Package 'rcDEA'

October 14, 2022

Title Robust and Conditional Data Envelopment Analysis (DEA)

Version 1.0

Maintainer Anna Mergoni <anna.mergoni@kuleuven.be>

Description With this package we provide an easy method to compute robust and conditional Data Envelopment Analysis (DEA),
Free Disposal Hull (FDH) and Benefit of the Doubt (BOD) scores.
The robust approach is based on the work of Cazals, Florens and Simar (2002) <doi:10.1016/S0304-4076(01)00080-X>.
The conditional approach is based on Daraio and Simar (2007) <doi:10.1007/s11123-007-0049-3>.
Besides we provide graphs to help with the choice of m.
We relay on the 'Benchmarking' package to compute the efficiency scores and on the 'np' package to compute non parametric estimation of similarity among units.

License GPL-2

Imports np , Benchmarking

Encoding UTF-8

RoxygenNote 7.1.2.9000

NeedsCompilation no

Author Anna Mergoni [aut, cre] (<https://orcid.org/0000-0002-9128-5341>)

Repository CRAN

Date/Publication 2021-12-14 08:20:02 UTC

R topics documented:

`conditional_BOD` *Conditional BOD function*

Description

This function allows to compute Robust and Conditional BOD scores.

Usage

```
conditional_BOD(
  output,
  exogenous = FALSE,
  m,
  B,
  alpha = FALSE,
  RTS = "CRS",
  ORIENTATION = "in",
  similarity = FALSE,
  inclusion = FALSE,
  print = FALSE
)
```

Arguments

`output`	matrix (or vector) of indicators along which the units are evaluated.
`exogenous`	matrix (or vector) of exogenous variables involved in the conditional analysis. The similarity among the units is determined according to the exogeneous variable(s) using the function npudensbw and npudens (from the package np) with epanechnikov kernel.
`m`	number of unit to be included in the reference set
`B`	number of bootstrap replicates
`alpha`	This allow to choose the size of the Confidence Intervals computed. By defaulta alpha = FALSE. In this case no confidence interval are computed
`RTS`	Default = "CRS". For more details see the dea function in the package Benchmarking. Text string or a number defining the underlying DEA technology / returns to scale assumption. 0 fdh Free disposability hull, no convexity assumption 1 vrs Variable returns to scale, convexity and free disposability 2 drs Decreasing returns to scale, convexity, down-scaling and free disposability 3 crs Constant returns to scale, convexity and free disposability 4 irs Increasing returns to scale, (up-scaling, but not down-scaling), convexity and free disposability 5 irs2 Increasing returns to scale (up-scaling, but not down-scaling), additivity, and free disposability 6 add Additivity (scaling up and down, but only with integers), and free disposability; also known af replicability and free disposability, the free disposability and replicability hull (frh) – no convexity assumption 7 fdh+ A combination of free disposability and restricted or local constant return to scale 10 vrs+ As vrs, but with restrictions on the individual lambdas via param

`ORIENTATION` Default = "in". For more details see the dea function in the package Benchmarking. Input efficiency "in" (1), output efficiency "out" (2), and graph efficiency "graph" (3). For use with DIRECT, an additional option is "in-out" (0).

`similarity` matrix of similarities. In alternative to provide the exogenous variables, the matrix of similarities can be directly provided. This allow to customize the estimation of the similarities.

`inclusion` If inclusion = TRUE the unit under analysis is included in the reference set. So, no super efficient scores are allowed. By default inclusion = FALSE.

`print` If print = TRUE the number of the unit under evaluation is printed. In case of large sample the function could require some time, so it could be useful to control how many units have already been evaluated and which one still have to be evaluated. By default print = FALSE.

Value

If the parameter alpha is specified, the function returns a data frame with three numeric columns. The first column is the vector representing the conditional BOD scores (eff); the second column is the vector representing the lower bound of the condifence interval (ci_low); the third column is the vector representing the upper bound of the confidence interval (Ci_up). If alpha is not specified, the functions returns only the first column of the data frame (eff).

