

Package ‘micEconCES’

July 22, 2025

Version 1.0-2

Date 2022-12-23

Title Analysis with the Constant Elasticity of Substitution (CES)
Function

Author Arne Henningsen and Geraldine Henningsen

Maintainer Arne Henningsen <arne.henningsen@gmail.com>

Depends R (>= 2.4.0), minpack.lm (>= 1.1-4), DEoptim (>= 2.0-4), car
(>= 2.0-0)

Suggests maxLik (>= 0.8-0), xtable (>= 1.5-6), AER (>= 1.1-9)

Imports systemfit (>= 1.0-0), micEcon (>= 0.6-1), miscTools (>= 0.6-1)

Description Tools for econometric analysis and economic modelling
with the traditional two-input Constant Elasticity of Substitution (CES) function
and with nested CES functions with three and four inputs.
The econometric estimation can be done by the Kmenta approximation,
or non-linear least-squares
using various gradient-based or global optimisation algorithms.
Some of these algorithms can constrain the parameters to certain ranges,
e.g. economically meaningful values.
Furthermore, the non-linear least-squares estimation
can be combined with a grid-search for the rho-parameter(s).
The estimation methods are described in Henningsen et al. (2021)
<doi:10.4337/9781788976480.00030>.

License GPL (>= 2)

Encoding UTF-8

BugReports <https://github.com/micEcon/micEconCES>

URL <http://www.micEcon.org>

NeedsCompilation no

Repository CRAN

Date/Publication 2023-01-06 13:40:02 UTC

Contents

cesCalc	2
cesEst	4
cesEst-methods	10
durbinWatsonTest.cesEst	12
GermanIndustry	13
MishraCES	15
plot.cesEst	16
summary.cesEst	17

Index	19
--------------	-----------

cesCalc	<i>Calculate CES function</i>
---------	-------------------------------

Description

Calculate the endogenous variable of a ‘Constant Elasticity of Substitution’ (CES) function.

The original CES function with two explanatory variables is

$$y = \gamma \exp(\lambda t) (\delta x_1^{-\rho} + (1 - \delta) x_2^{-\rho})^{-\frac{\nu}{\rho}}$$

and the non-nested CES function with N explanatory variables is

$$y = \gamma \exp(\lambda t) \left(\sum_{i=1}^N \delta_i x_i^{-\rho} \right)^{-\frac{\nu}{\rho}}$$

where in the latter case $\sum_{i=1}^N \delta_i = 1$.

In both cases, the elasticity of substitution is $s = \frac{1}{1+\rho}$.

The *nested* CES function with 3 explanatory variables proposed by Sato (1967) is

$$y = \gamma \exp(\lambda t) \left[\delta (\delta_1 x_1^{-\rho_1} + (1 - \delta_1) x_2^{-\rho_1})^{\frac{\rho}{\rho_1}} + (1 - \delta) x_3^{-\rho} \right]^{-\frac{\nu}{\rho}}$$

and the *nested* CES function with 4 explanatory variables (a generalisation of the version proposed by Sato, 1967) is

$$y = \gamma \exp(\lambda t) \left[\delta \cdot (\delta_1 x_1^{-\rho_1} + (1 - \delta_1) x_2^{-\rho_1})^{\frac{\rho}{\rho_1}} + (1 - \delta) \cdot (\delta_2 x_3^{-\rho_2} + (1 - \delta_2) x_4^{-\rho_2})^{\frac{\rho}{\rho_2}} \right]^{-\frac{\nu}{\rho}}$$

Usage

```
cesCalc( xNames, data, coef, tName = NULL, nested = FALSE, rhoApprox = 5e-6 )
```

Arguments

xNames	a vector of strings containing the names of the explanatory variables.
data	data frame containing the explanatory variables.
coef	numeric vector containing the coefficients of the CES: if the vector is unnamed, the order of the coefficients must be γ , eventually λ , δ , ρ , and eventually ν in case of two explanatory variables, γ , eventually λ , $\delta_1, \dots, \delta_N$, ρ , and eventually ν in case of the non-nested CES with $N > 2$ explanatory variables, γ , eventually λ , δ_1 , δ , ρ_1 , ρ , and eventually ν in case of the nested CES with 3 explanatory variables, and γ , eventually λ , δ_1 , δ_2 , δ , ρ_1 , ρ_2 , ρ , and eventually ν in case of the nested CES with 4 explanatory variables, where in all cases the ν is only required if the model has variable returns to scale. If the vector is named, the names must be "gamma", "delta", "rho", and eventually "nu" in case of two explanatory variables, "gamma", "delta_1", ..., "delta_N", "rho", and eventually "nu" in case of the non-nested CES with $N > 2$ explanatory variables, and "gamma", "delta_1", "delta_2", "rho_1", "rho_2", "rho", and eventually "nu" in case of the nested CES with 4 explanatory variables, where the order is irrelevant in all cases.
tName	optional character string specifying the name of the time variable (t).
nested	logical. ; if FALSE (the default), the original CES for n inputs proposed by Kmenta (1967) is used; if TRUE, the nested version of the CES for 3 or 4 inputs proposed by Sato (1967) is used.
rhoApprox	if the absolute value of the coefficient ρ , ρ_1 , or ρ_2 is smaller than or equal to this argument, the endogenous variable is calculated using a first-order Taylor series approximation at the point $\rho = 0$ (for non-nested CES functions) or a linear interpolation (for nested CES functions), because this avoids large numerical inaccuracies that frequently occur in calculations with non-linear CES functions if ρ , ρ_1 , or ρ_2 have very small values (in absolute terms).

