Package ‘rDEA’

November 25, 2016

Version 1.2-5

Title Robust Data Envelopment Analysis (DEA) for R

Description Data Envelopment Analysis for R, estimating robust DEA scores without and with environmental variables and doing returns-to-scale tests.

Imports slam (>= 0.1-9), truncreg (>= 0.2-1), trunchnorm (>= 1.0-7), maxLik

Suggests testthat

SystemRequirements GLPK (>= 4.52)

License GPL-2 | GPL-3

URL https://github.com/jaak-s/rDEA

Author Jaak Simm [aut, cre],
Galina Bestremyannaya [aut]

Maintainer Jaak Simm <jaak.simm@gmail.com>

NeedsCompilation yes

Repository CRAN

Date/Publication 2016-11-25 08:43:04

R topics documented:

dea ................................................................. 2
dea.env.robust ............................................ 4
dea.robust .................................................. 6
hospitals ...................................................... 8
multi_glpk_solve_LP ...................................... 9
rts.test ...................................................... 12

Index 16
Data envelopment analysis scores

Description
Naive scores in input-oriented, output-oriented and cost-minimization DEA models

Usage
dea(XREF, YREF, X, Y, W=NULL, model, RTS="variable")

Arguments
- XREF: a matrix of inputs for observations used for constructing the frontier.
- YREF: a matrix of outputs for observations used for constructing the frontier.
- X: a matrix of inputs for observations, for which DEA scores are estimated.
- Y: a matrix of outputs for observations, for which DEA scores are estimated.
- W: a matrix of input prices for observations, for which DEA scores are estimated in cost-minimization model, W=NULL for input- and output-oriented models.
- model: a string for the type of DEA model to be estimated, "input" for input-oriented, "output" for output-oriented, "costmin" for cost-minimization model.
- RTS: a string for returns-to-scale under which DEA scores are estimated, RTS can be "constant", "variable" or "non-increasing".

Details
Estimates technical efficiency scores (input- and output-oriented DEA models) and cost efficiency score (cost-minimization DEA).

Value
A list containing the optimal solutions for linear optimization problem for each firm, with the following components.
- thetaOpt: a vector of DEA scores in input- or output-oriented model, thetaOpt is in (0,1).
- gammaOpt: a vector of DEA scores in cost-minimization model.
- XOpt: the matrix of optimal values of inputs, only returned for cost-minimization model.
- lambda: the matrix of values for constraint coefficients in the corresponding linear optimization problem, lambda >=0.
- lambda_sum: the vector for sum of constraint coefficients in the corresponding linear optimization problem, lambda_sum=1 for variable returns-to-scale, lambda_sum <=1 for non-increasing returns-to-scale.
Author(s)
Jaak Simm, Galina Besstremyannaya

References


See Also
dea.robust, dea.env.robust and hospitals.

Examples
```r
## load data on Japanese hospitals (Besstremyannaya 2013, 2011)
data("hospitals", package="rDEA")

## inputs and outputs for analysis
Y = hospitals[c('inpatients', 'outpatients')]
X = hospitals[c('labor', 'capital')]
W = hospitals[c('labor_price', 'capital_price')]

## Naive input-oriented DEA score for the first 20 firms under variable returns-to-scale
firms=1:20
di_naive = dea(XREF=X, YREF=Y, X=X[firms,,], Y=Y[firms,,], model="input", RTS="variable")
di_naive$thetaOpt

## Naive DEA score in cost-minimization model for the first 20 firms under variable returns-to-scale
ci_naive = dea(XREF=X, YREF=Y, X=X[firms,,], Y=Y[firms,,], W=W[firms,,],
               model="costmin", RTS="variable")
ci_naive$XOpt
ci_naive$gammaOpt
```
dea.env.robust

Bias-corrected data envelopment analysis with environmental variables

Description

Estimates bias-corrected efficiency scores in input- and output-oriented DEA models with environmental (exogenous) variables.

Usage

dea.env.robust (X, Y, W=NULL, Z, model, RTS="variable", L1=100, L2=2000, alpha=0.05)

Arguments

X  a matrix of inputs for observations, for which DEA scores are estimated.
Y  a matrix of outputs for observations, for which DEA scores are estimated.
W  a matrix of input prices, only used if model="costmin".
Z  a matrix of environmental variables for observations, for which DEA scores are estimated. Constant is automatically included in Z.
model a string for the type of DEA model to be estimated, "input" for input-oriented, "output" for output-oriented, "costmin" for cost-minimization model.
RTS a string for returns-to-scale under which DEA scores are estimated, RTS can be "constant", "variable" or "non-increasing".
L1  an integer showing the number of bootstrap replications in the first loop of Simar and Wilson’s (2007) algorithm, default is 100.
L2  an integer showing the number of bootstrap replications in the second loop of Simar and Wilson’s (2007) algorithm, default is 2000.
alpha a number in (0,1) denoting the size of confidence interval for the bias-corrected DEA score, default is 0.05.