Examples

```
#Example with a very small sample to decrease computational time
         y1 <-runif(50, 50, 75)
         y2 <-runif(50, 30, 75)
         y <- cbind(y1, y2)
         z <- ifelse(rnorm(50, 0, 1)>0, 1, 0)

         #Conditional BOD
         c_BOD <- conditional_BOD(output = y, exogenous = z,
                                  m = 30, B = 50)
         summary(c_BOD$eff)

 #Example with bigger sample
         y1 <-runif(100, 50, 75)
         y2 <-runif(100, 30, 75)
         y <- cbind(y1, y2)
         z <- ifelse(rnorm(100, 0, 1)>0, 1, 0)

         #Conditional BOD
         c_BOD <- conditional_BOD(output = y, exogenous = z,
                                  similarity = FALSE,
                                  m = 30, B = 50)
         summary(c_BOD$eff)
```

`conditional_DEA` *Conditional DEA function*

Description

This function allows to compute Robust and Conditional DEA scores.

Usage

```
conditional_DEA(
  input,
  output,
  exogenous = FALSE,
  alpha = FALSE,
  m,
  B,
  RTS = "crs",
  ORIENTATION = "in",
  similarity = FALSE,
  inclusion = FALSE,
  print = FALSE
)
```

Arguments

`input`	matrix (or vector) of inputs along which the units are evaluated.
`output`	matrix (or vector) of outputs along which the units are evaluated.
`exogenous`	matrix (or vector) of exogenous variables involved in the conditional analysis. The similarity among the units is determined according to the exogeneous variable(s) using the function npudensbw and npudens (from the package np) with epanechnikov kernel.
`alpha`	This allow to choose the size of the Confidence Intervals computed. By defaulta alpha = FALSE. In this case no confidence interval are computed
`m`	number of unit to be included in the reference set
`B`	number of bootstrap replicates
`RTS`	For more details see the dea function in the package Benchmarking. Text string or a number defining the underlying DEA technology / returns to scale assumption. 0 fdh Free disposability hull, no convexity assumption 1 vrs Variable returns to scale, convexity and free disposability 2 drs Decreasing returns to scale, convexity, down-scaling and free disposability 3 crs Constant returns to scale, convexity and free disposability 4 irs Increasing returns to scale, (up-scaling, but not down-scaling), convexity and free disposability 5 irs2 Increasing returns to scale (up-scaling, but not down-scaling), additivity, and free disposability 6 add Additivity (scaling up and down, but only with integers), and free disposability; also known af replicability and free disposability, the free disposability and

replicability hull (frh) – no convexity assumption 7 fdh+ A combination of free disposability and restricted or local constant return to scale 10 vrs+ As vrs, but with restrictions on the individual lambdas via param

`ORIENTATION` For more details see the dea function in the package Benchmarking. Input efficiency "in" (1), output efficiency "out" (2), and graph efficiency "graph" (3). For use with DIRECT, an additional option is "in-out" (0).

`similarity` matrix of similarities. In alternative to provide the exogenous variables, the matrix of similarities can be directly provided. This allow to customize the estimation of the similarities.

`inclusion` If inclusion = TRUE the unit under analysis is included in the reference set. So, no super efficient scores are allowed. By default inclusion = FALSE.

`print` If print = TRUE the number of the unit under evaluation is printed. In case of large sample the function could require some time, so it could be useful to control how many units have already been evaluated and which one still have to be evaluated. By default print = FALSE.

Value

If the parameter alpha is specified, the function returns a data frame with three numeric columns. The first column is the vector representing the conditional DEA scores (eff); the second column is the vector representing the lower bound of the condifence interval (ci_low); the third column is the vector representing the upper bound of the confidence interval (Ci_up). If alpha is not specified, the functions returns only the first column of the data frame (eff).