Value

A numeric vector with length equal to the number of rows of the data set specified in argument data.

Author(s)

Arne Henningsen and Geraldine Henningsen

References

- Kmenta, J. (1967): On Estimation of the CES Production Function. *International Economic Review* 8, p. 180-189.
- Sato, K. (1967): A Two-Level Constant-Elasticity-of-Substitution Production Function. *Review of Economic Studies* 43, p. 201-218.

See Also

[cesEst](#).

Examples

```
data( germanFarms, package = "micEcon" )
# output quantity:
germanFarms$qOutput <- germanFarms$vOutput / germanFarms$pOutput
# quantity of intermediate inputs
germanFarms$qVarInput <- germanFarms$vVarInput / germanFarms$pVarInput

## Estimate CES: Land & Labor with fixed returns to scale
cesLandLabor <- cesEst( "qOutput", c( "land", "qLabor" ), germanFarms )

## Calculate fitted values
cesCalc( c( "land", "qLabor" ), germanFarms, coef( cesLandLabor ) )

# variable returns to scale
cesLandLaborVrs <- cesEst( "qOutput", c( "land", "qLabor" ), germanFarms,
  vrs = TRUE )

## Calculate fitted values
cesCalc( c( "land", "qLabor" ), germanFarms, coef( cesLandLaborVrs ) )
```

cesEst

Estimate a CES function

Description

Estimate a Constant-Elasticity-of-Substitution (CES) function with two exogenous variables or a nested Constant-Elasticity-of-Substitution (CES) function proposed by Sato (1967) with three or four exogenous variables by Least Squares. The functional forms are shown in the documentation of function [cesCalc](#).

Warning: The econometric estimation of a CES function is (almost) always very problematic, because very different parameter vectors could result in very similar values of the objective function (sum of squared residuals). Hence, even if the optimizer reports that the nonlinear minimization has converged, there might be another rather different parameter vector that results in a lower sum of squared residuals.

Usage

```
cesEst( yName, xNames, data, tName = NULL, vrs = FALSE, method = "LM",
  start = NULL, lower = NULL, upper = NULL, multErr = FALSE,
  rho1 = NULL, rho2, rho = NULL, returnGridAll = FALSE,
  returnGrad = FALSE, random.seed = 123,
  rhoApprox = c( y = 5e-6, gamma = 5e-6, delta = 5e-6,
    rho = 1e-3, nu = 5e-6 ),
  checkStart = TRUE, ... )

## S3 method for class 'cesEst'
```

```
print( x, digits = max(3, getOption("digits") - 3),
      ... )
```

Arguments

yName	a string containing the name of the dependent variable.
xNames	a vector of two, three or four character strings containing the names of the independent variables.
data	data frame containing the data.
tName	optional character string specifying the name of the time variable (t).
vrs	logical. Allow for variable returns to scale?
method	character string indicating the estimation method: either "Kmenta" for the Kmenta approximation or "LM", "NM", "Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN", "Newton", "PORT", or "DE" for non-linear least-squares (see section 'Details').
start	optional numeric vector giving the starting values of the parameters in the non-linear estimations (see section 'Details').
lower	lower bounds of the parameters (see section 'Details').
upper	upper bounds of the parameters (see section 'Details').
multErr	logical. If TRUE, the error term is assumed to be multiplicative, i.e. $y = \hat{y} \cdot \exp(\epsilon)$. If FALSE (the default), the error term is assumed to be additive, i.e. $y = \hat{y} + \epsilon$.
rho1, rho2, rho	numeric scalar or vector at which the coefficients ρ_1 , ρ_2 , and/or ρ should be fixed; if argument rho1, rho2, or rho is NULL (default), this coefficient is estimated together with the other parameters; if these arguments have more than one element, a grid search for ρ_1 , ρ_2 , and/or ρ is performed (see section 'Details').
returnGridAll	logical value that indicates whether the estimates for all values of ρ obtained during the grid search (not just the estimations with the 'best' ρ) should be returned (ignored if argument rho is NULL or has only a single element).
returnGrad	logical value that indicates whether a matrix with the gradients of the dependent variable (i.e., y if argument multErr is FALSE and $\log(y)$ if argument multErr is TRUE) with respect to the parameters (evaluated at the estimated parameters) should be returned.
random.seed	an integer used to seed R's random number generator. This is to ensure replicability when the "SANN" or "DE" method is used. Defaults to 123.
rhoApprox	numeric vector with exactly 5 elements; the endogenous variable of the CES and the derivatives with respect to its coefficients are calculated using a first-order Taylor series approximation at $\rho = 0$ (non-nested CES) or by interpolation between ρ , ρ_1 , or ρ_2 equal to zero and ρ , ρ_1 , or ρ_2 equal to \pm rhoApprox (nested CES), if the absolute value of the coefficients ρ , ρ_1 , or ρ_2 is smaller than or equal to the corresponding element of this argument (see also argument rhoApprox of cesCalc); the first element determines the threshold for calculating the endogenous variable; the second element determines the threshold for calculating the derivatives with respect to γ ; the third element determines the threshold for

