Package 'CESKalman'

January 28, 2021

Title CESKalman

Version 1.0

Description This package can be used to estimate the elasticity of substitution in a CES function with two inputs. The package uses the Kalman filter to account for unobserved structural changes, such as technical change. The main function is CESKalman.

License What license it uses

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

R topics documented:

build_SS 1
CESKalman 3
CESKalman_Bootstrap 5
CESKalman_Estimation 6
Load_Data 8
plot.CESKalman 9

Index **11**

`build_SS` *Function for building a state space representation to be used in the CESKalman function*

Description

This function builds a state space representation to be used in the CESKalman_Estimation, CESKalman, and CESKalman_Bootstrap functions.

Usage

```
build_SS(param,X,nlags,lambda,Leontief=FALSE,sigma_init,alpha_init)
```

Arguments

param	Parameter values of the observation and state variance (see details)
X	A matrix of all explanatory variables to be used (See details)
nlags	Number of lags used of relative prices and relative budget shares. One of 0,1,2.
lambda	The inverse of the signal-to-noise ratio. Can be any positive value larger than zero. If set to NA it is estimated
Leontief	TRUE or FALSE (default). Should the elasticity of substitution be fixed at zero? If so, set =TRUE
sigma_init	Initial value for the long run elasticity of substitution, σ
alpha_init	Initial value for the adjustment parameter in the ECM, α

Details

This function builds a state space representation to be used in the CESKalman_Estimation, CESKalman, and CESKalman_Bootstrap functions. It is in particular suitable for production functions with two different production factors where one of the factors is persistent, but can be used in any CES function with two factors.

The estimated function is the Error-Correction model from Kronborg et al (2019): $\Delta s_t = \alpha(s_{t-1} - (1-\sigma)p_{t-1} - \mu_{t-1}) + \sum_{i=0}^{nlags} \kappa_i \Delta p_{t-i} + \sum_{i=1}^{nlags} \gamma_i \Delta s_{t-i} + \epsilon_t$. s_t is the relative budget shares in logs (expenditure on factor 1 relative to factor 2), p_t is the relative prices in logs and $\mu_t = (\sigma - 1)log(\Gamma_t)$ where Γ_t is the relative augmenting technical change. The process of μ_t, the state variable, is an I(2) process: $\Delta\mu_t = \Delta\mu_{t-1} + \eta_t$.

param can be a string of length 1 or 2. If lambda is freely estimated (set to NA) both param[1] (variance of error term in observation equation, ϵ_t) and param[2] (variance of the error term in the relative augmenting technologies, η_t) are used. Else, only param[1] is used and $\lambda\Sigma^\eta = \Sigma^\epsilon$. Note that the variances are specified such that $\Sigma^\epsilon = exp(param[1])$ and $\Sigma^\eta = exp(param[2])$ to ensure that the variances are positive. We refer to Kronborg et al (2019) for further description of the methodology.

X should be a matrix with all the explanatory variables to be used (all in logs). The dimension is TxK, where T is time periods and K are the explanatory variables (not counting technical change as one). K=3 when nlags=0, =5 when nlags=1 and =7 when nlags=2. The column ordering should be: 1: lags of relative budget shares in log-levels. 2: lags of the relative prices in log-levels. 3: First-difference of log relative prices. 4: First-differences of log relative prices lagged one period. 5: First-differences of log relative budget shares lagged one period. 6: First-differences of log relative prices lagged two periods. 7: First-differences of log relative budget shares lagged two periods. When using the CESKalman function, data will be ordered in this way automatically.

Value

A list of class 'dlm' with all vectors and matrices to be used in the functions dlmMLE and dlmSmooth. We refer to Petris et al (2010) for further description.

Author(s)

Christian Sandholm Kastrup <CST@dreamgruppen.dk> , Anders Farver Kronborg <ANK@dreamgruppen.dk> and Peter Philip Stephensen <PSP@dreamgruppen.dk>

References

Kronborg et al (2019) and Petris et al (2010)

CESKalman *Function for estimating a CES production function with the Kalman filter.*

Description

The function applies the CESKalman_Estimation function to estimate the elasticity of substitution in a CES production function using the Kalman filter. CESKalman is more robust than the CESKalman_Estimation as it loops over different values of the signal-to-noise ratio, the number of lags, and initial parameter values. The combination that maximizes the likelihood given the model is well specified based on the bgtest from lmtest package and the Normalized Innovations Squared is chosen.