Examples

```
#Example with a very small sample to decrease computational time.
          x1 <-runif(50, 50, 75)
          x2 <-runif(50, 30, 75)
          x <- cbind(x1, x2)
          e <- rnorm(50, 0, 36)
          a1 <- 0.4
          a2 <- 0.6
          y <- a1*x1 + a2*x2 + e
          z <- ifelse(rnorm(50, 0, 1)>0, 1, 0)

          #Conditional DEA
          c_DEA <- conditional_DEA(input = x, output = y, exogenous = z,
                                   m = 30, B = 50,
                                   RTS = "crs", ORIENTATION = "in")
          summary(c_DEA$eff)

          #Example with bigger sample
          x1 <-runif(100, 50, 75)
          x2 <-runif(100, 30, 75)
          x <- cbind(x1, x2)
          a1 <- 0.4
          a2 <- 0.6
          y <- a1*x1 + a2*x2
          z <- ifelse(rnorm(100, 0, 1)>0, 1, 0)
```

```
#Conditional DEA
c_DEA <- conditional_DEA(input = x, output = y, exogenous = z,
                         m = 30, B = 50,
                         RTS = "crs", ORIENTATION = "in")
summary(c_DEA$eff)
```

---

graph1_m_BOD *Graph to select m*

Description

This function allows to draw a graph that relates the number of super efficient units and the choice of m

Usage

```
graph1_m_BOD(
  output,
  mseries,
  B,
  RTS = "crs",
  ORIENTATION = "in",
  check = c(1),
  col = c("black"),
  print = TRUE
)
```

Arguments

output	matrix (or vector) of indicators along which the units are evaluated.
mseries	vector containing the different values of f that needed to be tested.
B	number of bootstrap replicates
RTS	For more details see the dea function in the package Benchmarking. Text string or a number defining the underlying DEA technology / returns to scale assumption. 0 fdh Free disposability hull, no convexity assumption 1 vrs Variable returns to scale, convexity and free disposability 2 drs Decreasing returns to scale, convexity, down-scaling and free disposability 3 crs Constant returns to scale, convexity and free disposability 4 irs Increasing returns to scale, (up-scaling, but not down-scaling), convexity and free disposability 5 irs2 Increasing returns to scale (up-scaling, but not down-scaling), additivity, and free disposability 6 add Additivity (scaling up and down, but only with integers), and free disposability; also known af replicability and free disposability, the free disposability and replicability hull (frh) – no convexity assumption 7 fdh+ A combination of free disposability and restricted or local constant return to scale 10 vrs+ As vrs, but with restrictions on the individual lambdas via param

ORIENTATION
: For more details see the dea function in the package Benchmarking. Input efficiency "in" (1), output efficiency "out" (2), and graph efficiency "graph" (3). For use with DIRECT, an additional option is "in-out" (0).

check
: vector containing the values of the thresholds to be considered to define the superefficient units

col
: vector containing the colors. the vector col must contain the same number of element of the vector check.

print
: If print = TRUE the number of the unit under evaluation is printed. In case of large sample the function could require some time, so it could be useful to control how many units have already been evaluated and which one still have to be evaluated. By default print = FALSE.

Value

This function return a plot, representing the percentage of super-efficient units for the different values of m. A unit is defined as super-efficient if it gets a value higher than a certain treshold (normally 1) in the robust analysis. Each line of the plot represent different values of the tresholds.

Examples

```
#Example with a very small sample to decrease computational time.
y1 <-runif(20, 50, 75)
y2 <-runif(20, 30, 75)
y <- cbind(y1, y2)

check <- c(1, 1.05, 1.5)
colors <- c("black", "red", "blue")

graph1_m_BOD(output = y, mseries = c(5, 10, 15),
             B = 50, RTS = "crs", ORIENTATION = "in",
             check = check, col = colors)

 #An example with a larger sample size.
 x1 <-runif(100, 50, 75)
 x2 <-runif(100, 30, 75)
 x <- cbind(x1, x2)
 y <- cbind(x+runif(100, -10, 0), rnorm(100, 15, 4))

graph1_m_BOD(output = y,
       mseries = c(20, 30, 40, 50, 60, 70, 80),
       B = 50,
       RTS = "crs", ORIENTATION = "in",
       check = c(1, 1.05, 1.2, 1.5),
       col = c("black", "red", "blue", "green"))
```