	calculating the derivatives with respect to δ_1 , δ_2 , and δ ; the fourth element determines the threshold for calculating the derivatives with respect to ρ , ρ_1 , and ρ_2 ; the fifth element determines the threshold for calculating the derivatives with respect to ν .
checkStart	logical. If TRUE (the default), it is checked whether the starting values are in the expected ranges for a production function.
x	an object of class <code>cesEst</code> .
digits	number of digits.
...	further arguments to <code>cesEst</code> are passed to <code>optim</code> , <code>nls.lm</code> , <code>nlm</code> , <code>nlminb</code> , or <code>DEoptim</code> ; further arguments to <code>print.cesEst</code> are currently ignored.

Details

Estimation method

Argument `method` determines the estimation method. If it is "Kmenta", the CES is estimated by ordinary least squares using the Kmenta approximation; otherwise, it is estimated by non-linear least-squares. Several different optimizers can be used for the non-linear estimation. The optimization method LM (Levenberg-Marquardt, see Moré 1978) uses `nls.lm` for the optimization. The optimization methods NM or Nelder-Mead (Nelder and Mead 1965), BFGS (Broyden 1970, Fletcher 1970, Goldfarb 1970, Shanno 1970), CG (Conjugate Gradients based on Fletcher and Reeves 1964), L-BFGS-B (with box-constraints, Byrd, Lu, Nocedal, and Zhu 1995), and SANN (Simulated Annealing, Bélisle 1992) use `optim` for the optimization. The optimization method Newton (Newton-type, see Dennis and Schnabel 1983 and Schnabel, Koontz, and Weiss 1985) uses `nlm` for the optimization. The optimization method PORT (PORT routines, see Gay 1990) uses `nlminb` for the optimization. The optimization method DE (Differential Evolution, see Storn and Price 1997) uses `DEoptim` for the optimization. Analytical gradients are used in the LM, BFGS, CG, L-BFGS-B, Newton, and PORT method.

Starting values

Argument `start` should be a numeric vector. The order must be as described in the documentation of argument `coef` of function `cesCalc`. However, names of the elements are ignored. If argument `start` is NULL, pre-defined starting values are used. The starting value of λ (if present) is set to 0.015; the starting values of δ_1 , δ_2 , and δ (if present) are set to 0.5, the starting values of ρ_1 , ρ_2 , and ρ (if present and required) are set to 0.25 (i.e. elasticity of substitution = 0.8 in the two-input case), the starting value of ν (if present) is set to 1, and the starting value of γ is set to a value so that the mean of the error term is zero. Hence, in case of an additive error term (i.e. argument `multErr` is set to FALSE, the default) γ is set to $\text{mean}(y) / \text{mean}(\text{CES}(X, \text{start1}))$ and in case of a multiplicative error term (i.e. argument `multErr` is set to TRUE) γ is set to $\text{mean}(\log(y)) - \text{mean}(\log(\text{CES}(X, \text{start1})))$, where y is the dependent variable (defined by argument `yName`), X is the set of covariates (defined by arguments `xNames` and `tName`), `CES()` defines the (nested) CES function, and `start1` is a coefficient vector with $\gamma = 1$ and all other coefficients having the starting values described above.

Lower and upper bounds

Arguments `lower` and `upper` can be used to set lower and upper bounds for the estimated parameters. If these arguments are `-Inf` and `Inf`, respectively, the parameters are estimated without constraints. By default, arguments `lower` and `upper` are both NULL, which means that the bounds

are set automatically depending on the estimation method: In case of the L-BFGS-B, PORT, and DE method, the lower bound is 0 for γ , δ_1 , δ_2 , and δ (if present), -1 for ρ_1 , ρ_2 , and ρ (if present), and eventually 0 for ν . In case of the L-BFGS-B and PORT method, the upper bound is infinity for γ , 1 for δ_1 , δ_2 , and δ (if present), infinity for ρ_1 , ρ_2 , and ρ (if present), and eventually infinity for ν . Since the ‘Differential Evolution’ algorithm requires finite bounds, the upper bounds for the DE method are set to 1e10 for γ , 1 for δ_1 , δ_2 , and δ (if present), 10 for ρ_1 , ρ_2 , and ρ (if present), and eventually 10 for ν . In case of all other estimation methods, the lower and upper bounds are set to -Inf and Inf, respectively, because these methods do not support parameter constraints. Of course, the user can specify own lower and upper bounds by setting arguments lower and upper to numeric vectors that should have the same format as argument start (see above).

