Package ‘DJL’

April 30, 2019

Type Package

Title Distance Measure Based Judgment and Learning

Version 3.0

Date 2019-05-01

Author Dong-Joon Lim, PhD

Depends R (>= 3.4.0), car, lpSolveAPI

Maintainer Dong-Joon Lim <tgno3.com@gmail.com>

Description Implements various decision support tools related to the Econometrics & Technometrics. Subroutines include correlation reliability test, Mahalanobis distance measure for outlier detection, combinatorial search (all possible subset regression), non-parametric efficiency analysis measures: DDF (directional distance function), DEA (data envelopment analysis), HDF (hyperbolic distance function), SBM (slack-based measure), and SF (shortage function), benchmarking, Malmquist productivity analysis, risk analysis, technology adoption model, new product target setting, etc.

License GPL-2

LazyData true

NeedsCompilation no

Repository CRAN

Date/Publication 2019-04-30 12:40:10 UTC

R topics documented:

`dataset.airplane.2017` *Dataset of commercial airplanes from 1965 to 2017.*

Description

Commercial airplanes from 1965 to 2017.

Usage

```
data(dataset.airplane.2017)
```

Columns

`[,1]`	`Name`	Airplane name
`[,2]`	`EIS`	Entry into service
`[,3]`	`Range`	Maximum range at full payload in 1,000km
`[,4]`	`P.cap`	Passenger capacity
`[,5]`	`PFE`	Passenger fuel efficiency in passengers*km/L (log scale)
`[,6]`	`C.spd`	Cruising speed in km/hr
`[,7]`	`M.spd`	Maximum speed in km/hr

Author(s)

Dong-Joon Lim, PhD

Source

http://www.airbus.com/aircraftfamilies
http://www.boeing.com/commercial

References

Lim, Dong-Joon, and Timothy R. Anderson. Time series benchmarking analysis for new product scheduling: who are the competitors and how fast are they moving forward?. Advances in DEA Theory and Applications: with Examples in Forecasting Models. *Wiley (forthcoming)*, 2016.

Examples

```
# Load dataset
  data(dataset.airplane.2017)
```

`dataset.engine.2015` *Dataset of auto engines from MY2005 to MY2015.*

Description

Auto engines from MY2005 to MY2015.

Usage

```
data(dataset.engine.2015)
```

Columns

`[,1]`	`Name`	Vehicle name
`[,2]`	`MY`	Model year
`[,3]`	`Cylinder`	The number of cylinder
`[,4]`	`Displacement`	Displacement in liter
`[,5]`	`CO2`	CO2 emission in gram/mile
`[,6]`	`Power`	Engine power in HP
`[,7]`	`Torque`	Engine torque in lb.ft
`[,8]`	`Type`	Engine system and fuel type

Author(s)

Dong-Joon Lim, PhD

Source

http://www.fueleconomy.gov
http://www.autoevolution.com

References

D.-J. Lim, Internal combustion engine race: naturally aspirated vs turbo/super-charged, *working paper* (2015).

Examples

```
# Load dataset
  data(dataset.engine.2015)
```

`dataset.hev.2013` *Dataset of hybrid electric vehicles from MY1997 to MY2013.*

Description

Hybrid electric vehicles from MY1997 to MY2013.

Usage

```
data(dataset.hev.2013)
```

Columns

`[,1] Name`	Vehicle name
`[,2] MY`	Model year
`[,3] MSRP.2013`	MSRP converted to 2013 value
`[,4] Acc`	Acceleration (0-100km) in km/h/s
`[,5] MPG`	MPG in mile/gallon
`[,6] MPGe`	MPG equivalence for PHEV in mile/gallon

Author(s)

Dong-Joon Lim, PhD

Source

http://www.fueleconomy.gov

References

Lim, Dong-Joon, et al. "Comparing technological advancement of hybrid electric vehicles (HEV) in different market segments." *Technological Forecasting and Social Change* 97 (2015): 140~153.

Examples

```
# Load dataset
  data(dataset.hev.2013)
```

dm.ddf *Distance measure using DDF*

Description

Implements *Chambers*' directional distance function (non-radial & non-oriented measure).

Usage

```
dm.ddf(xdata, ydata, rts="crs", g=NULL,
       wd=NULL, se=FALSE, sg="ssm", date=NULL, cv="convex", o=NULL)
```

Arguments

`xdata`	Input(s) vector (*n* by *m*)
`ydata`	Output(s) vector (*n* by *s*)
`rts`	Returns to scale assumption `"crs"` Constant RTS (default) `"vrs"` Variable RTS `"irs"` Increasing RTS `"drs"` Decreasing RTS
`g`	Directional vector indicating a measurement direction (*n* by *(m+s)*) By default (`NULL`), `xdata` & `ydata` will be used
`wd`	Weak disposability vector indicating (an) undesirable output(s) (*1* by *s*)
`se`	Implements super-efficiency model alike *Anderson* & *Peterson*'s model if `TRUE`
`sg`	Employs second-stage optimization `"ssm"` Slack-sum maximization (default) `"max"` Date-sum maximization (only if `date` is defined) `"min"` Date-sum minimization (only if `date` is defined)
`date`	Production date (*n* by *1*)
`cv`	Convexity assumption `"convex"` Convexity holds (default) `"fdh"` Free disposal hull (this will override `rts`)
`o`	DMU index to calc. `NULL`(default) will calc for all

Value

`$eff`	Efficiency score
`$lambda`	Intensity vector
`$mu`	Secondary intensity vector for weak disposability under VRS
`$beta`	Input reduction factor
`$gamma`	Output augmentation factor
`$xslack`	Input slack
`$yslack`	Output slack

Author(s)

Dong-Joon Lim, PhD

References

Chambers, Robert G., Yangho Chung, and Rolf Fare. "Profit, directional distance functions, and Nerlovian efficiency." *Journal of optimization theory and applications* 98.2 (1998): 351~364.

Fare, Rolf, and Shawna Grosskopf. "Directional distance functions and slacks-based measures of efficiency." *European journal of operational research* 200.1 (2010): 320~322.

See Also

`dm.ddf` Distance measure using DDF
`dm.dea` Distance measure using DEA
`dm.hdf` Distance measure using HDF
`dm.sbm` Distance measure using SBM
`dm.sf` Distance measure using SF

Examples

```
# Additive form directional distance function
  # ready
  x <- matrix(c(5, 1, 4), ncol = 1)
  y <- matrix(c(8, 3, 5, 6, 4, 1), ncol = 2)
  g <- matrix(c(1), nrow = 3, ncol = 3)
  w <- matrix(c(1, 0), ncol = 2)
  # go
  dm.ddf(x, y, "crs", g, w)

# Multiplicative form directional distance function
  # ready
  g <- cbind(x, y)
  # go
  dm.ddf(x, y, "crs", g, w)
```

---

dm.dea *Distance measure using DEA*

Description

Implements *Charnes & Cooper*'s data envelopment analysis (radial & oriented measure).