`graph1_m_DEA` *Graph to select m*

Description

This function allows to draw a graph that relates the number of super efficient units and the choice of m

Usage

```
graph1_m_DEA(
  input,
  output,
  mseries,
  B,
  RTS = "crs",
  ORIENTATION = "in",
  check = c(1),
  col = c("black"),
  print = TRUE
)
```

Arguments

`input`	matrix (or vector) of inputs along which the units are evaluated.
`output`	matrix (or vector) of outputs along which the units are evaluated.
`mseries`	vector containing the different values of f that needed to be tested.
`B`	number of bootstrap replicates
`RTS`	For more details see the dea function in the package Benchmarking. Text string or a number defining the underlying DEA technology / returns to scale assumption. 0 fdh Free disposability hull, no convexity assumption 1 vrs Variable returns to scale, convexity and free disposability 2 drs Decreasing returns to scale, convexity, down-scaling and free disposability 3 crs Constant returns to scale, convexity and free disposability 4 irs Increasing returns to scale, (up-scaling, but not down-scaling), convexity and free disposability 5 irs2 Increasing returns to scale (up-scaling, but not down-scaling), additivity, and free disposability 6 add Additivity (scaling up and down, but only with integers), and free disposability; also known af replicability and free disposability, the free disposability and replicability hull (frh) – no convexity assumption 7 fdh+ A combination of free disposability and restricted or local constant return to scale 10 vrs+ As vrs, but with restrictions on the individual lambdas via param
`ORIENTATION`	For more details see the dea function in the package Benchmarking. Input efficiency "in" (1), output efficiency "out" (2), and graph efficiency "graph" (3). For use with DIRECT, an additional option is "in-out" (0).
`check`	vector containing the values of the thresholds to be considered to define the superefficient units

`col` vector containing the colors. the vector col must contain the same number of element of the vector check.

`print` If print = TRUE the number of the unit under evaluation is printed. In case of large sample the function could require some time, so it could be useful to control how many units have already been evaluated and which one still have to be evaluated. By default print = FALSE.

Value

This function return a plot, representing the percentage of super-efficient units for the different values of m. A unit is defined as super-efficient if it gets a value higher than a certain treshold (normally 1) in the robust analysis. Each line of the plot represent different values of the tresholds.

Examples

```
#Example with a very small sample to decrease computational time.
x1 <-runif(20, 50, 75)
x2 <-runif(20, 30, 75)
x <- cbind(x1, x2)
e <- rnorm(20, 0, 36)
a1 <- 0.4
a2 <- 0.6
y <- a1*x1 + a2*x2 + e

check <- c(1, 1.05, 1.5)
colors <- c("black", "red", "blue")

graph1_m_DEA(input = x, output = y, mseries = c(5, 10, 15, 20),
             B = 50, RTS = "crs", ORIENTATION = "in",
             check = check, col = colors)



#An example with a larger sample size.
x1 <-runif(100, 50, 75)
x2 <-runif(100, 30, 75)
x <- cbind(x1, x2)
y <- cbind(x+runif(100, -10, 0), rnorm(100, 15, 4))

check <- c(1, 1.05, 1.2, 1.5)
colors <- c("black", "red", "blue", "green")

graph1_m_DEA(input = x, output = y, mseries = c(20, 30, 40, 50, 60, 70, 80),
             B = 50, RTS = "crs", ORIENTATION = "in",
             check = check,
             col = colors)
```

`graph2_m_BOD` *Graph to select m*

Description

This function allows to draw a graph that relates the average efficiency score and the choice of m