Grid search for ρ

If arguments rho1, rho2, and/or rho have more than one element, a one-dimensional, two-dimensional, or three-dimensional grid search for ρ_1 , ρ_2 , and/or ρ is performed. The remaining (free) parameters of the CES are estimated by least-squares, where ρ_1 , ρ_2 , and/or ρ are fixed consecutively at each value defined in arguments rho1, rho2, and rho, respectively. Finally the estimation with the ρ_1 , ρ_2 , and/or ρ that results in the smallest sum of squared residuals is chosen (and returned).

Random numbers

The ‘state’ (or ‘seed’) of R’s random number generator is saved at the beginning of the cesEst function and restored at the end of this function so that this function does *not* affect the generation of random numbers although the random seed is set to argument random.seed and the ‘SANN’ and ‘DE’ algorithms use random numbers.

Value

cesEst returns a list of class cesEst that has following components:

coefficients	estimated coefficients/parameters of the CES (including a possible fixed ρ).
ela	constant elasticity/elasticities of substitution.
iter	number of iterations (only for non-linear least-squares estimations).
convergence	logical value indicating if the non-linear estimation has converged (only for non-linear least-squares estimations with solvers that have a convergence criterion).
message	additional information from the optimizer (only if a message was returned by optim or nls.lm).
vcov	approximate covariance matrix of the estimated parameters calculated from the parameters of the linearized model by the Delta method (only if argument method is "Kmenta").
cov.unscaled	unscaled covariance matrix of the estimated parameters (including a possible fixed ρ), i.e. the inverse of the cross-product of the gradient matrix evaluated at the estimated parameters.
fitted.values	the fitted values (\hat{y}).
residuals	the residuals (i.e. $y - \hat{y}$ if argument multErr is FALSE (the default), and $\log(y) - \log(\hat{y})$ if argument multErr is TRUE).
rss	the sum of the squared residuals (i.e. the value of the objective function of the non-linear least-squares estimation evaluated at the estimated parameters).

<code>call</code>	the matched call.
<code>method</code>	argument method.
<code>multErr</code>	argument <code>multErr</code> .
<code>start</code>	starting values for the non-linear estimation (not for the Kmenta and DE method).
<code>lower</code>	lower bounds of the parameters.
<code>upper</code>	upper bounds of the parameters.
<code>rho</code>	argument <code>rho</code> .
<code>nls.lm</code>	object returned by <code>nls.lm</code> (only if argument method is "LM").
<code>optim</code>	object returned by <code>optim</code> (only if argument method is "NM", "Nelder-Mead", "BFGS", "CG", "L-BFGS-B", or "SANN").
<code>nlm</code>	object returned by <code>nlm</code> (only if argument method is "Newton").
<code>nlminb</code>	object returned by <code>nlminb</code> (only if argument method is "PORT").
<code>DEoptim</code>	object returned by <code>DEoptim</code> (only if argument method is "DE").
<code>translog</code>	estimation results of the (unrestricted) translog model returned by <code>translogEst</code> (only if argument method is "Kmenta").
<code>kmenta</code>	estimation results of the Kmenta approximation (a restricted translog model) returned by <code>systemfit</code> (only if argument method is "Kmenta").
<code>testKmenta</code>	test of the restrictions implied by the Kmenta approximation (including constant returns to scale if argument <code>vrs</code> is FALSE) in the unrestricted translog model returned by <code>linear.hypothesis</code> (only if argument method is "Kmenta").
<code>allRhoSum</code>	data frame with summary results of the estimations with all values of ρ used in the grid search (only if a grid search was performed); this data frame has following columns: <code>rho</code> = the value of ρ , <code>rss</code> = the corresponding sum of squared residuals, and (if appropriate for the method used for the estimation) <code>convergence</code> = logical value indicating whether the estimation converged.
<code>allRhoFull</code>	list of estimation results returned by <code>cesEst</code> for all values of ρ used in the grid search (only if a grid search was performed and argument <code>returnGridAll</code> is set to TRUE).
<code>rho1Values, rho2Values, rhoValues</code>	numeric vectors giving the values that are used in the grid search for the coefficients ρ_1 and ρ , respectively (only if a grid search was performed).
<code>rssArray</code>	matrix or array of the RSS values obtained by a two-dimensional or three-dimensional grid search for the coefficients ρ_1 (first dimension, e.g. rows of a matrix), ρ_2 , and ρ (last dimension, e.g. columns of a matrix) (only if a two-dimensional or threedimensional grid search was performed).
<code>grad</code>	matrix with the gradients of the dependent variable (i.e., y if argument <code>multErr</code> is FALSE and $\log(y)$ if argument <code>multErr</code> is TRUE) with respect to the parameters evaluated at the estimated parameters (only if argument <code>returnGrad</code> is set to TRUE).