Usage

```
dm.dea(xdata, ydata, rts="crs", orientation,
       se=FALSE, sg="ssm", date=NULL, ncv=NULL, env=NULL, cv="convex", o=NULL)
```

Arguments

`xdata`	Input(s) vector (*n* by *m*)
`ydata`	Output(s) vector (*n* by *s*)
`rts`	Returns to scale assumption `"crs"` Constant RTS (default) `"vrs"` Variable RTS `"irs"` Increasing RTS `"drs"` Decreasing RTS
`orientation`	Orientation of the measurement `"i"` Input-orientation `"o"` Output-orientation
`se`	Implements *Anderson & Peterson*'s super-efficiency model if `TRUE`
`sg`	Employs second-stage optimization `"ssm"` Slack-sum maximization (default) `"max"` Date-sum maximization (only if `date` is defined) `"min"` Date-sum minimization (only if `date` is defined)
`date`	Production date (*n* by *1*)
`ncv`	Non-controllable variable index(binary) for internal NDF (*1* by *(m+s)*)
`env`	Environment index for external NDF (*n* by *1*)
`cv`	Convexity assumption `"convex"` Convexity holds (default) `"fdh"` Free disposal hull (this will override `rts`)
`o`	DMU index to calc. `NULL`(default) will calc for all

Value

`$eff`	Efficiency score
`$lambda`	Intensity vector
`$xslack`	Input slack
`$yslack`	Output slack
`$vx`	Input (dual) weight
`$uy`	Output (dual) weight
`$w`	Free (dual) variable

Author(s)

Dong-Joon Lim, PhD

References

Charnes, Abraham, William W. Cooper, and Edwardo Rhodes. "Measuring the efficiency of decision making units." *European journal of operational research* 2.6 (1978): 429~444.

Charnes, Abraham, William W. Cooper, and Edwardo Rhodes. "Evaluating program and managerial efficiency: an application of data envelopment analysis to program follow through." *Management science* 27.6 (1981): 668~697.

Banker, Rajiv D., and Richard C. Morey. "Efficiency analysis for exogenously fixed inputs and outputs." *Operations Research* 34.4 (1986): 513~521.

Ruggiero, John. "On the measurement of technical efficiency in the public sector." *European Journal of Operational Research* 90.3 (1996): 553~565.

Fried, Harold O., CA Knox Lovell, and Shelton S. Schmidt, eds. The measurement of productive efficiency and productivity growth. *Oxford University Press*, 2008.

See Also

dm.ddf Distance measure using DDF
dm.dea Distance measure using DEA
dm.hdf Distance measure using HDF
dm.sbm Distance measure using SBM
dm.sf Distance measure using SF

Examples

```
# Reproduce Table 3.9 (p.348) in Fried, H.O. et al.(2008)
  # ready
  X <- data.frame(x1 = c(8, 6,   3,   10, 6,   8,   8, 4),
                  x2 = c(8, 4.6, 1.9, 9,  3.6, 3.6, 9, 1.9))
  Y <- data.frame(y1 = c(8, 5,   2,   9,  4.5, 4.5, 7, 2))
  C <- data.frame(x1 = 0, x2 = 1, y1 = 0)

  # go
  data.frame(ALL_CRS = dm.dea(X, Y, "crs", "i")$eff,
             ALL_VRS = dm.dea(X, Y, "vrs", "i")$eff,
             NDF_CRS = dm.dea(X, Y, "crs", "i", ncv = C)$eff,
             NDF_VRS = dm.dea(X, Y, "vrs", "i", ncv = C)$eff,
             row.names = LETTERS[1 : 8])
```

---

dm.dynamic.bc *Dynamic DEA in the presence of intertemporal Budget Constraints*

Description

Employs the Farrell measure on carry-over budget as well as input or output

Usage

```
dm.dynamic.bc(xdata, ydata, zdata, bdata, rts="crs", orientation="i", wv=NULL)
```

Arguments

`xdata`	Input array (*n* by *m* by *t*)
`ydata`	Output array (*n* by *s* by *t*)
`zdata`	Budget(spent) array (*n* by *b* by *t*)
`bdata`	Budget(secured) array (*n* by *b*)
`rts`	Returns to scale assumption `"crs"` Constant RTS (default) `"vrs"` Variable RTS `"irs"` Increasing RTS `"drs"` Decreasing RTS
`orientation`	Orientation of the measurement `"i"` Input-orientation (default) `"o"` Output-orientation
`wv`	Weight vector for scalarization (*1* by *m* or *s*)

Value

`$eff.s`	System Efficiency
`$eff.t`	Period Efficiency
`$lambda`	Intensity vectors
`$xslack`	Input slack
`$yslack`	Output slack
`$zslack`	Budget(spent) slack
`$aslack`	Budget(available) slack

Author(s)

Dong-Joon Lim, PhD

References

Lim, D.-J., K.-W., Lee, & M.-S., Kim. (2019). "A revised dynamic DEA model with budget constraints" *(Forthcoming)* (TBA).

See Also

`dm.dea` Distance measure using DEA

Examples

```
# Load data
  df.io <- array(c(2, 4, 8, 4, 1, 2, 2, 2, 3, 6, 12, 6,
                   5, 4, 3, 8, 1, 1, 1, 1, 5, 4,  3, 8),
                 c(4, 3, 2),
                 dimnames = list(LETTERS[1:4], c("X", "Y", "z"), c("t1", "t2")))
  df.Z.0 <- array(c(9, 12, 18, 24), c(4, 1), dimnames = list(LETTERS[1:4], c("Z^0")))
```

```
# Run
  dm.dynamic.bc(df.io[,1,], df.io[,2,], df.io[,3,], df.Z.0)
```

---

dm.hdf *Distance measure using HDF*

Description

Implements *Fare*'s hyperbolic distance function (semi-radial & non-oriented measure).

Usage

```
dm.hdf(xdata, ydata, rts="crs",
       wd=NULL, se=FALSE, sg="ssm", date=NULL, cv="convex", o=NULL)
```

Arguments

`xdata`	Input(s) vector (*n* by *m*)
`ydata`	Output(s) vector (*n* by *s*)
`rts`	Returns to scale assumption `"crs"` Constant RTS (default) `"vrs"` Variable RTS `"irs"` Increasing RTS `"drs"` Decreasing RTS
`wd`	Weak disposability vector indicating (an) undesirable output(s) (*1* by *s*)
`se`	Implements super-efficiency model alike *Anderson & Peterson*'s model if `TRUE`
`sg`	Employs second-stage optimization `"ssm"` Slack-sum maximization (default) `"max"` Date-sum maximization (only if `date` is defined) `"min"` Date-sum minimization (only if `date` is defined)
`date`	Production date (*n* by *1*)
`cv`	Convexity assumption `"convex"` Convexity holds (default) `"fdh"` Free disposal hull (this will override `rts`)
`o`	DMU index to calc. `NULL`(default) will calc for all

Value

`$eff`	Efficiency score
`$lambda`	Intensity vector
`$mu`	Secondary intensity vector for weak disposability under VRS
`$xslack`	Input slack
`$yslack`	Output slack
`$iteration`	The number of iteration to obtain the hyperbolic efficiency score

Author(s)

Dong-Joon Lim, PhD

References

Fare, R., Shawna Grosskopf, and CA Knox Lovell. The Measurement of Efficiency of Production. *Boston: Kulwer-Nijhoff* (1985).

Fare, Rolf, et al. "Estimating the hyperbolic distance function: A directional distance function approach." *European Journal of Operational Research* 254.1 (2016): 312~319.