Usage

```
CESKalman(data,grid.param_init=c(-9,-1,5),max_nlags=2,grid.lambda,
                grid.alpha_init=c(-0.9,-0.1,0.3),grid.sigma_init=c(0,1.5,0.5),
          cVal_NIS=0.10,cVal_Auto=0.10,print_results=TRUE,lambda_est_freely=TRUE)
```

Arguments

data	A matrix or ts matrix with the data series (see details)
grid.param_init	A string of length 3. First and second argument are the, respectively, lower and upper bound of the grid of the initial variance of the error term used in the dlmMLE function. Third is the step size
max_nlags	Maximum number of lags used. One of 0,1,2
grid.lambda	A string of length 3. First and second argument are the, respectively, lower and upper bound of the grid of the inverse of the signal-to-noise ratio. Third is the step size. If set to NA, no grid is used
grid.alpha_init	A string of length 3. First and second argument are the lower and upper bound of initial values for the adjustment parameter, α, to loop over. Third is the step size
grid.sigma_init	A string of length 3. First and second argument are the lower and upper bound of initial values for the long run elasticity of substitution, σ, to loop over. Third is the step size
cVal_NIS	Critical value to be used to construct the confidence bands of the NIS test
cVal_Auto	Critical value to be used in the Breusch Godfrey test for autocorrelation
print_results	Do you want the function to print results while estimating?
lambda_est_freely	Should lambda also be estimated freely?

Details

This function calls the function CESKalman_Estimation multiple times as it loops over a grid of values of the signal-to-noise ratio, number of lags chosen to prevent autocorrelation and different combinations of initial parameter values of sigma and alpha. The optimal signal-to-noise ratio and

initial parameter values are chosen to maximize the likelihood, conditional on the model being well specified based on a Breusch Godfrey test for autocorrelation and a Normialized Innovations Squared test for filter misspecification.

The estimated function is the Error-Correction model from Kronborg et al (2019): $\Delta s_t = \alpha(s_{t-1} - (1-\sigma)p_{t-1} - \mu_{t-1}) + \sum_{i=0}^{nlags} \kappa_i \Delta p_{t-i} + \sum_{i=1}^{nlags} \gamma_i \Delta s_{t-i} + \epsilon_t$. s_t is the relative budget shares in logs (expenditure on factor 1 relative to factor 2), p_t is the relative prices in logs and $\mu_t = (\sigma - 1) log(\Gamma_t)$ where Γ_t is the relative augmenting technical change. The process of μ_t, the state variable, is an I(2) process: $\Delta \mu_t = \Delta \mu_{t-1} + \eta_t$.

data should be a matrix or ts matrix containing the data series and the dimension Tx4. First and second column is the price of the first and second factor, respectively. Third and forth is the quantity of factor 1 and factor 2, respectively.

The function loops over a grid of different signal-to-noise ratios in the range specified in grid.lambda. In addition, it is possible to also estimate this parameter freely by setting lambda_est_freely=TRUE. In this case, the preferred value from the grid search is compared to the free estimation and the one that maximizes likelihood given the estimation being well specified is chosen.

If sigma is estimated to be negative it is restricted to zero, but we still allow for short run fluctuations of prices to influence the budget shares.

Potential convergence errors of dlmMLE can most often be resolved by increasing the range of grid.param_init, e.g. to grid.param_init=c(-9,-1,1). Note that the variances are specified such that $\Sigma^{\epsilon} = exp(param[1])$ and $\Sigma^{\eta} = exp(param[2])$ if lambda is freely estimated. Else, only exp(param[1]) is used and $\lambda \Sigma^{\eta} = \Sigma^{\epsilon}$. When lambda is freely estimated the grid.param_init is still only over values of exp(param[1]), i.e. the observation variance. But for any observation variance we try five different values of lambda in the interval 10-1000 as initial value of exp(param[2]).

Value

Returns a list of class CESKalman from the preferred call of function CESKalman_Estimation (highest likelihood and well specified). See CESKalman_Estimation for description.

