sum(pmp(results))      # sums to one
pmp(results, top = 5)  # the five most probable models
```

posterior_dens

*Kernel Densities of the Coefficients***Description**

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

Usage

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

Arguments

x	An object of class <code>badp_bma</code> , typically returned by bma .
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 a Best-Models Selection

Description

Prints, to the console, which regressors enter each of the selected models together with their posterior model probabilities.

Usage

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

Arguments

x An object of class `badp_best_models`.

... Additional arguments (currently unused).

Value

Invisibly returns `x`.

See Also

[best_models](#), [summary.badp_best_models](#), [plot.badp_best_models](#)

Examples

```

data(small_model_space)
best <- best_models(bma(small_model_space), best = 3)

best

```

print.badp_bma	<i>Print Bayesian Model Averaging Results</i>
----------------	---

Description

Print method for objects of class badp_bma.

Usage

```
## S3 method for class 'badp_bma'  
print(x, n = 10, ...)
```

Arguments

x	An object of class badp_bma, typically the result of bma .
n	Maximum number of regressors to list: a single non-negative number, or Inf for all of them. Defaults to 10.
...	Additional arguments (currently unused).

Details

print gives a compact overview: the size of the model space, the prior and weighting settings, and the regressors ordered by posterior inclusion probability. Use [summary.badp_bma](#) for the full BMA statistics under both model priors, and [coef.badp_bma](#) to extract posterior means and standard deviations.

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

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

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`

Examples

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

coef(results)
print(coef(results, se = TRUE), digits = 3)
```

```
print.badp_model
```

Print a Single Model

Description

Prints the regressors included in one of the best models together with its posterior model probability and its coefficient table.

Usage

```
## S3 method for class 'badp_model'
print(x, robust = FALSE, ...)
```

Arguments

x	An object of class badp_model.
robust	Logical. If TRUE, robust standard errors are shown.
...	Additional arguments (currently unused).

Value

Invisibly returns x.

See Also

[best_models](#), [summary.badp_model](#)

Examples

```
data(small_model_space)
best <- best_models(bma(small_model_space), best = 3)

best[[1]]
print(best[[1]], robust = TRUE)
```

```
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.badp_plots	<i>Print a Collection of Plots</i>
------------------	------------------------------------

Description

Print method for the plot collections returned by `model_sizes`, `model_pmp`, `coef_hist` and `posterior_dens`.

Usage

```
## S3 method for class 'badp_plots'
print(x, which = NULL, ...)
```

Arguments

x An object of class `badp_plots`.
which Optional name or position of a single plot to draw instead of the default figure. A name must be one of `names(x)` and a position an index between 1 and `length(x)`.
... Additional arguments (currently unused).

Details

A bare list of **ggplot2** objects is auto-printed one element at a time, so on an interactive device each plot replaces the one before it and only the last remains visible. These functions therefore return a classed collection, and this method draws a single figure: the combined plot when the collection provides one, and otherwise every plot arranged together with [wrap_plots](#). Individual plots are still reached by name or position and printed on their own.

Value

Invisibly returns x.

See Also

[model_sizes](#), [model_pmp](#), [coef_hist](#), [posterior_dens](#)

Examples

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

sizes <- model_sizes(results)
sizes                                     # one combined figure
print(sizes, which = "binomial")
names(sizes)
```

```
print.summary.badp_best_models
```

Print the Summary of a Best-Models Selection

Description

Print the Summary of a Best-Models Selection

Usage

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

Arguments

x	An object of class <code>summary.badp_best_models</code> .
...	Additional arguments (currently unused).

Value

Invisibly returns x.

See Also[summary.badp_best_models](#)**Examples**

```
data(small_model_space)
best <- best_models(bma(small_model_space), best = 3)

print(summary(best))
```

`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

<code>x</code>	An object of class <code>summary.badp_bma</code> .
<code>...</code>	Additional arguments (currently unused).

Value

Invisibly returns the input object `x`.