See Also

`dm.ddf` Distance measure using DDF
`dm.dea` Distance measure using DEA
`dm.hdf` Distance measure using HDF
`dm.sbm` Distance measure using SBM
`dm.sf` Distance measure using SF

Examples

```
# Reproduce Table 2 in Fare et al.(2016)
  # ready
  x <- data.frame(x1 = c(2, 4, 9, 6.5, 10, 6, 9))
  y <- data.frame(y1 = c(3, 7, 10, 8.5, 4, 2, 8))

  # go
  sf  <- dm.sf (x, y, "vrs")$eff
  hdf <- dm.hdf(x, y, "vrs")$eff
  matrix(t(cbind(sf, hdf)), 2, 7,
         dimnames = list(c("SF", "HDF"),
                         paste0("DMU_", c(letters[1:4], "o", "p", "q"))))
```

---

dm.mahalanobis — *Distance measure using Mahalanobis distance for outlier detection*

Description

Implements *Mahalanobis* distance measure for outlier detection. In addition to the basic distance measure, boxplots are provided with potential outlier(s) to give an insight into the early stage of data cleansing task.

Usage

```
dm.mahalanobis(data, from="median", p=10, plot=FALSE, v.index=NULL, layout=NULL)
```

Arguments

`data`	Dataframe
`from`	Datum point from which the distance is measured `"mean"` Mean of each column `"median"` Median of each column (default)
`p`	Percentage to which outlier point(s) is noted (default of 10)
`plot`	Switch for boxplot(s)
`v.index`	Numeric vector indicating column(s) to be printed in the boxplot. Default value of `NULL` will present all.
`layout`	Numeric vector indicating dimension of boxplots. Default value of `NULL` will find an optimal layout.

Value

`$dist`	`Mahalanobis` distance from `from`
`$excluded`	Excluded row(s) in row number
`$order`	Distance order (decreasing) in row number
`$suspect`	Potential outlier(s) in row number

Author(s)

Dong-Joon Lim, PhD

References

Hair, Joseph F., et al. Multivariate data analysis. Vol. 7. *Upper Saddle River*, NJ: Pearson Prentice Hall, 2006.

Examples

```
# Generate a sample dataframe
df <- data.frame(replicate(6, sample(0 : 100, 50)))

# go
dm.mahalanobis(df, plot = TRUE)
```

`dm.sbm` *Distance measure using SBM*

Description

Implements *Tone*'s slack-based model (non-radial & (non-)oriented measure).

Usage

```
dm.sbm(xdata, ydata, rts="crs",
       orientation="n", se=FALSE, sg="ssm", date=NULL, cv="convex", o=NULL)
```

Arguments

`xdata`	Input(s) vector (*n* by *m*)
`ydata`	Output(s) vector (*n* by *s*)
`rts`	Returns to scale assumption `"crs"` Constant RTS (default) `"vrs"` Variable RTS `"irs"` Increasing RTS `"drs"` Decreasing RTS
`orientation`	Orientation of the measurement `"n"` Non-orientation (default) `"i"` Input-orientation `"o"` Output-orientation
`se`	Implements super-efficiency model alike *Anderson & Peterson*'s model if `TRUE`
`sg`	Employs second-stage optimization `"ssm"` Slack-sum maximization (default) `"max"` Date-sum maximization (only if `date` is defined) `"min"` Date-sum minimization (only if `date` is defined)
`date`	Production date (*n* by *1*)
`cv`	Convexity assumption `"convex"` Convexity holds (default) `"fdh"` Free disposal hull (this will override `rts`)
`o`	DMU index to calc. `NULL`(default) will calc for all

Value

`$eff`	Efficiency score
`$lambda`	Intensity vector
`$xslack`	Input slack
`$yslack`	Output slack
`$xtarget`	Input target
`$ytarget`	Output target

Author(s)

Dong-Joon Lim, PhD

References

Tone, Kaoru. "A slacks-based measure of efficiency in data envelopment analysis." *European journal of operational research* 130.3 (2001): 498~509.

Tone, Kaoru. "A slacks-based measure of super-efficiency in data envelopment analysis." *European journal of operational research* 143 (2002): 32~41.

See Also

`dm.ddf` Distance measure using DDF
`dm.dea` Distance measure using DEA
`dm.hdf` Distance measure using HDF
`dm.sbm` Distance measure using SBM
`dm.sf` Distance measure using SF

Examples

```
# Reproduce Table 2 in Tone.(2001)
  # ready
  X <- data.frame(x1 = c(4, 6, 8, 8, 2),
                  x2 = c(3, 3, 1, 1, 4))
  Y <- data.frame(y1 = c(2, 2, 6, 6, 1),
                  y2 = c(3, 3, 2, 1, 4))
  # go
  dm.sbm(X, Y)

# Reproduce Table 1 in Tone.(2002)
  # Published input slacks are alternate optima (confirmed by Tone)
  # ready
  X <- data.frame(x1 = c(4, 7, 8, 4, 2, 10, 12),
                  x2 = c(3, 3, 1, 2, 4, 1,  1))
  Y <- data.frame(y1 = c(1, 1, 1, 1, 1, 1,  1))
  # go
  dm.sbm(X, Y, se = TRUE)

# Reproduce Table 4 in Tone.(2002)
  # ready
  X <- data.frame(x1 = c(80,  65,  83,  40,   52,  94),
                  x2 = c(600, 200, 400, 1000, 600, 700),
                  x3 = c(54,  97,  72,  75,   20,  36),
                  x4 = c(8,   1,   4,   7,    3,   5))
  Y <- data.frame(y1 = c(90,  58,  60,  80,   72,  96),
                  y2 = c(5,   1,   7,   10,   8,   6))
  # go
  dm.sbm(X, Y, "crs", "i", se = TRUE)
```

`dm.sf` *Distance measure using SF*

Description

Implements *Luenberger*'s shortage (benefit) function (radial & non-oriented measure).

Usage

```
dm.sf(xdata, ydata, rts="crs", g=NULL,
      wd=NULL, se=FALSE, sg="ssm", date=NULL, cv="convex", o=NULL)
```

Arguments

`xdata`	Input(s) vector (*n* by *m*)
`ydata`	Output(s) vector (*n* by *s*)
`rts`	Returns to scale assumption `"crs"` Constant RTS (default) `"vrs"` Variable RTS `"irs"` Increasing RTS `"drs"` Decreasing RTS
`g`	Directional vector indicating a measurement direction (*n* by *(m+s)*) By default (`NULL`), `xdata` & `ydata` will be used
`wd`	Weak disposability vector indicating (an) undesirable output(s) (*1* by *s*)
`se`	Implements super-efficiency model alike *Anderson & Peterson*'s model if `TRUE`
`sg`	Employs second-stage optimization `"ssm"` Slack-sum maximization (default) `"max"` Date-sum maximization (only if `date` is defined) `"min"` Date-sum minimization (only if `date` is defined)
`date`	Production date (*n* by *1*)
`cv`	Convexity assumption `"convex"` Convexity holds (default) `"fdh"` Free disposal hull (this will override `rts`)
`o`	DMU index to calc. `NULL`(default) will calc for all

Value

`$eff`	Efficiency score
`$lambda`	Intensity vector
`$mu`	Secondary intensity vector for weak disposability under VRS
`$xslack`	Input slack
`$yslack`	Output slack
`$w`	Input (dual) weight
`$p`	Output (dual) weight
`$u`	Free (dual) variable

Author(s)

Dong-Joon Lim, PhD

References

Luenberger, David G. "Benefit functions and duality." *Journal of mathematical economics* 21.5 (1992): 461~481.

Chambers, Robert G., Yangho Chung, and Rolf Fare. "Profit, directional distance functions, and Nerlovian efficiency." *Journal of optimization theory and applications* 98.2 (1998): 351~364.