Author(s)

Arne Henningsen and Geraldine Henningsen

References

- Bélisle, C.J.P. (1992): Convergence theorems for a class of simulated annealing algorithms on \mathbb{R}^d , *Journal of Applied Probability* 29, p. 885-895.
- Broyden, C.G. (1970): The Convergence of a Class of Double-rank Minimization Algorithms, *Journal of the Institute of Mathematics and Its Applications* 6, p. 76-90.
- Byrd, R.H., Lu, P., Nocedal, J. and Zhu, C. (1995): A limited memory algorithm for bound constrained optimization, *SIAM J. Scientific Computing* 16, p. 1190-1208.
- Dennis, J.E. and Schnabel, R.B. (1983): *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Englewood Cliffs, NJ.
- Fletcher, R. (1970): A New Approach to Variable Metric Algorithms, *Computer Journal* 13, p. 317-322.
- Fletcher, R. and Reeves, C.M. (1964): Function minimization by conjugate gradients, *Computer Journal* 7, p. 148-154.
- Gay, D.M. (1990): Usage Summary for Selected Optimization Routines, Computing Science Technical Report No. 153, AT&T Bell Laboratories, Murray Hill NJ.
- Goldfarb, D. (1970): A Family of Variable Metric Updates Derived by Variational Means, *Mathematics of Computation* 24, p. 23-26.
- Moré, J.J. (1978): The Levenberg-Marquardt algorithm: implementation and theory, in G.A. Watson (Ed.), *Lecture Notes in Mathematics 630: Numerical Analysis*, pp. 105-116, Springer-Verlag: Berlin.
- Nelder, J.A. and Mead, R. (1965): A simplex algorithm for function minimization, *Computer Journal* 7, p. 308-313.
- Schnabel, R.B., Koontz, J.E. and Weiss, B.E. (1985): A modular system of algorithms for unconstrained minimization, *ACM Trans. Math. Software*, 11, pp. 419-440.
- Shanno, D.F. (1970): Conditioning of Quasi-Newton Methods for Function Minimization, *Mathematics of Computation* 24, p. 647-656.
- Storn, R. and Price, K. (1997): Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, *Journal of Global Optimization*, 11(4), p. 341-359.

See Also

[summary.cesEst](#) for the summary method, [plot.cesEst](#) for plotting the results of the grid search for ρ , [coef.cesEst](#) for several further methods, [cesCalc](#) for calculations or simulations with the CES, [translogEst](#) for estimating translog functions, and [quadFuncEst](#) for estimating quadratic functions.

Examples

```
data( germanFarms, package = "micEcon" )
# output quantity:
germanFarms$qOutput <- germanFarms$vOutput / germanFarms$pOutput
# quantity of intermediate inputs
germanFarms$qVarInput <- germanFarms$vVarInput / germanFarms$pVarInput
```

```

## CES: Land & Labor (Levenberg-Marquardt algorithm)
cesLandLabor <- cesEst( "qOutput", c( "land", "qLabor" ), germanFarms )

# variable returns to scale, increased max. number of iter. (LM algorithm)
cesLandLaborVrs <- cesEst( "qOutput", c( "land", "qLabor" ), germanFarms,
  vrs = TRUE, control = nls.lm.control( maxiter = 1000 ) )

# using the Nelder-Mead optimization method
cesLandLaborNm <- cesEst( "qOutput", c( "land", "qLabor" ), germanFarms,
  method = "NM" )

# using the BFGS optimization method
cesLandLaborBfgs <- cesEst( "qOutput", c( "land", "qLabor" ), germanFarms,
  method = "BFGS" )

# using the L-BFGS-B optimization method with constrained parameters
cesLandLaborBfgsCon <- cesEst( "qOutput", c( "land", "qLabor" ),
  germanFarms, method = "L-BFGS-B" )

# using the CG optimization method
cesLandLaborSann <- cesEst( "qOutput", c( "land", "qLabor" ), germanFarms,
  method = "CG" )

# using the SANN optimization method
# (with decreased number of iteration to decrease execution time)
cesLandLaborSann <- cesEst( "qOutput", c( "land", "qLabor" ), germanFarms,
  method = "SANN", control = list( maxit = 1000 ) )

# using the Kmenta approximation
cesLandLaborKmenta <- cesEst( "qOutput", c( "land", "qLabor" ), germanFarms,
  method = "Kmenta" )

# using the PORT optimization routine with unconstrained parameters
cesLandLaborPortCon <- cesEst( "qOutput", c( "land", "qLabor" ),
  germanFarms, vrs = TRUE, method = "PORT", lower = -Inf, upper = Inf )

# using the PORT optimization routine with constrained parameters and VRS
cesLandLaborPortCon <- cesEst( "qOutput", c( "land", "qLabor" ),
  germanFarms, vrs = TRUE, method = "PORT" )

# using the Differential Evolution optimization method
# (with decreased number of iteration to decrease execution time)
cesLandLaborDe <- cesEst( "qOutput", c( "land", "qLabor" ), germanFarms,
  method = "DE", control = DEoptim.control( itermax = 50 ) )

## estimation with a grid search for rho (using the LM algorithm)
cesLandInt <- cesEst( "qOutput", c( "land", "qLabor" ),
  data = germanFarms, rho = seq( from = -0.6, to = 0.9, by = 0.3 ) )

```

Description

Methods for Objects of Class `cesEst` and `cesEst`.