Details

Implements Simar and Wilson’s (2007) second algorithm for bias-correction of technical efficiency scores in input- and output-oriented DEA models. Computations are done in terms of distance function, i.e. the reciprocal of efficiency score, with the range from one to infinity.

Value

A list containing bias-corrected scores for each firm, with the following components.

delta_hat the vector of the reciprocal of DEA score (distance function), estimated in input or output-oriented model in with dea function.
beta_hat  the vector of coefficients in the truncated regression of the reciprocal of DEA score on environmental variables.

sigma_hat  the standard deviation of the errors in the truncated regression of reciprocal of DEA score on environmental variables.

beta_hat_hat  the vector of robust coefficients in the truncated regression of reciprocal of DEA score on environmental variables (after the second loop).

sigma_hat_hat  the robust standard deviation of the errors in the truncated regression of reciprocal of DEA score on environmental variables (after the second loop).

delta_hat_hat  the vector of robust reciprocal of DEA score (after the second loop).

bias  the vector of bias of the reciprocal of DEA score, bias is non-positive.

delta_ci_low  the vector of the lower bounds of confidence interval for delta_hat_hat (bias-corrected reciprocal of DEA score).

delta_ci_high  the vector of the upper bounds of confidence interval for delta_hat_hat (bias-corrected reciprocal of DEA score).

beta_ci  the matrix of lower and upper bounds for beta using alpha confidence intervals (beta is robust coefficients in the truncated regression of the reciprocal of DEA score on environmental variables).

sigma_ci  the matrix of lower and upper bounds for sigma using alpha confidence intervals (sigma is the robust standard deviation in the truncated regression of the reciprocal of DEA score on environmental variables).

Author(s)
Jaak Simm, Galina Besstremyannaya

References

See Also
dea, dea.robust and hospitals.
Examples

```r
## load data on Japanese hospitals (Besstremyannaya 2013, 2011)
data("hospitals", package="rDEA")
Y = hospitals[c('inpatients', 'outpatients')]
X = hospitals[c('labor', 'capital')]
Z = hospitals[c('z1')]

## Naive input-oriented DEA score for the first 20 firms
## under variable returns-to-scale
firms=1:20
di_naive = dea(XREF=X, YREF=Y, X=X[firms,], Y=Y[firms,],
model="input", RTS="variable")
di_naive$thetaOpt

di_env = dea.env.robust(
X=X[firms,], Y=Y[firms,], Z=Z[firms,], model="input",
RTS="variable", L2=100, alpha=0.05)
di_env$delta_hat_hat
```

**dea.robust**  
*Bias-corrected data envelopment analysis*

**Description**

Estimates bias-corrected scores for input- and output-oriented models

**Usage**

```r
dea.robust (X, Y, W=NULL, model, RTS="variable", B=1000, alpha=0.05,
bw="bw.ucv", bw_mult=1)
```

**Arguments**

- **X**
  a matrix of inputs for observations, for which DEA scores are estimated.
- **Y**
  a matrix of outputs for observations, for which DEA scores are estimated.
- **W**
  a matrix of input prices, only used if model="costmin".
- **model**
  a string for the type of DEA model to be estimated, "input" for input-oriented, "output" for output-oriented, "costmin" for cost-minimization model.
- **RTS**
  a string for returns-to-scale under which DEA scores are estimated, RTS can be "constant", "variable" or "non-increasing".
- **B**
  an integer showing the number of bootstrap replications, the default is B=1000.
alpha a number in (0,1) for the size of confidence interval for the bias-corrected DEA score.

bw a string for the type of bandwidth used as a smoothing parameter in sampling with reflection, "cv" or "bw.ucv" for cross-validation bandwidth, "silverman" or "bw.nrd0" for Silverman’s (1986) rule.

bw_mult bandwidth multiplier, default is 1 that means no change.