See Also

`dm.ddf` Distance measure using DDF
`dm.dea` Distance measure using DEA
`dm.hdf` Distance measure using HDF
`dm.sbm` Distance measure using SBM
`dm.sf` Distance measure using SF

Examples

```
# Additive form shortage function
  # ready
  x <- matrix(c(5, 1, 4), ncol = 1)
  y <- matrix(c(8, 3, 5, 6, 4, 1), ncol = 2)
  g <- matrix(c(1), nrow = 3, ncol = 3)
  w <- matrix(c(1, 0), ncol = 2)
  # go
  dm.sf(x, y, "crs", g, w)

# Multiplicative form shortage function
  # ready
  g <- cbind(x, y)
  # go
  dm.sf(x, y, "crs", g, w)
```

---

ma.aps.reg — *Combinatorial search (all possible subset) for regression analysis*

Description

Implements combinatorial (exhaustive) search algorithm, aka all-possible-subsets regression. As opposed to the sequential approach (stepwise, forward addition, or backward elimination) that has a potential bias resulting from considering only one variable for selection at a time, all possible combinations of the independent variables are examined, and sets satisfying designated conditions are returned.

Usage

```
ma.aps.reg(dv, iv, min=1, max, mad=FALSE, aic=FALSE, bic=FALSE,
           model.sig=TRUE, coeff.sig=TRUE, coeff.vif=TRUE, coeff.cor=FALSE)
```

Arguments

`dv`	Dependent variable (*r* by *1*)
`iv`	Independent variable(s) (*r* by *c*)
`min`	Minimum number of independent variable to explore (>=*1*)
`max`	Maximum number of independent variable to explore (<=*r*/10)
`mad`	Returns mean absolute deviation when `TRUE`
`aic`	Returns Akaike's information criterion when `TRUE`
`bic`	Returns Bayesian information criterion when `TRUE`
`model.sig`	Returns models statistically significant only when `TRUE`
`coeff.sig`	Returns models with statistically significant coefficients only when `TRUE`
`coeff.vif`	Returns models with allowable level of multicollinearity only when `TRUE`
`coeff.cor`	Returns models without suppression effects only when `TRUE`

Author(s)

Dong-Joon Lim, PhD

References

Hair, Joseph F., et al. Multivariate data analysis. Vol. 7. *Upper Saddle River*, NJ: Pearson Prentice Hall, 2006.

Examples

```
# Load airplane dataset
df <- dataset.airplane.2017

# ready
dv <- subset(df, select = 2)
iv <- subset(df, select = 3 : 7)

# go
ma.aps.reg(dv, iv, 1, 3, mad = TRUE, coeff.cor = TRUE)
```

`map.corr` *Correlation mapping for reliability test*

Description

Implements a series of correlation analysis by dropping extreme data points one by one using *Mahalanobis* distance measure. Correlation reliability can be investigated with identified anchoring point(s). Correlation map as well as summary table is provided.

Usage

```
map.corr(data, from = "median", threshold = 0.3, r.name = FALSE)
```

Arguments

data	Dataframe
from	Datum point from which the distance is measured "mean" Mean of each column "median" Median of each column (default)
threshold	Threshold of correlation change to be noted on the map
r.name	Dropped points are shown in row name when TRUE

Value

$reliability	Summary table

Author(s)

Dong-Joon Lim, PhD

See Also

dm.mahalanobis Distance measure using Mahalanobis distance

Examples

```
# Generate a sample dataframe
df <- data.frame(replicate(2, sample(0 : 100, 50)))

# go
map.corr(df)
```

---

map.soa.ddf *SOA mapping using DDF*

Description

Employs dm.ddf over time to generate a state-of-the-art map.

Usage

```
map.soa.ddf(xdata, ydata, date,
            rts="crs", g=NULL, wd=NULL, sg="ssm", cv="convex", mk="dmu")
```

Arguments

xdata	Input(s) vector (*n* by *m*)
ydata	Output(s) vector (*n* by *s*)
date	Production date (*n* by *1*)
rts	Returns to scale assumption "crs" Constant RTS (default) "vrs" Variable RTS "irs" Increasing RTS "drs" Decreasing RTS
g	Directional vector indicating a measurement direction (*n* by *(m+s)*) By default (NULL), xdata & ydata will be used
wd	Weak disposability vector indicating (an) undesirable output(s) (*1* by *s*)
sg	Employs second-stage optimization "ssm" Slack-sum maximization (default) "max" Date-sum maximization (only if date is defined) "min" Date-sum minimization (only if date is defined)
cv	Convexity assumption "convex" Convexity holds (default) "fdh" Free disposal hull (this will override rts)
mk	Marker on the map "dmu" DMU index (default) "eff" Efficiency score

Author(s)

Dong-Joon Lim, PhD

See Also

map.soa.ddf SOA mapping using DDF
map.soa.dea SOA mapping using DEA
map.soa.hdf SOA mapping using HDF
map.soa.sbm SOA mapping using SBM
map.soa.sf SOA mapping using SF

Examples

```
# Load engine dataset
  df <- dataset.engine.2015

# Subset for forced induction systems
  fis <- subset(df, grepl("^.C..", df[, 8]))

# Parameters
  x <- subset(fis, select = 4)
  y <- subset(fis, select = 6 : 7)
  d <- subset(fis, select = 2)
```

```
  g <- matrix(c(1), nrow = nrow(x), ncol = 3)

# Generate an SOA map
  map.soa.ddf(x, y, d, "crs", g)
```

---

map.soa.dea *SOA mapping using DEA*

Description

Employs dm.dea over time to generate a state-of-the-art map.

Usage

```
map.soa.dea(xdata, ydata, date, rts="crs", orientation,
            sg="ssm", ncv=NULL, env=NULL, cv="convex", mk="dmu")
```

Arguments

xdata	Input(s) vector (*n* by *m*)
ydata	Output(s) vector (*n* by *s*)
date	Production date (*n* by *1*)
rts	Returns to scale assumption "crs" Constant RTS (default) "vrs" Variable RTS "irs" Increasing RTS "drs" Decreasing RTS
orientation	Orientation of the measurement "i" Input-orientation "o" Output-orientation
sg	Employs second-stage optimization "ssm" Slack-sum maximization (default) "max" Date-sum maximization (only if date is defined) "min" Date-sum minimization (only if date is defined)
ncv	Non-controllable variable index(binary) for internal NDF (*1* by *(m+s)*)
env	Environment index for external NDF (*n* by *1*)
cv	Convexity assumption "convex" Convexity holds (default) "fdh" Free disposal hull (this will override rts)
mk	Marker on the map "dmu" DMU index (default) "eff" Efficiency score

Author(s)

Dong-Joon Lim, PhD

See Also

map.soa.ddf SOA mapping using DDF
map.soa.dea SOA mapping using DEA
map.soa.hdf SOA mapping using HDF
map.soa.sbm SOA mapping using SBM
map.soa.sf SOA mapping using SF

Examples

```
# Load engine dataset
  df <- dataset.engine.2015

# Subset for forced induction systems
  fis <- subset(df, grepl("^.C..", df[, 8]))

# Parameters
  x <- subset(fis, select = 4)
  y <- subset(fis, select = 6 : 7)
  d <- subset(fis, select = 2)

# Generate an SOA map
  map.soa.dea(x, y, d, "crs", "o")
```

---

map.soa.hdf *SOA mapping using HDF*

Description

Employs dm.hdf over time to generate a state-of-the-art map.