Usage

```
## S3 method for class 'cesEst'
coef( object, ... )
## S3 method for class 'summary.cesEst'
coef( object, ... )
## S3 method for class 'cesEst'
fitted( object, ... )
## S3 method for class 'cesEst'
residuals( object, ... )
## S3 method for class 'cesEst'
vcov( object, ... )
```

Arguments

<code>object</code>	an object of class <code>cesEst</code> or <code>summary.cesEst</code> .
<code>...</code>	further arguments are currently ignored.

Value

`coef.cesEst` returns a vector of the estimated coefficients.

`coef.summary.cesEst` returns a matrix with four columns: the estimated coefficients/parameters of the CES, their standard errors, the t-statistic, and corresponding (two-sided) P-values.

`fitted.cesEst` returns a vector of the fitted values.

`residuals.cesEst` returns a vector of the residuals.

`vcov.cesEst` returns the variance covariance matrix of the estimated coefficients.

Author(s)

Arne Henningsen and Geraldine Henningsen

See Also

`cesEst` and `summary.cesEst`.

Examples

```
data( germanFarms, package = "micEcon" )
# output quantity:
germanFarms$qOutput <- germanFarms$vOutput / germanFarms$pOutput
# quantity of intermediate inputs
germanFarms$qVarInput <- germanFarms$vVarInput / germanFarms$pVarInput

## CES: Land & Labor
```

```

cesLandLabor <- cesEst( "qOutput", c( "land", "qLabor" ), germanFarms )

# estimated coefficients
coef( cesLandLabor )

# estimated coefficients, their standard errors, t-statistic, P-values
coef( summary( cesLandLabor ) )

# fitted values of the estimated model
fitted( cesLandLabor )

# residuals of the estimated model
residuals( cesLandLabor )

# covariance matrix of the estimated coefficients
vcov( cesLandLabor )

```

`durbinWatsonTest.cesEst`

Durbin Watson Test for Estimated CES Functions

Description

Conduct a generalized Durbin-Watson-Test as suggested by White (1992) to test for serial correlation of the residuals. `dwt` is an abbreviation for `durbinWatsonTest`.

Usage

```

## S3 method for class 'cesEst'
durbinWatsonTest( model, ... )

```

Arguments

<code>model</code>	object returned by <code>cesEst</code> ; in the call to <code>cesEst</code> , argument <code>returnGrad</code> must be set to <code>TRUE</code> .
<code>...</code>	further arguments are passed to durbinWatsonTest.lm .

Author(s)

Arne Henningsen

References

White, K.J. (1992): The Durbin-Watson Test for Autocorrelation in Nonlinear Models. *The Review of Economics and Statistics* 74(2), p. 370-373.

See Also

[cesEst](#), [durbinWatsonTest](#).

Examples

```
data( germanFarms, package = "micEcon" )
# output quantity:
germanFarms$qOutput <- germanFarms$vOutput / germanFarms$pOutput
# quantity of intermediate inputs
germanFarms$qVarInput <- germanFarms$vVarInput / germanFarms$pVarInput

## CES: Land & Intermediate Inputs
cesLandInt <- cesEst( yName = "qOutput",
  xNames = c( "land", "qVarInput" ), data = germanFarms,
  returnGrad = TRUE )

# conduct the generalized Durbin-Watson test
dwt( cesLandInt )
```

GermanIndustry

Aggregated Time Series Data for the West German Industry

Description

The data frame GermanIndustry contains annual aggregated data of the entire West German industry from 1960 until 1993 as well as data of seven industrial sectors from 1970 to 1988/1992. This data set has been used by Kemfert (1998).

Usage

```
data(GermanIndustry)
```

Format

This data frame contains the following columns/variables:

year the year.

Y output: gross value added of the West German industrial sector (in billion Deutsche Mark at prices of 1991).

K capital: gross stock of fixed assets of the West German industrial sector (in billion Deutsche Mark at prices of 1991).

A labor: total number of persons employed in the West German industrial sector (in million).

E energy: final energy consumption in the West German industrial sector (in GWh).

C_Y gross value added of the West German chemical industry (in billion Deutsche Mark at prices of 1991).

C_K capital: gross stock of fixed assets of the West German chemical industry (in billion Deutsche Mark at prices of 1991).

C_A labor: total number of persons employed in the West German chemical industry (in thousands).

C_E final energy consumption in the West German chemical industry (in GWh).