Details


Value

A list containing bias-corrected scores for each firm, with the following components.

theta_hat_hat the vector of bias-corrected DEA score for each firm, theta_hat_hat is in the range of zero to one.

bias the vector of bias for naive DEA scores, bias is non-negative.

theta_ci_low the vector for the lower bounds of confidence interval for bias-corrected DEA score.

theta_ci_high the vector for the upper bounds of confidence interval for bias-corrected DEA score.

Author(s)

Jaak Simm, Galina Besstremyannaya

References


See Also
dea, dea.env.robust and hospitals.

Examples

```r
## load data on Japanese hospitals (Besstremyannaya 2013, 2011)
data("hospitals", package="rDEA")
Y = hospitals[c('inpatients', 'outpatients')]
X = hospitals[c('labor', 'capital')]

## Naive input-oriented DEA score for the first 20 firms under variable returns-to-scale
firms=1:20
di_naive = dea(XREF=X, YREF=Y, X=X[firms,], Y=Y[firms,],
    model="input", RTS="variable")
di_naive$thetaOpt

## Bias-corrected DEA score in input-oriented model under variable returns-to-scale
firms

di_robust = dea.robust(X=X[firms,], Y=Y[firms,], model="input",
                    RTS="variable", B, alpha=0.05, bw="cv")
di_robust$theta_hat_hat
di_robust$bias
```

**hospitals**

Data on Japanese local public hospitals

Description

Administrative dataset for financial variables and selected characteristics of Japanese local public hospitals. Processed for fiscal year 1999 (Heisei 11) and exploited in Besstremyannaya (2013, 2011)

Usage

hospitals

Format

The processed dataset contains anonymous observations for 958 local public hospitals, identified by a researcher-generated variable "firm_id". Hospital outputs are annual numbers of inpatients and outpatients, denoted "inpatients" and "outpatients", respectively. Hospital inputs are "labor" (total number of employees) and "capital" (total number of beds). The price of labor is per capita annual salary (th.yen) and the price of capital is the sum of depreciation and interest per bed (th.yen). Corresponding variable names are "labor_price" and "capital_price". A vector of environmental variables (z1) is the number of examinations per 100 patients.
References


Examples

data("hospitals", package="rDEA")
Y = hospitals[,c('inpatients', 'outpatients')]
X = hospitals[,c('labor', 'capital')]
W = hospitals[,c('labor_price', 'capital_price')]
Z= hospitals[,c('z1')]

## DEA score in input-oriented model
firms=1:20
di_naive = dea(XREF=X, YREF=Y, X=X[firms,], Y=Y[firms,], model="input", RTS="variable")
di_naive$thetaOpt

## added only for testing of the manual library(maxLik)

## robust DEA score in input-oriented model with environmental inputs
di_env = dea.env.robust(X=X[firms,], Y=Y[firms,], Z=Z[firms,],
model="input", RTS="variable", L1, L2=100, alpha=0.05)
di_env$delta_hat_hat

## DEA score in cost-minimization model
ci_naive = dea(XREF=X, YREF=Y, X=X[firms,], Y=Y[firms,], W=W[firms,],
model="costmin", RTS="variable")
ci_naive$xOpt
ci_naive$gammaOpt

---

multi_glpk_solve_LP  Multi Problem Solver for Linear and Mixed Integer Programming Using GLPK

Description

High level R interface to the GNU Linear Programming Kit (GLPK) for solving multiple linear as well as mixed integer linear programming (MILP) problems. Solving multiple problems at the same time allows to avoid R communication overhead, critical when solving many small problems.