Usage

```
map.soa.hdf(xdata, ydata, date,
            rts="crs", wd=NULL, sg="ssm", cv="convex", mk="dmu")
```

Arguments

xdata	Input(s) vector (*n* by *m*)
ydata	Output(s) vector (*n* by *s*)
date	Production date (*n* by *1*)
rts	Returns to scale assumption "crs" Constant RTS (default) "vrs" Variable RTS "irs" Increasing RTS "drs" Decreasing RTS
wd	Weak disposability vector indicating (an) undesirable output(s) (*1* by *s*)

sg — Employs second-stage optimization
- "ssm" Slack-sum maximization (default)
- "max" Date-sum maximization (only if date is defined)
- "min" Date-sum minimization (only if date is defined)

cv — Convexity assumption
- "convex" Convexity holds (default)
- "fdh" Free disposal hull (this will override rts)

mk — Marker on the map
- "dmu" DMU index (default)
- "eff" Efficiency score

Author(s)

Dong-Joon Lim, PhD

References

D.-J. Lim, Internal combustion engine race: naturally aspirated vs turbo/super-charged, *working paper* (2015).

See Also

map.soa.ddf SOA mapping using DDF
map.soa.dea SOA mapping using DEA
map.soa.hdf SOA mapping using HDF
map.soa.sbm SOA mapping using SBM
map.soa.sf SOA mapping using SF

Examples

```
# Load engine dataset
  df <- dataset.engine.2015

# Subset for SC/TC 8 cylinder engines
  stc.8 <- subset(df, grepl("^.C..", df[, 8]) & df[, 3] == 8)

# Parameters
  x <- subset(stc.8, select = 4)
  y <- subset(stc.8, select = 5:7)
  d <- subset(stc.8, select = 2)

# Generate an SOA map
  map.soa.hdf(x, y, d, "vrs")
```

map.soa.sbm *SOA mapping using SBM*

Description

Employs dm.sbm over time to generate a state-of-the-art map.

Usage

```
map.soa.sbm(xdata, ydata, date,
            rts="crs", orientation="n", sg="ssm", cv="convex", mk="dmu")
```

Arguments

xdata	Input(s) vector (*n* by *m*)
ydata	Output(s) vector (*n* by *s*)
date	Production date (*n* by *1*)
rts	Returns to scale assumption "crs" Constant RTS (default) "vrs" Variable RTS "irs" Increasing RTS "drs" Decreasing RTS
orientation	Orientation of the measurement "n" Non-orientation (default) "i" Input-orientation "o" Output-orientation
sg	Employs second-stage optimization "ssm" Slack-sum maximization (default) "max" Date-sum maximization (only if date is defined) "min" Date-sum minimization (only if date is defined)
cv	Convexity assumption "convex" Convexity holds (default) "fdh" Free disposal hull (this will override rts)
mk	Marker on the map "dmu" DMU index (default) "eff" Efficiency score

Author(s)

Dong-Joon Lim, PhD

See Also

map.soa.ddf SOA mapping using DDF
map.soa.dea SOA mapping using DEA
map.soa.hdf SOA mapping using HDF
map.soa.sbm SOA mapping using SBM
map.soa.sf SOA mapping using SF

Examples

```
# Load engine dataset
  df <- dataset.engine.2015

# Subset for forced induction systems
  fis <- subset(df, grepl("^.C..", df[, 8]))

# Parameters
  x <- subset(fis, select = 4)
  y <- subset(fis, select = 6 : 7)
  d <- subset(fis, select = 2)

# Generate an SOA map
  map.soa.sbm(x, y, d)
```

---

map.soa.sf — *SOA mapping using SF*

Description

Employs `dm.sf` over time to generate a state-of-the-art map.

Usage

```
map.soa.sf(xdata, ydata, date,
           rts="crs", g=NULL, wd=NULL, sg="ssm", cv="convex", mk="dmu")
```

Arguments

xdata	Input(s) vector (*n* by *m*)
ydata	Output(s) vector (*n* by *s*)
date	Production date (*n* by *1*)
rts	Returns to scale assumption "crs" Constant RTS (default) "vrs" Variable RTS "irs" Increasing RTS "drs" Decreasing RTS
g	Directional vector indicating a measurement direction (*n* by *(m+s)*) By default (`NULL`), `xdata` & `ydata` will be used

wd	Weak disposability vector indicating (an) undesirable output(s) (*1* by *s*)
sg	Employs second-stage optimization "ssm" Slack-sum maximization (default) "max" Date-sum maximization (only if date is defined) "min" Date-sum minimization (only if date is defined)
cv	Convexity assumption "convex" Convexity holds (default) "fdh" Free disposal hull (this will override rts)
mk	Marker on the map "dmu" DMU index (default) "eff" Efficiency score

Author(s)

Dong-Joon Lim, PhD

References

D.-J. Lim, Internal combustion engine race: naturally aspirated vs turbo/super-charged, *working paper* (2015).

See Also

map.soa.ddf SOA mapping using DDF
map.soa.dea SOA mapping using DEA
map.soa.hdf SOA mapping using HDF
map.soa.sbm SOA mapping using SBM
map.soa.sf SOA mapping using SF

Examples

```
# Reproduce Table 2 in Lim, D-J. (2015)
  # Load engine dataset
    df <- dataset.engine.2015

  # Subset for 4 cylinder engines
    fce <- subset(df, df[, 3] == 4)

  # Parameters
    x <- subset(fce, select = 4)
    y <- subset(fce, select = 5 : 7)
    d <- subset(fce, select = 2)
    g <- data.frame(0, y)
    w <- matrix(c(1, 0, 0), ncol = 3)

  # Generate an SOA map
    map.soa.sf(x, y, d, "crs", g, w, mk = "eff")
```

`roc.dea` *Rate of change (RoC) calculation using DEA*

Description

Employs `dm.dea` over time to calculate RoCs.

Usage

```
roc.dea(xdata, ydata, date, t, rts="crs", orientation,
        sg="ssm", ftype="d", ncv=NULL, env=NULL, cv="convex")
```

Arguments

`xdata`	Input(s) vector (*n* by *m*)
`ydata`	Output(s) vector (*n* by *s*)
`date`	Production date (*n* by *1*)
`t`	A vantage point from which the RoC is captured
`rts`	Returns to scale assumption `"crs"` Constant RTS (default) `"vrs"` Variable RTS `"irs"` Increasing RTS `"drs"` Decreasing RTS
`orientation`	Orientation of the measurement `"i"` Input-orientation `"o"` Output-orientation
`sg`	Employs second-stage optimization `"ssm"` Slack-sum maximization (default) `"max"` Date-sum maximization `"min"` Date-sum minimization
`ftype`	Frontier type `"d"` Dynamic frontier (default) `"s"` Static frontier
`ncv`	Non-controllable variable index(binary) for internal NDF (*1* by *(m+s)*)
`env`	Environment index for external NDF (*n* by *1*)
`cv`	Convexity assumption `"convex"` Convexity holds (default) `"fdh"` Free disposal hull (this will override `rts`)

Value

`$eff_r`	Efficiency at release (i.e., at each production date)
`$eff_t`	Efficiency at `t`

`$lambda_t`	Intensity vector at `t`
`$eft_date`	Effective date
`$roc_past`	RoC observed from the obsolete DMUs in the past
`$roc_avg`	Average RoC
`$roc_local`	Local RoC

Author(s)

Dong-Joon Lim, PhD

References

Lim, Dong-Joon, Timothy R. Anderson, and Oliver Lane Inman. "Choosing effective dates from multiple optima in Technology Forecasting using Data Envelopment Analysis (TFDEA)." *Technological Forecasting and Social Change* 88 (2014): 91~97.

