Package: specs (via r-universe)

September 7, 2024

Title Single-Equation Penalized Error-Correction Selector (SPECS)

Version 1.0.1

Description Implementation of SPECS, your favourite Single-Equation Penalized Error-Correction Selector developed in Smeekes and Wijler (2021) <doi:10.1016/j.jeconom.2020.07.021>. SPECS provides a fully automated estimation procedure for large and potentially (co)integrated datasets. The dataset in levels is converted to a conditional error-correction model, either by the user or by means of the functions included in this package, and various specialised forms of penalized regression can be applied to the model. Automated options for initializing and selecting a sequence of penalties, as well as the construction of penalty weights via an initial estimator, are available. Moreover, the user may choose from a number of pre-specified deterministic configurations to further simplify the model building process.

Depends R (>= 3.5.0)

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

LinkingTo Rcpp, RcppArmadillo

Imports Rcpp

NeedsCompilation yes

Author Etienne Wijler [aut, cre], Stephan Smeekes [aut] (<https://orcid.org/0000-0002-0157-639X>)

Maintainer Etienne Wijler <e.j.j.wijler@vu.nl>

Date/Publication 2024-02-29 21:10:02 UTC

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

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

RemoteRef HEAD

RemoteSha 398c6d77883ea57ce4edfcf03f07d1959b0c1379

Contents

specs *SPECS*

Description

This function estimates the Single-equation Penalized Error Correction Selector as described in Smeekes and Wijler (2020). The function takes a dependent variable y and a matrix of independent variables x as input, and transforms it to a conditional error correction model. This model is estimated by means of penalized regression, involving $L1$-penalty on individual coefficients and a potential $L2$-penalty on the coefficients of the lagged levels in the model, see Smeekes and Wijler (2020) for details.

Usage

```
specs(
  y,
  x,
  p = 1,
  deterministics = c("constant", "trend", "both", "none"),
  ADL = FALSE,
  weights = c("ridge", "ols", "none"),
  k_delta = 1,
  k_pi = 1,
  lambda_g = NULL,
  lambda_i = NULL,
  thresh = 1e-04,
  max_iter_delta = 1e+05,
  max_iter_pi = 1e+05,
  max_iter_gamma = 1e+05
)
```

Arguments

y A vector containing the dependent variable in levels.

x A matrix containing the independent variables in levels.

p Integer indicating the desired number of lagged differences to include. Default is 1.

deterministics A character object indicating which deterministic variables should be added ("none","constant","trend","both"). Default is "constant".

ADL Logical object indicating whether an ADL model without error-correction term should be estimated. Default is FALSE.

weights Choice of penalty weights. The weights can be automatically generated by ridge regression (default) or ols. Alternatively, a conformable vector of non-negative weights can be supplied or no weights can be applied.

k_delta The power to which the weights for delta should be raised, if weights are set to "ridge" or "ols".

k_pi The power to which the weights for pi should be raised, if weights are set to "ridge" or "ols".

lambda_g An optional user-specified grid for the group penalty may be supplied. If left empty, a 10-dimensional grid containing 0 as the minimum value is generated.

lambda_i An optional user-specified grid for the individual penalty may be supplied. If left empty, a 10-dimensional grid containing 0 as the minimum value is generated.

thresh The treshold for convergence.

max_iter_delta Maximum number of updates for delta. Default is 10^5.

max_iter_pi Maximum number of updates for pi. Default is 10^5.

max_iter_gamma Maximum number of updates for gamma. Default is 10^5.

Details

The function can generate an automated sequence of penalty parameters and offers the option to compute and include adaptive penalty weights. In addition, it is possible to estimate a penalized ADL model in differences by excluding the lagged levels from the model. For automated selection of an optimal penalty value, see the function specs_opt(...).

Value

D A matrix containing the deterministic variables included in the model.

gammas A matrix containing the estimated coefficients of the stochastic variables in the conditional error-correction model.

lambda_g The grid of group penalties.

lambda_i The grid of individual penalties.

Mv A matrix containing the independent variables, after regressing out the deterministic components.