Usage

multi_glpk_solve_LP(obj, mat, dir, rhs, bounds = NULL, types = NULL, max = FALSE, control = list(),
  mobj_i = NULL, mobj_val = NULL,
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obj</td>
<td>A numeric vector representing the objective coefficients.</td>
</tr>
<tr>
<td>mat</td>
<td>A numeric vector or a matrix of constraint coefficients.</td>
</tr>
<tr>
<td>dir</td>
<td>A character vector with the directions of the constraints. Each element must be one of &quot;&lt;&quot;, &quot;\leq&quot;, &quot;&gt;&quot;, &quot;\geq&quot;, or &quot;\neq&quot;.</td>
</tr>
<tr>
<td>rhs</td>
<td>The right hand side of the constraints.</td>
</tr>
<tr>
<td>bounds</td>
<td>NULL (default) or a list with elements upper and lower containing the indices and corresponding bounds of the objective variables. The default for each variable is a bound between 0 and Inf.</td>
</tr>
<tr>
<td>types</td>
<td>A character vector indicating the types of the objective variables. types can be either &quot;B&quot; for binary, &quot;C&quot; for continuous or &quot;I&quot; for integer. By default NULL, taken as all-continuous. Recycled as needed.</td>
</tr>
<tr>
<td>max</td>
<td>A logical giving the direction of the optimization. TRUE means that the objective is to maximize the objective function, FALSE (default) means to minimize it.</td>
</tr>
<tr>
<td>control</td>
<td>A list of parameters to the solver. Currently the only options are: verbose, a logical for turning on/off additional solver output; canonicalize_status, a logical indicating whether to canonicalize GLPK status codes or not. Defaults: FALSE; TRUE.</td>
</tr>
<tr>
<td>mobj_i</td>
<td>A vector of objective coefficient indices which will get different values in each optimization problem. Defaults: NULL.</td>
</tr>
<tr>
<td>mobj_val</td>
<td>A matrix of objective coefficient values. Each column specifies for one optimization problem the values of the objective coefficients specified by in mobj_i.</td>
</tr>
<tr>
<td>mmat_i</td>
<td>A matrix of coordinates of mat matrix. Each row specifies one constraint cell (its row and column). The cell specified in row i will be assigned values from row i of matrix mmat_val. Cells not specified in mat will be left unchanged. Defaults: NULL.</td>
</tr>
<tr>
<td>mmat_val</td>
<td>A matrix of values, each column specifies values for one optimization task. Cell specified in row i in mmat_i gets values from row i of mmat_val. Defaults: NULL.</td>
</tr>
<tr>
<td>mrhs_i</td>
<td>A vector of RHS constraint rows that will get different values in each optimization problem. Defaults: NULL.</td>
</tr>
<tr>
<td>mrhs_val</td>
<td>A matrix of RHS values. Element mrhs_val[i,j] specifies RHS value for row mrhs_i[i] in optimization problem j. Defaults: NULL.</td>
</tr>
</tbody>
</table>

Details

Package rDEA provides method for Data Envelopment Analysis (DEA), including standard input, output and cost-minimization DEA estimation and also robust DEA solvers. The latter can be with or without additional environmental variables.
Value
A list containing the optimal solutions for each problem, with the following components.

solution the matrix of optimal coefficients, each column is one problem
objval the vector of values of the objective function at the optimum, for each problem
status the vector of integers with status information about the solution returned, for each problem. If the control parameter canonicalize_status is set (the default) then it will return 0 for the optimal solution being found, and non-zero otherwise. If the control parameter is set to FALSE it will return the GLPK status codes.

Author(s)
Jaak Simm

References
GNU Linear Programming Kit (http://www.gnu.org/software/glpk/glpk.html).

See Also
glpk and glpkAPI for C API bindings; Rglpk_solve in package Rglpk.

Examples

## Simple linear program.
## maximize: 2 x_1 + 4 x_2 + 3 x_3
## subject to: 3 x_1 + 4 x_2 + 2 x_3 <= 60
## 2 x_1 + x_2 + 2 x_3 <= 40
## x_1 + 3 x_2 + 2 x_3 <= 80
## x_1, x_2, x_3 are non-negative real numbers

obj <- c(2, 4, 3)
mat <- matrix(c(3, 2, 1, 4, 1, 3, 2, 2, 2), nrow = 3)
dir <- c("<=", "<=", "<=")
rhs <- c(60, 40, 80)
max <- TRUE

multi_glpk_solve_LP(obj, mat, dir, rhs, max = max)

## Simple mixed integer linear program.
## maximize: 3 x_1 + 1 x_2 + 3 x_3
## subject to: -1 x_1 + 2 x_2 + x_3 <= 4
## 4 x_2 - 3 x_3 <= 2
## x_1 - 3 x_2 + 2 x_3 <= 3
## x_1, x_3 are non-negative integers
## x_2 is a non-negative real number

obj <- c(3, 1, 3)
mat <- matrix(c(-1, 0, 1, 2, 4, -3, 1, -3, 2), nrow = 3)
dir <- c("<=", "<=", "<=")
rts.test

Test for returns-to-scale in data envelopment analysis

Description
Tests for returns-to-scale in input- and output-oriented DEA models

Usage
rts.test(X, Y, W=NULL, model, H0, bw, B, alpha)

Arguments
X a matrix of inputs for observations, for which DEA scores are estimated.
Y a matrix of outputs for observations, for which DEA scores are estimated.
rts.test

\[ W \]
a matrix of input prices, only used if model="costmin".