Lim, Dong-Joon, et al. "Comparing technological advancement of hybrid electric vehicles (HEV) in different market segments." *Technological Forecasting and Social Change* 97 (2015): 140~153.

Lim, Dong-Joon, et al. Technometrics Study Using DEA on Hybrid Electric Vehicles (HEVs). Handbook of Operations Analytics Using Data Envelopment Analysis. *Springer (forthcoming)*, 2016.

See Also

`dm.dea` Distance measure using DEA
`roc.dea` RoC calculation using DEA
`map.soa.dea` SOA mapping using DEA
`target.arrival.dea` Arrival target setting using DEA
`target.spec.dea` Spec target setting using DEA

Examples

```
# Reproduce Table 3 in Lim, D-J. et al.(2014)
  # Load airplane dataset
  df <- dataset.airplane.2017

  # ready
  x <- data.frame(Flew = rep(1, 28))
  y <- subset(df, select = 3 : 7)
  d <- subset(df, select = 2)

  # go
  roc.dea(x, y, d, 2007, "vrs", "o", "min", "d")$roc_past

# Reproduce Table 3 in Lim, D-J. et al.(2015)
  # Load hev dataset
  df <- dataset.hev.2013

  # ready
  x <- subset(df, select = 3)
```

```
y <- subset(df, select = 4 : 6)
d <- subset(df, select = 2)
c <- subset(df, select = 7)

# go
results <- roc.dea(x, y, d, 2013, "vrs", "o", "min", "d", env = c)
hev <- which(results$roc_local > 0)
data.frame(Class = c[hev, ],
           SOA = hev,
           LocalRoC = results$roc_local[hev, ])[order(c[hev, ]), ]
# NOTE: the published results include a typo on roc_local[82,]
#       this will be corrected in forthcoming book chapter(Lim, D-J. et al., 2016).
```

---

roc.hdf *Rate of change (RoC) calculation using HDF*

Description

Employs dm.hdf over time to calculate RoCs.

Usage

```
roc.hdf(xdata, ydata, date, t,
         rts="crs", wd=NULL, sg="ssm", ftype="d", cv="convex")
```

Arguments

xdata	Input(s) vector (*n* by *m*)
ydata	Output(s) vector (*n* by *s*)
date	Production date (*n* by *1*)
t	A vantage point from which the RoC is captured
rts	Returns to scale assumption "crs" Constant RTS (default) "vrs" Variable RTS "irs" Increasing RTS "drs" Decreasing RTS
wd	Weak disposability vector indicating (an) undesirable output(s) (*1* by *s*)
sg	Employs second-stage optimization "ssm" Slack-sum maximization (default) "max" Date-sum maximization "min" Date-sum minimization
ftype	Frontier type "d" Dynamic frontier (default) "s" Static frontier
cv	Convexity assumption "convex" Convexity holds (default) "fdh" Free disposal hull (this will override rts)

Value

`$eff_r`	Efficiency at release (i.e., at each production date)
`$eff_t`	Efficiency at `t`
`$lambda_t`	Intensity vector at `t`
`$eft_date`	Effective date
`$roc_past`	RoC observed from the obsolete DMUs in the past
`$roc_avg`	Average RoC
`$roc_local`	Local RoC

Author(s)

Dong-Joon Lim, PhD

References

D.-J. Lim, Internal combustion engine race: naturally aspirated vs turbo/super-charged, *working paper* (2015).

See Also

`dm.hdf` Distance measure using HDF
`roc.hdf` RoC calculation using HDF
`map.soa.hdf` SOA mapping using HDF
`target.arrival.hdf` Arrival target setting using HDF

Examples

```
# Load engine dataset
  df <- dataset.engine.2015

# Subset for 8 cylinder TC-P engines
  et <- subset(df, df[, 3] == 8 & df[, 8] == "TC-P")

# Parameters
  x <- subset(et, select = 4)
  y <- subset(et, select = 5 : 7)
  d <- subset(et, select = 2)
  w <- matrix(c(1, 0, 0), ncol = 3)

# Calc local Roc
  roc.hdf(x, y, d, 2015, "vrs", w, "min")
```

`roc.malmquist` *Malmquist Index: time-series productivity analysis*

Description

Employs distance measure over time to calculate the productivity changes.

Usage

```
roc.malmquist(xdata, ydata, tm=NULL, dm="dea", rts="crs", orientation,
              g=NULL, wd=NULL, ncv=NULL, env=NULL, cv="convex")
```

Arguments

`xdata`	Input(s) array (*n* by *m* by *t*)
`ydata`	Output(s) array (*n* by *s* by *t*)
`tm`	Tick mark of production dates (a vector length of *t*)
`dm`	Distance measure to calculate the productivity `"dea"` Data Envelopment Analysis (default) `"sbm"` Slack Based Model `"ddf"` Directional Distance Function `"hdf"` Hyperbolic Distance Function `"sf"` Shortage Function
`rts`	Returns to scale assumption `"crs"` Constant RTS (default) `"vrs"` Variable RTS `"irs"` Increasing RTS `"drs"` Decreasing RTS
`orientation`	Orientation of the measurement `"n"` Non-orientation (default) `"i"` Input-orientation `"o"` Output-orientation
`g`	Directional vector indicating a measurement direction (*n* by *(m+s)*) By default (`NULL`), `xdata` & `ydata` will be used
`wd`	Weak disposability vector indicating (an) undesirable output(s) (*1* by *s*)
`ncv`	Non-controllable variable index(binary) for internal NDF (*1* by *(m+s)*)
`env`	Environment index for external NDF (*n* by *1*)
`cv`	Convexity assumption `"convex"` Convexity holds (default) `"fdh"` Free disposal hull (this will override `rts`)

Value

`$cu`	Catching Up (aka technical efficiency change: TEC) index
`$fs`	Frontier Shift (FS) Index
`$mi`	Malmquist Index

Author(s)

Dong-Joon Lim, PhD

References

R. Fare, S. Grosskopf, and C. A. K. Lovell, Production Frontiers. *Cambridge University Press*, 1994.

See Also

`dm.ddf` Distance measure using DDF
`dm.dea` Distance measure using DEA
`dm.hdf` Distance measure using HDF
`dm.sbm` Distance measure using SBM
`dm.sf` Distance measure using SF

Examples

```
# Load data
  df <- array(c(4,      3,    9,  10,   7, 4,  3, 5,
                5,     12,    3,   8,   1, 4, 14, 3,
                1,      1,    1,   1,   1, 1,  1, 1,
                3.4,    2,   10,   8, 10, 4,  1, 5,
                6,     10,  3.5,   7,   2, 4, 12, 3,
                1,      1,    1,   1,   1, 1,  1, 1,
                2.8,  1.8,    8,   7, 10, 3,  1, 5,
                5.7,  8.8,  2.8,   5,   2, 5,  9, 3,
                1,      1,    1,   1,   1, 1,  1, 1,
                2.2,  1.5,    8,   5,   8, 3,  1, 5,
                6,      8,  2.3, 3.5,   2, 5,  7, 3,
                1,      1,    1,   1,   1, 1,  1, 1),
              c(8, 3, 4))

# Run
  roc.malmquist(df[,1:2,], df[,3,], dm = "sbm", orientation = "n")
```

roc.sf *Rate of change (RoC) calculation using SF*

Description

Employs `dm.sf` over time to calculate RoCs. This function is valid only when multiplicative form of directional vector is used.