See Also[summary.badp_bma](#)**Examples**

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

print(summary(results))
```

```
print.summary.badp_model
```

Print the Summary of a Single Model

Description

Print the Summary of a Single Model

Usage

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

Arguments

x	An object of class <code>summary.badp_model</code> .
...	Additional arguments (currently unused).

Value

Invisibly returns x.

See Also

[summary.badp_model](#)

Examples

```
data(small_model_space)
best <- best_models(bma(small_model_space), best = 3)

print(summary(best[[1]]))
```

```
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](#)

Examples

```
data(small_model_space)

print(summary(small_model_space))
```

regressors

Names of the Regressors

Description

Returns the names of the regressors considered in the analysis.

Usage

```
regressors(x, ...)
```

```
## S3 method for class 'badp_bma'
regressors(x, include_lagged = FALSE, ...)
```

```
## S3 method for class 'badp_model_space'
regressors(x, include_lagged = FALSE, ...)
```

Arguments

`x` An object of class `badp_bma` or `badp_model_space`.
`...` Arguments passed to methods.
`include_lagged` Logical. Include the lagged dependent variable, which is the first name and enters every model. Defaults to `FALSE`.

Value

A character vector of regressor names.

See Also

[bma](#), [optim_model_space](#)

Examples

```
data(full_model_space)
regressors(full_model_space)
regressors(full_model_space, include_lagged = TRUE)
```

sem_likelihood	<i>Likelihood for the SEM Model</i>
----------------	-------------------------------------

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 data frame. If a data frame, 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 $\text{trace}(\text{HG}_{22})$ 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 (Σ_{11} matrix, σ_{ϵ}^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. $n_{\text{regressors}} * (n_{\text{periods}} - 1)$

`psis` double vector which together with `phis` 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. $n_{\text{regressors}} * (n_{\text{periods}} - 1) * n_{\text{periods}} / 2$

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	<i>Example Model Space (Small Version)</i>
-------------------	--

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 $\text{tr}(H^{-1} J)$, 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.

convergence A matrix of per-model convergence diagnostics; see [optim_model_space](#).

```
summary.badp_best_models
```

Summarize a Best-Models Selection

Description

Summarize a Best-Models Selection

Usage

```
## S3 method for class 'badp_best_models'
summary(object, robust = FALSE, ...)
```

Arguments

object	An object of class badp_best_models.
robust	Logical. If TRUE, robust standard errors are shown in the table of estimates.
...	Additional arguments (currently unused).

Value

An object of class summary.badp_best_models holding the inclusion table, the table of formatted estimates, the posterior model probabilities and the regressors entering each model.

See Also

[best_models](#), [print.badp_best_models](#)

Examples

```
data(small_model_space)
best <- best_models(bma(small_model_space), best = 3)

summary(best)
summary(best, robust = TRUE)
```

summary.badp_bma	<i>Summarize Bayesian Model Averaging Results</i>
------------------	---

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
- `weighting` - Marginal-likelihood approximation used
- `eta` - Realized learning rate
- `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)
```

<code>summary.badp_model</code>	<i>Summarize a Single Model</i>
---------------------------------	---------------------------------

Description

Summarize a Single Model

Usage

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

Arguments

<code>object</code>	An object of class <code>badp_model</code> .
<code>...</code>	Additional arguments (currently unused).

Value

An object of class `summary.badp_model` holding the rank, the posterior model probability, the included regressors and the full coefficient matrix.

See Also

[best_models](#), [print.badp_model](#)

Examples

```
data(small_model_space)
best <- best_models(bma(small_model_space), best = 3)

summary(best[[1]])
```

```
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

<code>object</code>	An object of class <code>badp_model_space</code> , typically the result of optim_model_space .
<code>...</code>	Additional arguments (currently unused).

Value

An object of class `summary.badp_model_space` containing:

- `num_models` - Number of models in the model space, two to the power of `num_regressors`.
- `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)
```

weighting

Marginal-Likelihood Weighting Used

Description

Returns the approximation to the marginal likelihood used to weight the models, and the learning rate it implies.

Usage

```
weighting(x, ...)

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

learning_rate(x, ...)

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

Arguments

`x` An object of class `badp_bma`.
`...` Arguments passed to methods.

Details

The four approximations are described in [bma](#). Three of them share the same construction at different learning rates, so the rate is the quantity that distinguishes them; the fourth, "uip", alters the penalty instead and has no learning rate.

Value

For weighting, a character string. For learning_rate, a single number, or NA when the weighting does not correspond to a learning rate.