Author(s)

Christian Sandholm Kastrup <CST@dreamgruppen.dk> , Anders Farver Kronborg <ANK@dreamgruppen.dk> and Peter Philip Stephensen <psp@dreamgruppen.dk>

References

Kronborg et al (2019)

Examples

```
## First, data is loaded with the Load_Data function (or any other data set)
data = Load_Data(Country="USA",tstart=1970,tend=2017)

data = data.frame(data)

data = cbind(data$q,data$w,data$K,data$L)

## We can then estimate with four different approaches that all depends on the value of lambda.

# 1. Grid combined with free estimation (our preffered method)
Kalman = CESKalman(data=data,grid.lambda=c(10,500,20), lambda_est_freely=TRUE)
```

```
# 2. Grid only
Kalman = CESKalman(data=data,grid.lambda=c(10,500,20), lambda_est_freely=FALSE)

# 3. Free estimation only
Kalman = CESKalman(data=data,grid.lambda=NA, lambda_est_freely=TRUE)

# 4. A fixed value of lambda, e.g. lambda=100
Kalman = CESKalman(data=data,grid.lambda=c(100,100,100), lambda_est_freely=FALSE) ## Only argument [1] used

## Output is displayed with e.g. the plot function:
plot(Kalman)

sigma=Kalman$sigma # This is the elasticity
alpha=Kalman$alpha # This is the adjustment parameter
Gamma=Kalman$Gamma # This is the relative log augmenting technologies
```

`CESKalman_Bootstrap` *Function for bootstrapping confidence intervals of the elasticity and adjustment parameter*

Description

The function uses a recursive-design bootstrapping procedure to produce confidence intervals of the elasticity of substitution and adjustment parameter. The CESKalman_Estimation function is applied and estimated for every new draw.

Usage

```
CESKalman_Bootstrap(Estimation,grid.param_init=c(-9,-1,3),ndraw=1000,print_results=TRUE)
```

Arguments

`Estimation`	An object of class CESKalman
`grid.param_init`	A vector of initial Parameter values of the observation variance to loop over (see details)
`ndraw`	The number of draws in the bootstrapping procedure, 1000 is default.
`print_results`	Should results be printed while bootstrapping?

Details

This function takes an object of class CESKalman as input and calls the function CESKalman_Estimation for every new draw. The parameter estimates from CESKalman are used as initial values of sigma and alpha. Also, the number of lags and lambda are determined by the CESKalman function.

grid.param_init can be a vector of any length. Values to loop over for the variance of error term in observation equation, ϵ_t, in the numerical optimization. When lambda is FALSE, differen values for the signal-to-noise ratio in the range from 10-1000 found to be optimal in Kronborg et al (2019) are tried. The one that maximizes the likelihood is chosen.

Value

Returns a matrix of dimension ndrawX3. First column is the draws of sigma, second is the draws of alpha and third is the likelihood value.

Author(s)

Christian Sandholm Kastrup <CST@dreamgruppen.dk>, Anders Farver Kronborg <ANK@dreamgruppen.dk> and Peter Philip Stephensen <PSP@dreamgruppen.dk>

References

Kronborg et al (2019) and Petris et al (2010)

Examples

```
## First, data is loaded with the Load_Data function (or any other data set)
data = Load_Data(Country="USA",tstart=1970,tend=2017)

data = data.frame(data)

data = cbind(data$q,data$w,data$K,data$L)

## Next, the CESKalman function is called
Kalman = CESKalman(data=data,grid.lambda=c(10,500,20), lambda_est_freely=TRUE)

## Bootstrapping confidence intervals
Bootstrap = CESKalman_Bootstrap(Estimation=Kalman)
```

CESKalman_Estimation *Function for estimating a CES production function with the Kalman filter.*

Description

This function first estimates the variances with the dlmMLE function from dlm package. Second, the dlmSmooth function is applied to estimate the state parameters. Third, several statistical tests are performed using the bgtest and bptest from lmtest package and the jb.norm.test from package normtest. Also, it calculates the normalized innovations squared.

Usage

```
CESKalman_Estimation(Y,data,grid.param_init=c(-9,-1,1),nlags,lambda,

Leontief=FALSE,alpha_init,sigma_init,cVal_NIS=0.10)
```

Arguments

Y	First-difference of log relative budget shares
X	A matrix of all explanatory variables to be used (See details)
grid.param_init	A string of length 3. First and second argument are the, respectively, lower and upper bound of the grid of the initial variance of the error term used in the dlmMLE function. Third is the step size
nlags	Number of lags used of relative prices and relative budget shares. One of 0,1,2
lambda	The inverse of the signal-to-noise ratio. Can be any positive value. If set to FALSE (the default) it is estimated
Leontief	TRUE or FALSE (default). Should the elasticity of substitution be fixed at zero? If so, set =TRUE
alpha_init	Initial value for the adjustment parameter, α
sigma_init	Initial value for the long run elasticity of substitution, σ
cVal_NIS	Critical value used to construct the confidence bands of the NIS test. Default is 0.10

Details

This function first estimates the variances with the dlmMLE function from dlm package. Second, the dlmSmooth function is applied to estimate the state parameters. Third, several statistical tests are performed using the bgtest and bptest from lmtest package and the jaruqe.bera.test from package normtest. Also, it calculates the normalized innovations squared.