\text{model}
a string for the type of DEA model to be estimated, "input" for input-oriented, "output" for output-oriented model, "costmin" for cost-minimization model.

\text{H0}
a string for the NULL hypothesis to be tested, "constant" for constant returns-to-scale, "non-increasing" for non-increasing returns-to-scale.

\text{bw}
a string for the type of bandwidth used as a smoothing parameter in sampling with reflection, "cv" or "bw.ucv" for cross-validation bandwidth, "silverman" or "bw.nrd0" for Silverman’s (1986) rule.

\text{B}
an integer showing the number of bootstrap replications in constructing the test statistics. Recommended is 100.

\text{alpha}
a number in (0,1) for the size of confidence interval for the test statistics.

\text{Details}
Implements Simar and Wilson’s (2002, 2011) returns-to-scale tests for input- and output-oriented DEA models, using ratio of means, mean of ratios or mean of ratios less unity of DEA scores under the null and the alternative hypotheses as test statistics (denoted respectively, statistics 4.5 and 4.6 in Simar and Wilson (2002) and statistics 48 in Simar and Wilson (2011)). Tests the null hypothesis of constant returns to scale vs. the alternative hypothesis of variable returns to scale, and the null hypothesis of non-increasing returns to scale vs. the alternative hypothesis of variable returns to scale.

\text{Value}
A list containing the results of returns-to-scale test, with the following components.

\text{w_hat}
the vector for the values of the test statistics 4.5.

\text{w48_hat}
the vector for the values of the test statistics 48 or statistics 4.6.

\text{theta_H0_hat}
the vector for the values of naive DEA scores under the Null hypothesis (H0).

\text{theta_vrs_hat}
the vector for the values of naive DEA scores under the alternative hypothesis.

\text{w_hat_boot}
the vector for bootstrap values of the test statistics 4.5.

\text{pvalue}
the \text{p}
\text{value} for rejecting the Null hypothesis, according to test statistics 4.5.

\text{pvalue48}
the \text{p}
\text{value} for rejecting the Null hypothesis, according to test statistics 4.6 and 48.

\text{H0reject}
FALSE if H0 is not rejected according to \text{pvalue45} and confidence level alpha, TRUE otherwise.

\text{H0reject48}
FALSE if H0 is not rejected according to \text{pvalue48} and confidence level alpha, TRUE otherwise.

\text{H0level}
the cut-off value of w45_hat_boot for confidence level alpha.

\text{H0level48}
the cut-off value of w48_hat_boot for confidence level alpha.

\text{H0}
the Null hypothesis tested.

\text{bw}
the type of bandwidth employed.

\text{bw_value}
the value for the bandwidth.
Author(s)
Jaak Simm, Galina Besstremyannaya

References


See Also
dea, dea.robust, dea.env.robust and hospitals.

Examples
## load data on Japanese hospitals (Besstremyannaya 2013, 2011)
data("hospitals", package="rDEA")
firms = 1:50
Y = hospitals$firms[, c('inpatients', 'outpatients')] 
X = hospitals$firms[, c('labor', 'capital')] 

## Returns-to-scale test in the input-oriented DEA model, 
## Testing the null hypothesis of constant returns-to-scale 
## vs. an alternative of variable returns-to-scale 
rts_input = rts.test(X=X, Y=Y, W=NULL, model="input", H0="constant", bw="cv", B=100, alpha=0.05) 

tests_input$pvalue 
rt$tests_input$H0reject 
rt$tests_input$H0level

## Test for cost-efficiency DEA model, 
## Testing the null hypothesis of constant returns-to-scale 
## vs. an alternative of variable returns-to-scale 
W = hospitals[firms[, c('labor_price', 'capital_price')] 

rts_cost = rts.test(X=X, Y=Y, W=W, model="costmin", H0="constant", "cv")
bw="cv", B=100, alpha=0.05)

rts.cost$pvalue
rts.cost$H0reject
rts.cost$H0level
Index

*Topic **DEA, non-parametric**
  efficiency, technical
  efficiency, cost efficiency
  dea, 2
  dea.env.robust, 4
  dea.robust, 6
  rts.test, 12
*Topic **optimize**
  multi_glpk_solve_LP, 9
  dea, 2, 5, 8, 14
  dea.env.robust, 3, 4, 8, 14
  dea.robust, 3, 5, 6, 14
  hospitals, 3, 5, 8, 8, 14
  multi_glpk_solve_LP, 9
  Rglpk_solve, 11
  rts.test, 12