Usage

```
roc.sf(xdata, ydata, date, t,
       rts="crs", g=NULL, wd=NULL, sg="ssm", ftype="d", cv="convex")
```

Arguments

`xdata`	Input(s) vector (*n* by *m*)
`ydata`	Output(s) vector (*n* by *s*)
`date`	Production date (*n* by *1*)
`t`	A vantage point from which the RoC is captured
`rts`	Returns to scale assumption `"crs"` Constant RTS (default) `"vrs"` Variable RTS `"irs"` Increasing RTS `"drs"` Decreasing RTS
`g`	Directional vector indicating a measurement direction (*n* by *(m+s)*) By default (`NULL`), `xdata` & `ydata` will be used
`wd`	Weak disposability vector indicating (an) undesirable output(s) (*1* by *s*)
`sg`	Employs second-stage optimization `"ssm"` Slack-sum maximization (default) `"max"` Date-sum maximization `"min"` Date-sum minimization
`ftype`	Frontier type `"d"` Dynamic frontier (default) `"s"` Static frontier
`cv`	Convexity assumption `"convex"` Convexity holds (default) `"fdh"` Free disposal hull (this will override `rts`)

Value

`$eff_r`	Efficiency at release (i.e., at each production date)
`$eff_t`	Efficiency at `t`
`$lambda_t`	Intensity vector at `t`

`$eft_date`	Effective date
`$roc_past`	RoC observed from the obsolete DMUs in the past
`$roc_avg`	Average RoC
`$roc_local`	Local RoC

Author(s)

Dong-Joon Lim, PhD

References

D.-J. Lim, Internal combustion engine race: naturally aspirated vs turbo/super-charged, *working paper* (2015).

See Also

`dm.sf` Distance measure using SF
`roc.sf` RoC calculation using SF
`map.soa.sf` SOA mapping using SF
`target.arrival.sf` Arrival target setting using SF

Examples

```
# Reproduce Mercedes-Benz CLA45 AMG's local RoC in Table 5 in Lim, D-J. (2015)
  # Load engine dataset
    df <- dataset.engine.2015

  # Subset for 4 cylinder engines
    fce <- subset(df, df[, 3] == 4)

  # Parameters
    x <- subset(fce, select = 4)
    y <- subset(fce, select = 5 : 7)
    d <- subset(fce, select = 2)
    g <- as.matrix(data.frame(0, y))
    w <- matrix(c(1, 0, 0), ncol = 3)

  # Calc local Roc
    roc.sf(x, y, d, 2014, "crs", g, w, "min")$roc_local[348, ]
```

`target.arrival.dea` *Arrival target setting using DEA*

Description

Employs `dm.dea` over time to estimate the arrival of known specifications.

Usage

```
target.arrival.dea(xdata, ydata, date, t, rts="crs", orientation,
                   sg="ssm", ftype="d", ncv=NULL, env=NULL, cv="convex")
```

Arguments

- `xdata` Input(s) vector (*n* by *m*)
- `ydata` Output(s) vector (*n* by *s*)
- `date` Production date (*n* by *1*)
- `t` A vantage point from which the RoC is captured
- `rts` Returns to scale assumption
  - `"crs"` Constant RTS (default)
  - `"vrs"` Variable RTS
  - `"irs"` Increasing RTS
  - `"drs"` Decreasing RTS
- `orientation` Orientation of the measurement
  - `"i"` Input-orientation
  - `"o"` Output-orientation
- `sg` Employs second-stage optimization
  - `"ssm"` Slack-sum maximization (default)
  - `"max"` Date-sum maximization
  - `"min"` Date-sum minimization
- `ftype` Frontier type
  - `"d"` Dynamic frontier (default)
  - `"s"` Static frontier
- `ncv` Non-controllable variable index(binary) for internal NDF (*1* by *(m+s)*)
- `env` Environment index for external NDF (*n* by *1*)
- `cv` Convexity assumption
  - `"convex"` Convexity holds (default)
  - `"fdh"` Free disposal hull (this will override `rts`)

Value

- `$eff_t` Efficiency at `t`
- `$lambda_t` Intensity vector at `t`
- `$eft_date` Effective date
- `$roc_avg` Average RoC
- `$roc_local` Local RoC
- `$roc_ind` Individualized RoC
- `$arrival_avg` Estimated arrival using `roc_avg`
- `$arrival_seg` Estimated arrival using `roc_ind`

Author(s)

Dong-Joon Lim, PhD

References

Lim, Dong-Joon, Timothy R. Anderson, and Oliver Lane Inman. "Choosing effective dates from multiple optima in Technology Forecasting using Data Envelopment Analysis (TFDEA)." *Technological Forecasting and Social Change* 88 (2014): 91~97.

Lim, Dong-Joon, and Timothy R. Anderson. Time series benchmarking analysis for new product scheduling: who are the competitors and how fast are they moving forward?. Advances in DEA Theory and Applications: with Examples in Forecasting Models. *Wiley (forthcoming)*, 2016.

See Also

`dm.dea` Distance measure using DEA
`roc.dea` RoC calculation using DEA
`map.soa.dea` SOA mapping using DEA
`target.arrival.dea` Arrival target setting using DEA
`target.spec.dea` Spec target setting using DEA

Examples

```
# Reproduce Table 4 in Lim, D-J., and Timothy R. Anderson.(2016)
  # Load airplane dataset
  df <- dataset.airplane.2017

  # ready
  x <- data.frame(Flew = rep(1, 28))
  y <- subset(df, select = 3 : 7)
  d <- subset(df, select = 2)

  # go
  target.arrival.dea(x, y, d, 2007, "vrs", "o", "min", "d")$arrival_seg
```

`target.arrival.hdf` *Arrival target setting using HDF*

Description

Employs `dm.hdf` over time to estimate the arrival of known specifications.

Usage

```
target.arrival.hdf(xdata, ydata, date, t, rts="crs",
                   wd=NULL, sg="ssm", ftype="d", cv="convex")
```

Arguments

`xdata`	Input(s) vector (*n* by *m*)
`ydata`	Output(s) vector (*n* by *s*)
`date`	Production date (*n* by *1*)
`t`	A vantage point from which the RoC is captured
`rts`	Returns to scale assumption `"crs"` Constant RTS (default) `"vrs"` Variable RTS `"irs"` Increasing RTS `"drs"` Decreasing RTS
`wd`	Weak disposability vector indicating (an) undesirable output(s) (*1* by *s*)
`sg`	Employs second-stage optimization `"ssm"` Slack-sum maximization (default) `"max"` Date-sum maximization `"min"` Date-sum minimization
`ftype`	Frontier type `"d"` Dynamic frontier (default) `"s"` Static frontier
`cv`	Convexity assumption `"convex"` Convexity holds (default) `"fdh"` Free disposal hull (this will override `rts`)

Value

`$eff_t`	Efficiency at `t`
`$lambda_t`	Intensity vector at `t`
`$eft_date`	Effective date
`$roc_avg`	Average RoC
`$roc_local`	Local RoC
`$roc_ind`	Individualized RoC
`$arrival_avg`	Estimated arrival using `roc_avg`
`$arrival_seg`	Estimated arrival using `roc_ind`

Author(s)

Dong-Joon Lim, PhD

References

Lim, Dong-Joon, et al. "Comparing technological advancement of hybrid electric vehicles (HEV) in different market segments." *Technological Forecasting and Social Change* 97 (2015): 140~153.

Lim, Dong-Joon, and Timothy R. Anderson. Time series benchmarking analysis for new product scheduling: who are the competitors and how fast are they moving forward?. Advances in DEA Theory and Applications: with Examples in Forecasting Models. *Wiley (forthcoming)*, 2016.