Usage

```
graph2_m_BOD(output, mseries, B, RTS = "crs", ORIENTATION = "in", print = TRUE)
```

Arguments

`output`	matrix (or vector) of indicators along which the units are evaluated.
`mseries`	vector containing the different values of m that needed to be tested.
`B`	number of bootstrap replicates
`RTS`	For more details see the dea function in the package Benchmarking. Text string or a number defining the underlying DEA technology / returns to scale assumption. 0 fdh Free disposability hull, no convexity assumption 1 vrs Variable returns to scale, convexity and free disposability 2 drs Decreasing returns to scale, convexity, down-scaling and free disposability 3 crs Constant returns to scale, convexity and free disposability 4 irs Increasing returns to scale, (up-scaling, but not down-scaling), convexity and free disposability 5 irs2 Increasing returns to scale (up-scaling, but not down-scaling), additivity, and free disposability 6 add Additivity (scaling up and down, but only with integers), and free disposability; also known af replicability and free disposability, the free disposability and replicability hull (frh) – no convexity assumption 7 fdh+ A combination of free disposability and restricted or local constant return to scale 10 vrs+ As vrs, but with restrictions on the individual lambdas via param
`ORIENTATION`	For more details see the dea function in the package Benchmarking. Input efficiency "in" (1), output efficiency "out" (2), and graph efficiency "graph" (3). For use with DIRECT, an additional option is "in-out" (0).
`print`	If print = TRUE the number of the unit under evaluation is printed. In case of large sample the function could require some time, so it could be useful to control how many units have already been evaluated and which one still have to be evaluated. By default print = FALSE.

Value

This function return a plot representing the average score from the robust analysis for the different values of m chosen.

Examples

```
#Example with a very small sample to decrease computational time.
y1 <-runif(20, 50, 75)
y2 <-runif(20, 30, 75)
y <- cbind(y1, y2)

graph2_m_BOD(output = y, mseries = c(5, 10, 15), B = 50,
             RTS = "crs", ORIENTATION = "in")



#An example with a larger sample size.
y1 <-runif(100, 50, 75)
y2 <-runif(100, 30, 75)
y <- cbind(y1, y2)

graph2_m_BOD(output = y,
             mseries = c(20, 30, 40, 50, 60, 70, 80),
             B = 50, RTS = "crs", ORIENTATION = "in")
```

---

graph2_m_DEA   *Graph to select m*

Description

This function allows to draw a graph that relates the average efficiency score and the choice of m

Usage

```
graph2_m_DEA(
  input,
  output,
  mseries,
  B,
  RTS = "crs",
  ORIENTATION = "in",
  print = TRUE
)
```

Arguments

input	matrix (or vector) of inputs along which the units are evaluated.
output	matrix (or vector) of outputs along which the units are evaluated.
mseries	vector containing the different values of m that needed to be tested.
B	number of bootstrap replicates

RTS	For more details see the dea function in the package Benchmarking. Text string or a number defining the underlying DEA technology / returns to scale assumption. 0 fdh Free disposability hull, no convexity assumption 1 vrs Variable returns to scale, convexity and free disposability 2 drs Decreasing returns to scale, convexity, down-scaling and free disposability 3 crs Constant returns to scale, convexity and free disposability 4 irs Increasing returns to scale, (up-scaling, but not down-scaling), convexity and free disposability 5 irs2 Increasing returns to scale (up-scaling, but not down-scaling), additivity, and free disposability 6 add Additivity (scaling up and down, but only with integers), and free disposability; also known af replicability and free disposability, the free disposability and replicability hull (frh) – no convexity assumption 7 fdh+ A combination of free disposability and restricted or local constant return to scale 10 vrs+ As vrs, but with restrictions on the individual lambdas via param
ORIENTATION	For more details see the dea function in the package Benchmarking. Input efficiency "in" (1), output efficiency "out" (2), and graph efficiency "graph" (3). For use with DIRECT, an additional option is "in-out" (0).
print	If print = TRUE, the number of the unit under evaluation is printed. In case of large sample the function could require some time, so it could be useful to control how many units have already been evaluated and which one still have to be evaluated. By default print = FALSE.