NOTE: This function can be used on its own, but we encourage to use the CESKalman instead as it is more robust.

The estimated function is the Error-Correction model from Kronborg et al (2019): $\Delta s_t = \alpha(s_{t-1} - (1-\sigma)p_{t-1} - \mu_{t-1}) + \sum_{i=0}^{nlags} \kappa_i \Delta p_{t-i} + \sum_{i=1}^{nlags} \gamma_i \Delta s_{t-i} + \epsilon_t$. s_t is the relative budget shares in logs (expenditure on factor 1 relative to factor 2), p_t is the relative prices in logs and $\mu_t = (\sigma - 1)log(\Gamma_t)$ where Γ_t is the relative augmenting technical change. The process of μ_t, the state variable, is an I(2) process: $\Delta\mu_t = \Delta\mu_{t-1} + \eta_t$.

X should be a matrix with all the explanatory variables to be used (all in logs). The dimension is TxK, where T is time periods and K are the explanatory variables (not counting technical change as one). K=3 when nlags=0, =5 when nlags=1 and =7 when nlags=2. The column ordering should be: 1: lags of relative budget shares in log-levels. 2: lags of the relative prices in log-levels. 3: First-difference of log relative prices. 4: First-differences of log relative prices lagged one period. 5: First-differences of log relative budget shares lagged one period. 6: First-differences of log relative prices lagged two periods. 7: First-differences of log relative budget shares lagged two periods. When using the CESKalman, CESKalman_Estimation or CESKalman_Bootstrap functions, data will be ordered in this way automatically.

grid.param_init can be a vector of any length. It is the values to loop over for the variance of the error term in observation equation, ϵ_t, in the numerical optimization. When lambda is set to NA, different values for the signal-to-noise ratio in the range from 10-1000 found to be optimal in Kronborg et al (2019) are tried. The one that maximizes the likelihood is chosen.

Value

Returns a list with the following objects:

Smooth: The list returned from the dlmSmooth function. **Gamma:** The relative augmenting technologies in logarithms (input 1 relative to input 2). **residuals:** The residuals from the observation

equation. **sigma:** The long run elasticity of substitution, σ. **alpha:** The adjustment parameter, α. **LV:** The likelihood value. **est.lambda:** The estimated value of lambda. (only estimated if lambda was specified as FALSE, else just the chosen value). **BG_test:** p-value from the Breusch Godfrey test for autocorrelation from function bgtest. **BP_test:** p-value from the Breusch Pagan test for heteroscedasticity from function bptest. **JB_test:** p-value from the Jarque Bera test for normality from function jbtest. **NIS_test:** Value, lower and upper confidence bands of the Normalized Innovations Squared test. **data:** The data applied in estimation.

Author(s)

Christian Sandholm Kastrup <CST@dreamgruppen.dk> , Anders Farver Kronborg <ANK@dreamgruppen.dk> and Peter Philip Stephensen <psp@dreamgruppen.dk>

See Also

See dlm package for a decription of dlm objects, in particular the dlmSmooth function. See Kronborg et al (2019) for a description of the methodology.

`Load_Data` *Data series for 16 OECD countries obtained from the PWT and OECD database*

Description

This function loads data from the PWT and OECD databases and is suitable for estimating the elasticity of substitution between capital and labor at the country level.

Usage

```
Load_Data(Country,tstart=1970,tend=2017)
```

Arguments

`Country`	The ISO code of the country (see details)
`tstart`	Initial time period. Earliest is 1970 (default)
`tend`	Last time period. Latest possible is 2017 (default)

Details

This function loads various types of data and returns a time series matrix for the chosen country. **Note:** We do not take responsibility for potential errors and the data is primarily included for illustration.

The data series are obtained from the PWT version 9.1 and the OECD data base.

Available countries are Australia (AUS), Austria (AUT), Belgium (BEL), Canada (CAN), Denmark (DNK), Finland (FIN) France (FRA), Germany (DEU), Italy (ITA), Japan (JPN), the Netherlands (NLD), New Zealand (NZL), Norway (NOR), Sweden (SWE), Great Britain (GBR) and the United States (USA).