My_d A vector containing the dependent variable, after regressing out the deterministic components.

theta The estimated coefficients for the constant and trend. If a deterministic component is excluded, its coefficient is set to zero.

v A matrix containing the independent variables (excluding deterministic components).

weights The vector of penalty weights.

y_d A vector containing the dependent variable, i.e. the differences of y.

Examples

```
#Estimate a model for unemployment and ten google trends

#Organize data
y <- Unempl_GT[,1]
index_GT <- sample(c(2:ncol(Unempl_GT)),10)
x <- Unempl_GT[,index_GT]

#Estimate a CECM with 1 lagged differences
my_specs <- specs(y,x,p=1)

#Estimate a CECM with 1 lagged differences and no group penalty
my_specs2 <- specs(y,x,p=1,lambda_g=0)

#Estimate an autoregressive distributed lag model with 2 lagged differences
my_specs3 <- specs(y,x,ADL=TRUE,p=2)
```

`specs_opt` *SPECS with data transformation and penalty optimization*

Description

This function estimates SPECS and selects the optimal penalty parameter based on a selection rule. All arguments correspond to those of the function specs(...), but it contains the additional arguments rule and CV_cutoff. Selection of the penalty parameter can be carried out by BIC or AIC or by time series cross-validation (TSCV). The degrees of freedom for the information criteria (BIC or AIC) are approximated by the number of non-zero coefficients in the estimated model. TSCV cuts the sample in two, based on the argument CV_cutoff which determines the proportion of the training sample. SPECS is estimated on the first part and the estimated model is used to predict the values in the second part. The selection is then based on the lowest Mean-Squared Forecast Error (MSFE) obtained over the test sample.

Usage

```
specs_opt(
  y,
  x,
  p = 1,
  rule = c("BIC", "AIC", "TSCV"),
  CV_cutoff = 2/3,
  deterministics = c("constant", "trend", "both", "none"),
  ADL = FALSE,
  weights = c("ridge", "ols", "none"),
  k_delta = 1,
  k_pi = 1,
  lambda_g = NULL,
  lambda_i = NULL,
  thresh = 1e-04,
  max_iter_delta = 1e+05,
  max_iter_pi = 1e+05,
  max_iter_gamma = 1e+05
)
```

Arguments

`y` A vector containing the dependent variable in levels.

`x` A matrix containing the independent variables in levels.

`p` Integer indicating the desired number of lagged differences to include. Default is 1.

`rule` A charcater object indicating which selection rule the optimal choice of the penalty parameters is based on. Default is "BIC".

`CV_cutoff` A numeric value between 0 and 1 that decides the proportion of the training sample as a fraction of the complete sample. Applies only when rule="TSCV". Default is 2/3.

`deterministics` A character object indicating which deterministic variables should be added ("none","constant","trend","both"). Default is "constant".

`ADL` Logical object indicating whether an ADL model without error-correction term should be estimated. Default is FALSE.

`weights` Choice of penalty weights. The weights can be automatically generated by ridge regression (default) or ols. Alternatively, a conformable vector of non-negative weights can be supplied.

`k_delta` The power to which the weights for delta should be raised, if weights are set to "ridge" or "ols".

`k_pi` The power to which the weights for pi should be raised, if weights are set to "ridge" or "ols".

`lambda_g` An optional user-specified grid for the group penalty may be supplied. If left empty, a 10-dimensional grid containing 0 as the minimum value is generated.

`lambda_i` An optional user-specified grid for the individual penalty may be supplied. If left empty, a 10-dimensional grid containing 0 as the minimum value is generated.

`thresh` The treshold for convergence.

`max_iter_delta` Maximum number of updates for delta. Default is 10^5.

`max_iter_pi` Maximum number of updates for pi. Default is 10^5.

`max_iter_gamma` Maximum number of updates for gamma. Default is 10^5.

Value

`D` A matrix containing the deterministic variables included in the model.

`gammas` A matrix containing the estimated coefficients of the stochastic variables in the conditional error-correction model.

`gamma_opt` A vector containing the estimated coefficients corresponding to the optimal model.

`lambda_g` The grid of group penalties.

`lambda_i` The grid of individual penalties.

`Mv` A matrix containing the independent variables, after regressing out the deterministic components.