Value

This function return a plot representing the average score from the robust analysis for the different values of m chosen.

Examples

```
#Example with a very small sample to decrease computational time.
x1 <-runif(20, 50, 75)
x2 <-runif(20, 30, 75)
x <- cbind(x1, x2)
e <- rnorm(20, 0, 36)
a1 <- 0.4
a2 <- 0.6
y <- a1*x1 + a2*x2 + e

graph2_m_DEA(input = x, output = y,  mseries = c(5, 10, 15, 20),
             B = 50, RTS = "crs", ORIENTATION = "in")


#An example with a larger sample size.
x1 <-runif(100, 50, 75)
x2 <-runif(100, 30, 75)
x <- cbind(x1, x2)
y <- cbind(x+runif(100, -10, 0), rnorm(100, 15, 4))

graph2_m_DEA(input = x, output = y,
             mseries = c(20, 30, 40, 50, 60, 70, 80), B = 50,
             RTS = "crs", ORIENTATION = "in")
```

robust_BOD *Robust BOD function*

Description

This function allows to compute Robust BOD scores.

Usage

```
robust_BOD(
  output,
  m,
  B,
  alpha = FALSE,
  RTS = "CRS",
  ORIENTATION = "in",
  inclusion = FALSE,
  print = FALSE
)
```

Arguments

output	matrix (or vector) of indicators along which the units are evaluated.
m	number of unit to be included in the reference set
B	number of bootstrap replicates
alpha	This allow to choose the size of the Confidence Intervals computed. By defaulta alpha = FALSE. In this case no confidence interval are computed
RTS	Default = "CRS". For more details see the dea function in the package Benchmarking. Text string or a number defining the underlying DEA technology / returns to scale assumption. 0 fdh Free disposability hull, no convexity assumption 1 vrs Variable returns to scale, convexity and free disposability 2 drs Decreasing returns to scale, convexity, down-scaling and free disposability 3 crs Constant returns to scale, convexity and free disposability 4 irs Increasing returns to scale, (up-scaling, but not down-scaling), convexity and free disposability 5 irs2 Increasing returns to scale (up-scaling, but not down-scaling), additivity, and free disposability 6 add Additivity (scaling up and down, but only with integers), and free disposability; also known af replicability and free disposability, the free disposability and replicability hull (frh) – no convexity assumption 7 fdh+ A combination of free disposability and restricted or local constant return to scale 10 vrs+ As vrs, but with restrictions on the individual lambdas via param
ORIENTATION	Default = "in". For more details see the dea function in the package Benchmarking. Input efficiency "in" (1), output efficiency "out" (2), and graph efficiency "graph" (3). For use with DIRECT, an additional option is "in-out" (0).

`inclusion` If inclusion = TRUE the unit under analysis is included in the reference set. So, no super efficient scores are allowed. By default inclusion = FALSE.

`print` If print = TRUE the number of the unit under evaluation is printed. In case of large sample the function could require some time, so it could be useful to control how many units have already been evaluated and which one still have to be evaluated. By default print = FALSE.

Value

If the parameter alpha is specified, the function returns a data frame with three numeric columns. The first column is the vector representing the robust BOD scores (eff); the second column is the vector representing the lower bound of the condifence interval (ci_low); the third column is the vector representing the upper bound of the confidence interval (Ci_up). If alpha is not specified, the functions returns only the first column of the data frame (eff).