See Also

dm.hdf Distance measure using HDF
roc.hdf RoC calculation using HDF
map.soa.hdf SOA mapping using HDF
target.arrival.hdf Arrival target setting using HDF

Examples

```
# Estimate arrivals of MY2015 SC/TC 8 cylinder engines
  # Load engine dataset
    df <- dataset.engine.2015

  # Subset for SC/TC 8 cylinder engines
    stc.8 <- subset(df, grepl("^.C..", df[, 8]) & df[, 3] == 8)

  # Parameters
    x <- subset(stc.8, select = 4)
    y <- subset(stc.8, select = 5:7)
    d <- subset(stc.8, select = 2)

  # Generate an SOA map
    target.arrival.hdf(x, y, d, 2014, "vrs")
```

---

target.arrival.sf — *Arrival target setting using SF*

Description

Employs dm.sf over time to estimate the arrival of known specifications. This function is valid only when multiplicative form of directional vector is used.

Usage

```
target.arrival.sf(xdata, ydata, date, t, rts="crs", g=NULL,
                  wd=NULL, sg="ssm", ftype="d", cv="convex")
```

Arguments

xdata	Input(s) vector (*n* by *m*)
ydata	Output(s) vector (*n* by *s*)
date	Production date (*n* by *1*)
t	A vantage point from which the RoC is captured
rts	Returns to scale assumption "crs" Constant RTS (default) "vrs" Variable RTS "irs" Increasing RTS "drs" Decreasing RTS

g	Directional vector indicating a measurement direction (*n* by *(m+s)*) By default (NULL), xdata & ydata will be used
wd	Weak disposability vector indicating (an) undesirable output(s) (*1* by *s*)
sg	Employs second-stage optimization "ssm" Slack-sum maximization (default) "max" Date-sum maximization "min" Date-sum minimization
ftype	Frontier type "d" Dynamic frontier (default) "s" Static frontier
cv	Convexity assumption "convex" Convexity holds (default) "fdh" Free disposal hull (this will override rts)

Value

$eff_t	Efficiency at t
$lambda_t	Intensity vector at t
$eft_date	Effective date
$roc_avg	Average RoC
$roc_local	Local RoC
$roc_ind	Individualized RoC
$arrival_avg	Estimated arrival using roc_avg
$arrival_seg	Estimated arrival using roc_ind

Author(s)

Dong-Joon Lim, PhD

References

Lim, Dong-Joon, et al. "Comparing technological advancement of hybrid electric vehicles (HEV) in different market segments." *Technological Forecasting and Social Change* 97 (2015): 140~153.

Lim, Dong-Joon, and Timothy R. Anderson. Time series benchmarking analysis for new product scheduling: who are the competitors and how fast are they moving forward?. Advances in DEA Theory and Applications: with Examples in Forecasting Models. *Wiley (forthcoming)*, 2016.

See Also

dm.sf Distance measure using SF
roc.sf RoC calculation using SF
map.soa.sf SOA mapping using SF
target.arrival.sf Arrival target setting using SF

Examples

```
# Estimate arrivals of MY2013 hev models
  # Load hev dataset
  df <- dataset.hev.2013

  # ready
  x <- subset(df, select = 3)
  y <- subset(df, select = 4 : 6)
  d <- subset(df, select = 2)
  g <- data.frame(x, y)

  # go
  target.arrival.sf(x, y, d, 2012, "vrs", g)$arrival_seg
```

`target.spec.dea` *Spec target setting using DEA*

Description

Employs `inverse DEA` to estimate specifications(in/out-puts) to achieve a predetermined efficiency.

Usage

```
target.spec.dea(xdata, ydata, date=NULL, t=NULL, dt=NULL, dmu, et="c",
                alpha=NULL, beta=NULL, wv=NULL, rts="crs", sg="ssm", ftype="d",
                ncv=NULL, env=NULL, cv="convex", bound=TRUE)
```

Arguments

`xdata`	Input(s) vector (*n* by *m*)
`ydata`	Output(s) vector (*n* by *s*)
`date`	Production date (*n* by *1*)
`t`	A vantage point from which the RoC is captured
`dt`	Delta *t* i.e., specs are estimated within PPS at *t+dt*
`dmu`	DMU whose inputs(or outputs) are to be estimated
`et`	Efficiency target; default value ("c") retains the current efficiency
`alpha`	Perturbed input(s) of designated DMU (*1* by *m*)
`beta`	Perturbed output(s) of designated DMU (*1* by *s*)
`wv`	Weight vector for scalarization (*1* by *m* or *s*)
`rts`	Returns to scale assumption `"crs"` Constant RTS (default) `"vrs"` Variable RTS `"irs"` Increasing RTS `"drs"` Decreasing RTS

`sg`	Employs second-stage optimization `"ssm"` Slack-sum maximization (default) `"max"` Date-sum maximization `"min"` Date-sum minimization
`ftype`	Frontier type `"d"` Dynamic frontier (default) `"s"` Static frontier
`ncv`	Non-controllable variable index(binary) for internal NDF (*1* by *(m+s)*)
`env`	Environment index for external NDF (*n* by *1*)
`cv`	Convexity assumption `"convex"` Convexity holds (default) `"fdh"` Free disposal hull (this will override `rts`)
`bound`	Puts upper/lower bounds on alpha/beta if `TRUE`(default)

Value

`$alpha`	Estimated input(s)
`$beta`	Estimated output(s)
`$lambda`	Intensity vector
`$xslack`	Input slack
`$yslack`	Output slack

Author(s)

Dong-Joon Lim, PhD

References

Lim, Dong-Joon, "Inverse DEA with frontier changes for new product target setting." *European Journal of Operational Research* 254.2 (2016): 510~516.

Wei, Quanling, Jianzhong Zhang, and Xiangsun Zhang. "An inverse DEA model for inputs/outputs estimate." *European Journal of Operational Research* 121.1 (2000): 151~163.

See Also

`dm.dea` Distance measure using DEA
`roc.dea` RoC calculation using DEA
`target.arrival.dea` Arrival target setting using DEA

Examples

```
# Reproduce Example 2 in Wei, Q. et al.(2000)
  # ready
  x <- matrix(c(1, 1, 1), 3)
  y <- matrix(c(4, 8, 5, 8, 4, 5), 3)
  a <- matrix(1.8, 1)
  w <- matrix(c(0.5, 0.5), 1)
```

```
  # go
  target.spec.dea(x, y, dmu = 3, alpha = a, wv = w, rts = "crs")$beta

# Reproduce Table 4 in Lim, D-J. (2016)
  # Load engine dataset
    df <- dataset.engine.2015

  # Subset for forced induction systems
    fis <- subset(df, grepl("^.C..", df[, 8]))

  # ready
    # Suppose one wants to estimate Porsche 911 turbo s' engine specs
    # to retain its current competitiveness with downsized 3.5 litre engine in 2018.
    # What might be the minimum specs to achieve this goal
    # considering the technological changes we've seen so far?
    # Plus, the CEO wants to put more emphasis on the torque improvement over HP.
      d <- subset(fis, select = 2)
      x <- subset(fis, select = 4)
      y <- subset(fis, select = 6 : 7)
      a <- as.matrix(3.5)
      w <- matrix(c(0.3, 0.7), 1)

  # go
   target.spec.dea(x, y, d, 2015, 3, 262, alpha = a, wv = w, rts = "vrs", sg = "min")$beta
```

Index