`My_d` A vector containing the dependent variable, after regressing out the deterministic components.

`theta` The estimated coefficients for the constant and trend. If a deterministic component is excluded, its coefficient is set to zero.

`theta_opt` The estimated coefficients for the constant and trend in the optimal model.

`v` A matrix containing the independent variables (excluding deterministic components).

`weights` The vector of penalty weights.

`y_d` A vector containing the dependent variable, i.e. the differences of y.

Examples

```
#Estimate an automatically optimized model for unemployment and ten google trends

#Organize data
y <- Unempl_GT[,1]
index_GT <- sample(c(2:ncol(Unempl_GT)),10)
x <- Unempl_GT[,index_GT]

#Estimate a CECM with 1 lagged difference and penalty chosen by the minimum BIC
my_specs <- specs_opt(y,x,p=1,rule="BIC")
coefs <- my_specs$gamma_opt
```

specs_tr *SPECS on pre-transformed data*

Description

This function computes the Single-equation Penalized Error Correction Selector as described in Smeekes and Wijler (2020) based on data that is already in the form of a conditional error-correction model.

Usage

```
specs_tr(
  y_d,
  z_l = NULL,
  w,
  deterministics = c("constant", "trend", "both", "none"),
  ADL = FALSE,
  weights = c("ridge", "ols", "none"),
  k_delta = 1,
  k_pi = 1,
  lambda_g = NULL,
  lambda_i = NULL,
  thresh = 1e-04,
  max_iter_delta = 1e+05,
  max_iter_pi = 1e+05,
  max_iter_gamma = 1e+05
)
```

Arguments

`y_d` A vector containing the differences of the dependent variable.

`z_l` A matrix containing the lagged levels.

`w` A matrix containing the required difference

`deterministics` Indicates which deterministic variables should be added (0 = none, 1=constant, 2=constant and linear trend).

`ADL` Boolean indicating whether an ADL model without error-correction term should be estimated. Default is FALSE.

`weights` Choice of penalty weights. The weights can be automatically generated by ridge regression (default) or ols. Alternatively, a conformable vector of non-negative weights can be supplied.

`k_delta` The power to which the weights for delta should be raised, if weights are set to "ridge" or "ols".

`k_pi` The power to which the weights for pi should be raised, if weights are set to "ridge" or "ols".

`lambda_g` An optional user-specified grid for the group penalty may be supplied. If left empty, a 10-dimensional grid containing 0 as the minimum value is generated.

`lambda_i` An optional user-specified grid for the individual penalty may be supplied. If left empty, a 10-dimensional grid containing 0 as the minimum value is generated.

`thresh` The treshold for convergence.

`max_iter_delta` Maximum number of updates for delta. Defaults is 1e5.

`max_iter_pi` Maximum number of updates for pi. Defaults is 1e5.

`max_iter_gamma` Maximum number of updates for gamma. Defaults is 1e5.

Value

`D` A matrix containing the deterministic variables included in the model.

`gammas` A matrix containing the estimated coefficients of the stochastic variables in the conditional error-correction model.

`gamma_opt` A vector containing the estimated coefficients corresponding to the optimal model.

`lambda_g` The grid of group penalties.

`lambda_i` The grid of individual penalties.

`theta` The estimated coefficients for the constant and trend. If a deterministic component is excluded, its coefficient is set to zero.

`theta_opt` The estimated coefficients for the constant and trend in the optimal model.

`weights` The vector of penalty weights.

Examples

```
#Estimate a conditional error-correction model on pre-transformed data with a constant

#Organize data
y <- Unempl_GT[,1]
index_GT <- sample(c(2:ncol(Unempl_GT)),10)
x <- Unempl_GT[,index_GT]
y_d <- y[-1]-y[-100]
z_l <- cbind(y[-100],x[-100,])
w <- x[-1,]-x[-100,] #This w corresponds to a cecm with p=0 lagged differences

my_specs <- specs_tr(y_d,z_l,w,deterministics="constant")

#Estimate an ADL model on pre-transformed data with a constant

my_specs <- specs_tr(y_d,NULL,w,ADL=TRUE,deterministics="constant")
```

`specs_tr_opt` *SPECS with data transformation and penalty optimization*

Description

The same function as specs_tr(...), but on data that is pre-transformed to a CECM.