Examples

```
#Example with a very small sample to decrease computational time.
          y1 <-runif(50, 50, 75)
          y2 <-runif(50, 30, 75)
          y <- cbind(y1, y2)

          #Robust BOD
          r_BOD <- robust_BOD(output = y, m = 30, B = 50,
                              RTS = "crs", ORIENTATION = "in", print = TRUE)
          summary(r_BOD$eff)

          ## Not run: #Example with random data x and y
          y1 <-runif(100, 50, 75)
          y2 <-runif(100, 30, 75)
          y <- cbind(y1, y2)

          #Robust BOD
          r_BOD <- robust_BOD(output = y, m = 30, B = 50,
                              RTS = "crs", ORIENTATION = "in", print = TRUE)
          summary(r_BOD$eff)
## End(Not run)
```

---

robust_DEA *Robust Data Envelopment Analysis (DEA)*

Description

This function allows to compute Robust DEA scores.

Usage

```
robust_DEA(
  input,
  output,
  m,
  B,
  RTS = "crs",
  ORIENTATION = "in",
  alpha = FALSE,
  inclusion = FALSE,
  print = FALSE
)
```

Arguments

`input`	matrix (or vector) of inputs along which the units are evaluated.
`output`	matrix (or vector) of outputs along which the units are evaluated.
`m`	number of unit to be included in the reference set
`B`	number of bootstrap replicates
`RTS`	For more details see the dea function in the package Benchmarking. Text string or a number defining the underlying DEA technology / returns to scale assumption. 0 fdh Free disposability hull, no convexity assumption 1 vrs Variable returns to scale, convexity and free disposability 2 drs Decreasing returns to scale, convexity, down-scaling and free disposability 3 crs Constant returns to scale, convexity and free disposability 4 irs Increasing returns to scale, (up-scaling, but not down-scaling), convexity and free disposability 5 irs2 Increasing returns to scale (up-scaling, but not down-scaling), additivity, and free disposability 6 add Additivity (scaling up and down, but only with integers), and free disposability; also known af replicability and free disposability, the free disposability and replicability hull (frh) – no convexity assumption 7 fdh+ A combination of free disposability and restricted or local constant return to scale 10 vrs+ As vrs, but with restrictions on the individual lambdas via param
`ORIENTATION`	For more details see the dea function in the package Benchmarking. Input efficiency "in" (1), output efficiency "out" (2), and graph efficiency "graph" (3). For use with DIRECT, an additional option is "in-out" (0).
`alpha`	This allow to choose the size of the Confidence Intervals computed. By defaulta alpha = FALSE. In this case no confidence interval are computed
`inclusion`	If inclusion = TRUE the unit under analysis is included in the reference set. So, no super efficient scores are allowed. By default inclusion = FALSE.
`print`	If print = TRUE the number of the unit under evaluation is printed. In case of large sample the function could require some time, so it could be useful to control how many units have already been evaluated and which one still have to be evaluated. By default print = FALSE.

Value

If the parameter alpha is specified, the function returns a data frame with three numeric columns. The first column is the vector representing the robust DEA scores (eff); the second column is the vector representing the lower bound of the condifence interval (ci_low); the third column is the vector representing the upper bound of the confidence interval (Ci_up). If alpha is not specified, the functions returns only the first column of the data frame (eff).

Examples

```
#Example with a very small sample to decrease computational time.
         x1 <-runif(50, 50, 75)
         x2 <-runif(50, 30, 75)
         x <- cbind(x1, x2)
         e <- rnorm(50, 0, 36)
         a1 <- 0.4
         a2 <- 0.6
         y <- a1*x1 + a2*x2 + e

         #Robust DEA
         r_DEA <- robust_DEA(input = x, output = y, m = 20, B = 50,
         RTS = "crs", ORIENTATION = "in", print = TRUE)
         summary(r_DEA$eff)

 #Example with random data x and y
         x1 <-runif(100, 50, 75)
         x2 <-runif(100, 30, 75)
         x <- cbind(x1, x2)
         y <- cbind(x+runif(100, -10, 0), rnorm(100, 15, 4))

         #Robust DEA
         r_DEA <- robust_DEA(input = x, output = y, m = 30, B = 40,
         RTS = "crs", ORIENTATION = "in", print = TRUE)
         summary(r_DEA$eff)
```

Index