See Also

[bma](#)

Examples

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

weighting(results)
learning_rate(results)
```

[.badp_best_models] *Subset a Best-Models Selection*

Description

Keeps the class and the attributes of the selection when a subset of the models is taken.

Usage

```
## S3 method for class 'badp_best_models'
x[i]
```

Arguments

x	An object of class badp_best_models.
i	Indices of the models to keep.

Value

An object of class badp_best_models.

Examples

```
data(small_model_space)
best <- best_models(bma(small_model_space), best = 5)

best[1:2]  # a badp_best_models object holding the two best models
best[[1]]  # a single badp_model
```

Index

* datasets

- economic_growth, 17
- full_bma_results, 19
- full_model_space, 19
- migration_data, 24
- migration_model_space, 24
- migration_model_space_nonnested, 25
- model_space_nonnested, 29
- original_economic_growth, 40
- small_model_space, 57
- [.badp_best_models, 62
- AIC, 22
- best_models, 3, 11, 23, 42–44, 46, 49, 58, 60
- BIC, 22, 23
- bma, 3, 4, 4, 8–10, 13, 15, 20, 23, 26, 27, 29, 31, 33, 36, 41, 43–45, 47, 50, 55, 58, 59, 61, 62
- bma_table, 7, 8, 41
- coef.badp_bma, 7, 8, 9, 47, 48, 59
- coef.badp_model, 11
- coef.badp_model_space, 12, 31, 36
- coef_hist, 13, 44, 50, 51
- compute_model_space_stats, 14, 19, 25, 30, 57
- convergence, 16, 32, 36
- economic_growth, 17, 19, 22, 29, 57
- feature_standardization, 18, 22
- full_bma_results, 19
- full_model_space, 19, 29
- join_lagged_col, 21
- jointness, 20, 44
- learning_rate, 7
- learning_rate (weighting), 61

- logLik, 23
- logLik.badp, 22
- logLik.badp_model (logLik.badp), 22
- logLik.badp_model_space, 31, 32, 36
- logLik.badp_model_space (logLik.badp), 22
- migration_data, 22, 24, 24, 25
- migration_model_space, 24
- migration_model_space_nonnested, 25
- model_pmp, 26, 44, 50, 51
- model_size_table, 7, 28
- model_sizes, 27, 29, 44, 50, 51
- model_space_nonnested, 29
- model_stats, 12, 30, 36
- model_table, 12, 23, 31, 31, 36
- n_models, 7, 32, 33, 36
- optim, 16, 35, 36, 38, 39
- optim_model_space, 4, 12, 16, 17, 19, 24–26, 29–33, 34, 50, 55, 57, 60, 61
- optim_model_space_params, 15, 37
- original_economic_growth, 40
- pip, 7, 8, 41, 44
- plot.badp_best_models, 4, 42, 43, 46
- plot.badp_bma, 7, 43
- pmp, 7, 41, 44
- posterior_dens, 44, 45, 50, 51
- print.badp_best_models, 42, 46, 58
- print.badp_bma, 7, 47, 59
- print.badp_bma_coef, 9, 10, 48
- print.badp_model, 4, 49, 60
- print.badp_model_space, 36, 49, 61
- print.badp_plots, 50
- print.data.frame, 48
- print.summary.badp_best_models, 51
- print.summary.badp_bma, 52
- print.summary.badp_model, 53

`print.summary.badp_model_space`, [53](#), [61](#)

`regressors`, [7](#), [32](#), [36](#), [54](#)

`round`, [11](#)

`score_rank`, [6](#), [7](#), [15](#), [19](#), [25](#), [30](#), [36](#), [57](#)

`sem_likelihood`, [15](#), [34](#), [38](#), [55](#)

`sem_sigma_matrix`, [12](#)

`small_model_space`, [57](#)

`summary.badp_best_models`, [4](#), [46](#), [52](#), [57](#)

`summary.badp_bma`, [7](#), [8](#), [10](#), [47](#), [52](#), [58](#)

`summary.badp_model`, [11](#), [49](#), [53](#), [59](#)

`summary.badp_model_space`, [17](#), [36](#), [50](#), [54](#),
[60](#)

`weighting`, [7](#), [61](#)

`wrap_plots`, [51](#)