- S_Y** gross value added of the West German stone and earth industry (in billion Deutsche Mark at prices of 1991).
- S_K** capital: gross stock of fixed assets of the West German stone and earth industry (in billion Deutsche Mark at prices of 1991).
- S_A** labor: total number of persons employed in the West German stone and earth industry (in thousands).
- S_E** final energy consumption in the West German stone and earth industry (in GWh).
- I_Y** gross value added of the West German iron industry (in billion Deutsche Mark at prices of 1991).
- I_K** capital: gross stock of fixed assets of the West German iron industry (in billion Deutsche Mark at prices of 1991).
- I_A** labor: total number of persons employed in the West German iron industry (in thousands).
- I_E** final energy consumption in the West German iron industry (in GWh).
- N_Y** gross value added of the West German non-ferrous industry (in billion Deutsche Mark at prices of 1991).
- N_K** capital: gross stock of fixed assets of the West German non-ferrous industry (in billion Deutsche Mark at prices of 1991).
- N_A** labor: total number of persons employed in the West German non-ferrous industry (in thousands).
- N_E** final energy consumption in the West German non-ferrous industry (in GWh).
- V_Y** gross value added of the West German vehicle industry (in billion Deutsche Mark at prices of 1991).
- V_K** capital: gross stock of fixed assets of the West German vehicle industry (in billion Deutsche Mark at prices of 1991).
- V_A** labor: total number of persons employed in the West German vehicle industry (in thousands).
- V_E** final energy consumption in the West German vehicle industry (in GWh).
- P_Y** gross value added of the West German paper industry (in billion Deutsche Mark at prices of 1991).
- P_K** capital: gross stock of fixed assets of the West German paper industry (in billion Deutsche Mark at prices of 1991).
- P_A** labor: total number of persons employed in the West German paper industry (in thousands).
- P_E** final energy consumption in the West German paper industry (in GWh).
- F_Y** gross value added of the West German food industry (in billion Deutsche Mark at prices of 1991).
- F_K** capital: gross stock of fixed assets of the West German food industry (in billion Deutsche Mark at prices of 1991).
- F_A** labor: total number of persons employed in the West German food industry (in thousands).
- F_E** final energy consumption in the West German food industry (in GWh).

Note

Please note that Kemfert (1998) disregards the years 1973-1975 in her estimations due to economic disruptions.

Source

German Federal Statistical Office (Statistisches Bundesamt), data taken from Kemfert (1998).

References

Kemfert, Claudia (1998): Estimated Substitution Elasticities of a Nested CES Production Funktion Approach for Germany, Energy Economics 20: 249-264 (doi:10.1016/S0140-9883(97)00014-5)

MishraCES

Mishra's (2006) CES data

Description

The MishraCES data set contains artificial production data. It has 50 observations (e.g. firms, sectors, or countries).

Usage

```
data(MishraCES)
```

Format

This data frame contains the following columns:

No Firm number.

Y Output quantity.

X1 Quantity of first input.

X2 Quantity of second input.

X3 Quantity of third input.

X4 Quantity of fourth input.

Source

Mishra, SK (2006): A Note on Numerical Estimation of Sato's Two-Level CES Production Function MPRA Working Paper No. 1019, <https://mpra.ub.uni-muenchen.de/1019/>.

Examples

```
# load the data set
data( "MishraCES" )

# show mean values of all variables
colMeans( MishraCES )

# re-calculate the endogenous variable (see Mishra 2006)
# coefficients of the nested CES function with 4 inputs
b <- c( "gamma" = 200 * 0.5^(1/0.6), "delta_1" = 0.6, "delta_2" = 0.3,
```

```
"delta" = 0.5, "rho_1" = 0.5, "rho_2" = -0.17, "rho" = 0.6 )
MishraCES$Y2 <- cesCalc( xNames = c( "X1", "X2", "X3", "X4" ),
  data = MishraCES, coef = b, nested = TRUE )
all.equal( MishraCES$Y, MishraCES$Y2 )
```

plot.cesEst

*Plot RSSs of a CES Function Estimated by Grid Search***Description**

Plot a scatter plot, where the values of ρ are on the x axis and the corresponding sums of the squared residuals obtained by a grid search for ρ are on the y axis. Estimations that did not converge are marked with red.

Note that this method can be applied only if the model was estimated by a grid search for ρ , i.e. `cesEst` was called with argument `rho` set to a vector of more than one values for ρ .

Usage

```
## S3 method for class 'cesEst'
plot( x, negRss = TRUE, bw = FALSE, ... )
```

Arguments

<code>x</code>	object returned by <code>cesEst</code> if it was called with argument <code>rho</code> set a vector containing more than one value for ρ so that a grid search was performed.
<code>negRss</code>	logical. Indicates whether the <i>negative</i> sum of squared residuals should be plotted in 3D plots (ignored in 2D plots).
<code>bw</code>	logical. Indicates whether 3D plots should be in black-and-white or colored.
<code>...</code>	All further arguments are passed to <code>plot.default</code> or <code>persp</code> .

Author(s)

Arne Henningsen and Geraldine Henningsen

See Also

[cesEst](#).

Examples

```
data( germanFarms, package = "micEcon" )
# output quantity:
germanFarms$qOutput <- germanFarms$vOutput / germanFarms$pOutput
# quantity of intermediate inputs
germanFarms$qVarInput <- germanFarms$vVarInput / germanFarms$pVarInput

## CES: Land & Intermediate Inputs
```

```
cesLandInt <- cesEst( yName = "qOutput",
  xNames = c( "land", "qVarInput" ), data = germanFarms,
  rho = seq( from = -0.6, to = 0.9, by = 0.3 ) )

# plot the rhos against the sum of squared residuals
plot( cesLandInt )
```

summary.cesEst

*Summarize Estimation of a CES Function***Description**

summary method for objects of class `cesEst`.