The user cost of capital is calculated as: $q_t = (v_{it}/q_{it}) * (irr_t + delta_t)$ with i denoting investments

Value

Returns a time series matrix with the objects: **q_gdp:** Real GDP in national currency. **v_gdp:** Nominal GDP in national currency. **emp:** Employment in national currency. **avh:** Average number of yearly working hours. **labsh:** Labor share in nominal GDP. **irr:** Real Internal Rate of Return (see PWT version 9.1). **delta:** Residually calculated depreciation rate. **v_i:** Nominal investments in national currency. **q_i:** Real investments in national currency. **K:** Net capital stock. **L:** Number of yearly labor hours. **w:** The wage. **pi:** Inflation rate. **q:** A simplified Hall-Jorgenson user cost (see details). **KL:** Capital/labor ratio. **markup:** The calculated profits rate. **GDP_US:** Real GDP in USD. Can be used in weighted regressions.

Author(s)

Christian Sandholm Kastrup <CST@dreamgruppen.dk>, Anders Farver Kronborg <ANK@dreamgruppen.dk> and Peter Philip Stephensen <PSP@dreamgruppen.dk>

References

Feenstra et al (2015)

See Also

https://stats.oecd.org/Index.aspx?DataSetCode=STAN and https://www.rug.nl/ggdc/productivity/pwt/

Examples

```
Data = Load_Data(Country="USA",tstart=1970,tend=2017)

Data = data.frame(Data)

## Create the data object needed in the CESKalman function:
data = cbind(Data$q,Data$w,Data$K,Data$L)
```

---

plot.CESKalman *Function for plotting the model fit and trend/price decomposition of a CESKalman object*

Description

The first plot contains the model fit of the relative quantities. The second is a decomposition of the fitted values in a price and a trend component. The third plot is the demeaned-data series applied in estimation.

Usage

```
plot.CESKalman(Kalman,t0=1,tEnd=nrow(Kalman$data))
```

Arguments

Kalman	an object of class CESKalman returned from the CESKalman function
t0	Start date. Only available for yearly data, else it is an index
tEnd	End date. Only available for yearly data, else it is an index

Details

NOTE: As we are primarily interested to explain the model fit of the relative quantities, the estimated equation is first translated into quantities, denoted x_t: $\Delta x_t = \alpha(x_{t-1} + \sigma p_{t-1} - \mu_{t-1}) + (\kappa_i - 1)\Delta p_t + \epsilon_t$. Note that nlags=0 is currently only available option.

This constitutes a differential equation. The fitted values can be decomposed into two terms, $\hat{x}_t = \hat{x}_t^{Trend} + \hat{x}_t^{Price}$. The trend component is $\hat{x}_t^{Trend} \equiv \hat{x}_0(1+\hat{\alpha})^t - \hat{\alpha}\sum_{s=1}^{t}\hat{\mu}_{s-1} * (1+\hat{\alpha})^{t-s}$. The price component is $\hat{x}_t^{Price} \equiv \sum_{s=1}^{t}(\hat{\alpha}\hat{\sigma}p_{s-1} + (\hat{\kappa}-1)\Delta p_s)(1+\hat{\alpha})^{t-s}$ For now, the function can only handle yearly data, if you want to specify the start and end date. If using data at another frequency, you will have to use the default t0 and tEnd. In addition, the function is only suitable when nlags=0.

Author(s)

Christian Sandholm Kastrup <CST@dreamgruppen.dk>, Anders Farver Kronborg <ANK@dreamgruppen.dk> and Peter Philip Stephensen <PSP@dreamgruppen.dk>

References

Kronborg et al (2019)

Examples

```
## First, data is loaded with the Load_Data function (or any other data set)
data = Load_Data(Country="USA",tstart=1970,tend=2017)

data = data.frame(data)

data = cbind(data$q,data$w,data$K,data$L)

## The second step is a call to the CESKalman function
Kalman = CESKalman(data,grid.lambda=c(10,1000,20),lambda_est_freely = F,max_nlags=0)

## Lastly we can plot the fit of the model:
plot(Kalman)
```

Index

build_SS, 1

CESKalman, 3
CESKalman_Bootstrap, 5
CESKalman_Estimation, 6

Load_Data, 8

plot.CESKalman, 9