Usage

```
specs_tr_opt(
  y_d,
  z_l = NULL,
  w,
  rule = c("BIC", "AIC", "TSCV"),
  CV_cutoff = 2/3,
  deterministics = c("constant", "trend", "both", "none"),
  ADL = FALSE,
  weights = c("ridge", "ols", "none"),
  k_delta = 1,
  k_pi = 1,
  lambda_g = NULL,
  lambda_i = NULL,
  thresh = 1e-04,
  max_iter_delta = 1e+05,
  max_iter_pi = 1e+05,
  max_iter_gamma = 1e+05
)
```

Arguments

`y_d` A vector containing the differences of the dependent variable.

`z_l` A matrix containing the lagged levels.

`w` A matrix containing the required difference.

`rule` A charcater object indicating which selection rule the optimal choice of the penalty parameters is based on. Default is "BIC".

`CV_cutoff` A numeric value between 0 and 1 that decides the proportion of the training sample as a fraction of the complete sample. Applies only when rule="TSCV". Default is 2/3.

`deterministics` A character object indicating which deterministic variables should be added ("none","constant","trend","both"). Default is "constant".

`ADL` Logical object indicating whether an ADL model without error-correction term should be estimated. Default is FALSE.

`weights` Choice of penalty weights. The weights can be automatically generated by ridge regression (default) or ols. Alternatively, a conformable vector of non-negative weights can be supplied.

`k_delta` The power to which the weights for delta should be raised, if weights are set to "ridge" or "ols".

`k_pi` The power to which the weights for pi should be raised, if weights are set to "ridge" or "ols".

`lambda_g` An optional user-specified grid for the group penalty may be supplied. If left empty, a 10-dimensional grid containing 0 as the minimum value is generated.

`lambda_i` An optional user-specified grid for the individual penalty may be supplied. If left empty, a 10-dimensional grid containing 0 as the minimum value is generated.

`thresh` The treshold for convergence.

`max_iter_delta` Maximum number of updates for delta. Default is 10^5.

`max_iter_pi` Maximum number of updates for pi. Default is 10^5.

`max_iter_gamma` Maximum number of updates for gamma. Default is 10^5.

Value

`D` A matrix containing the deterministic variables included in the model.

`gammas` A matrix containing the estimated coefficients of the stochastic variables in the conditional error-correction model.

`gamma_opt` A vector containing the estimated coefficients corresponding to the optimal model.

`lambda_g` The grid of group penalties.

`lambda_i` The grid of individual penalties.

theta The estimated coefficients for the constant and trend. If a deterministic component is excluded, its coefficient is set to zero.

theta_opt The estimated coefficients for the constant and trend in the optimal model.

v A matrix containing the independent variables (excluding deterministic components).

weights The vector of penalty weights.

y_d A vector containing the dependent variable, i.e. the differences of y.

Examples

```
#Estimate a CECM with a constant, ols initial weights and penalty chosen by the minimum AIC

#Organize data
y <- Unempl_GT[,1]
index_GT <- sample(c(2:ncol(Unempl_GT)),10)
x <- Unempl_GT[,index_GT]
y_d <- y[-1]-y[-100]
z_l <- cbind(y[-100],x[-100,])
w <- x[-1,]-x[-100,] #This w corresponds to a cecm with p=0 lagged differences

my_specs <- specs_tr_opt(y_d,z_l,w,rule="AIC",weights="ols",deterministics="constant")
```

Unempl_GT *Unemployment and Google Trends Data*

Description

Time series data on Dutch unemployment from Statistics Netherlands, and Google Trends popularity index for search terms related to unemployment. The Google Trends data can be used to nowcast unemployment.

Usage

```
Unempl_GT
```

Format

A time series object where the first column contains monthly total unemployment in the Netherlands (x1000, seasonally unadjusted), and the remaining 87 columns are monthly Google Trends series with popularity of Dutch search terms related to unemployment.

Source

CBS StatLine, https://opendata.cbs.nl/statline, and Google Trends, https://www.google.nl/trends

Index