Usage

```
## S3 method for class 'cesEst'
summary( object, rSquaredLog = object$multErr, ela = TRUE, ... )
## S3 method for class 'summary.cesEst'
print( x, ela = TRUE, digits = max(3, getOption("digits") - 3),
  ... )
```

Arguments

<code>object</code>	an object returned by <code>cesEst</code> .
<code>rSquaredLog</code>	logical. If FALSE (the default for models with additive error term), the returned R^2 -value measures the fraction of the explained variance of the dependent variable in natural units. If TRUE (the default for models with multiplicative error term), the returned R^2 -value measures the fraction of the explained variance of the <i>logarithmized</i> dependent variable.
<code>ela</code>	logical. If TRUE (the default), the summary method calculates the (co)variances of the constant elasticities of substitution and the print method prints these elasticities together with corresponding summary statistics. If FALSE, the summary method does not calculate the (co)variances of the constant elasticities of substitution and the print method does not print these elasticities.
<code>x</code>	an object returned by <code>summary.cesEst</code> .
<code>digits</code>	number of digits.
<code>...</code>	further arguments are currently ignored.

Value

`summary.cesEst` returns a list of class `summary.cesEst` that contains the elements of the provided object with with following changes or additions:

<code>coefficients</code>	a matrix with four columns: the estimated coefficients/parameters of the CES (including a possible fixed ρ), their standard errors, the t-statistic, and corresponding (two-sided) P-values.
---------------------------	--

sigma	square root of the estimated (asymptotic) variance of the random error (calculated without correcting for degrees of freedom).
r.squared	R^2 -value, i.e. the ‘fraction of variance explained by the model’. If argument rSquaredLog is TRUE, the R^2 -value measures the fraction of the explained variance of the <i>logarithmized</i> dependent variable.
vcov	covariance matrix of the estimated parameters (including a possible fixed ρ).
ela	a matrix with four columns: the estimated elasticities of substitution, their standard errors, the t-statistic, and corresponding (two-sided) P-values (only if argument ela is TRUE).
elaCov	covariance matrix of the estimated elasticities of substitution (only if argument ela is TRUE).

Author(s)

Arne Henningsen

See Also[cesEst](#) and [cesCalc](#).**Examples**

```
data( germanFarms, package = "micEcon" )
# output quantity:
germanFarms$qOutput <- germanFarms$vOutput / germanFarms$pOutput
# quantity of intermediate inputs
germanFarms$qVarInput <- germanFarms$vVarInput / germanFarms$pVarInput

## CES: Land & Labor
cesLandLabor <- cesEst( "qOutput", c( "land", "qLabor" ), germanFarms )

# print summary results
summary( cesLandLabor )
```

Index

- * **datasets**
 - GermanIndustry, [13](#)
 - MishraCES, [15](#)
- * **models**
 - cesCalc, [2](#)
 - cesEst, [4](#)
 - cesEst-methods, [10](#)
 - durbinWatsonTest.cesEst, [12](#)
 - plot.cesEst, [16](#)
 - summary.cesEst, [17](#)
- * **nonlinear**
 - cesEst, [4](#)
 - durbinWatsonTest.cesEst, [12](#)
 - plot.cesEst, [16](#)
- * **regression**
 - cesEst, [4](#)
 - durbinWatsonTest.cesEst, [12](#)
 - plot.cesEst, [16](#)

cesCalc, [2](#), [4–6](#), [9](#), [18](#)
cesEst, [3](#), [4](#), [8](#), [11](#), [12](#), [16–18](#)
cesEst-methods, [10](#)
coef.cesEst, [9](#)
coef.cesEst (cesEst-methods), [10](#)
coef.summary.cesEst (cesEst-methods), [10](#)

DEoptim, [6](#), [8](#)
durbinWatsonTest, [12](#)
durbinWatsonTest.cesEst, [12](#)
durbinWatsonTest.lm, [12](#)
dwt.cesEst (durbinWatsonTest.cesEst), [12](#)

fitted.cesEst (cesEst-methods), [10](#)

GermanIndustry, [13](#)

linear.hypothesis, [8](#)

MishraCES, [15](#)

nlm, [6](#), [8](#)

nlminb, [6](#), [8](#)
nls.lm, [6–8](#)

optim, [6–8](#)

persp, [16](#)
plot.cesEst, [9](#), [16](#)
plot.default, [16](#)
print.cesEst (cesEst), [4](#)
print.summary.cesEst (summary.cesEst),
[17](#)

quadFuncEst, [9](#)

residuals.cesEst (cesEst-methods), [10](#)

summary.cesEst, [9](#), [11](#), [17](#)
systemfit, [8](#)

translogEst, [8](#), [9](#)

vcov.cesEst (cesEst-methods), [10](#)