Package 'TFDEA'

March 30, 2015

Version 0.9.8.3

Date 2015-03-28

Title Technology Forecasting using DEA (Data Envelopment Analysis)

Depends R (>= 3.0.0), lpSolveAPI

License GPL-2

Description The TFDEA algorithm for technology forecasts when future products
will be introduced based upon their features.
It also includes DEA (Data Envelopment Analysis) functions including extensions dealing with
with infeasibility.
In addition it includes some standard technology forecasting data sets.

URL http://www.pdx.edu/extreme-technology-analytics/open-tfdea-r-package

Author Tom Shott [aut, cre],
Dong-Joon Lim [aut]

Maintainer Tom Shott <tshott@pdx.edu>

NeedsCompilation no

Repository CRAN

Date/Publication 2015-03-30 15:07:52

R topics documented:

DEA *DEA: Data Envelopment Analysis*

Description

Estimates a Data Envelopment Analysis frontier and calculates efficiency measures

Usage

```
DEA(x, y, rts="vrs", orientation="input", slack=TRUE, dual=FALSE,
  second="none", z=0, round=FALSE, debug=1)
```

Arguments

x	Inputs or resources used by each decision making unit.
y	Outputs or products of each decision making unit. Must have same number of rows as x
rts	Returns to scale for the application, production technology, or industry studied

vrs	Variable returns to scale, convexity and free disposability
drs	Decreasing returns to scale, convexity, down-scaling and free disposability
crs	Constant returns to scale, convexity and free disposability
irs	Increasing returns to scale, (up-scaling, but not down-scaling), convexity and free disposability

orientation	Orientation of the DEA model - primary emphasis on input-reduction input or output-augmentation output
slack	Optional: slack=TRUE indicates a secondary objective function of maximizing radial slacks to identify weakly efficient DMUs
dual	Optional: dual=TRUE reports back the dual weights (multipliers) for the inputs and outputs
round	Optional: round=TRUE rounds efficiency values to 0 and 1 if close.
second	Optional: Enables an alternate secondary objective function based on lambda and the z argument. The default is none. Other options are min or max which will then minimize or maximize z*lambda while holding efficiency constant for each decision making unit. Note that this precludes slack maximization in the current implementation
z	Optional:a matrix with one column and the same number of rows (decision making units) as x and y, it is only used when second=min or max
debug	Optional: Only for debugging. If debug is a integer greater then zero debug information will be output.

Details

This DEA function draws inspiration from previous R packages for doing DEA including Benchmarking and FEAR. As such it was designed to use similar parameters and return similar results

to allow users to switch between packages. The DEA function was developed to support a function for doing Technology Forecasting using Data Envelopment Analysis or TFDEA. In particular, TFDEA requires an option to resolve multiple optima that is similar to but different from the standard slack maximization approach in Data Envelopment Analysis. This feature is exposed through DEA function's second and z parameters.

Value

`$status`	If the solver returned a non-zero status for each decision making unit
`$eff`	Efficiency score for each decision making unit
`$lambda`	Lambda values for each decision making unit
`$vx`	Input weights used by each decision making unit, only returned when dual=TRUE
`$uy`	Output weights used by each decision making unit, only returned when dual=TRUE
`$w`	W value for each decision making unit, only returned when dual=TRUE
`$sx`	Radial input slacks, only returned when slack=TRUE
`$sy`	Radial output slacks, only returned when slack=TRUE

References

Bogetoft and Otto; Benchmarking with DEA, SFA, and R; Springer 2011

Paul W. Wilson (2008), FEAR 1.0: A Software Package for Frontier Efficiency Analysis with R, Socio-Economic Planning Sciences 42, 247-254

See Also

SDEA Super-efficiency - an extension to regular DEA that allows for differentiating between efficient DMUs.

TFDEA Technology Forecasting using Data Envelopment Analysis - a method of technology forecasting using past data to predict future capabilities

Examples

```
x <- matrix(c(8,2,4,7,2,8,4,3),ncol=2,dimnames=list(LETTERS[1:4]))
colnames(x) = c("X1", "X2")
y <- matrix(c(1,1,1,1),ncol=1,dimnames=list(LETTERS[1:4],"Y"))

# Simple radial DEA efficiency
DEA(x, y, rts="crs", orientation="input")

# Simple radial DEA efficiency with slack maximization
DEA(x, y, rts="crs", orientation="input", slack=TRUE)

# Example of secondary objective function
x <- matrix(c(8,2,4,7,10,12,2,8,4,3,2,2),ncol=2,dimnames=list(LETTERS[1:6]))
colnames(x) = c("X1", "X2")
y <- matrix(c(1,1,1,1,1,1),ncol=1,dimnames=list(LETTERS[1:6],"Y"))
z <- matrix(c(1:6),ncol=1,dimnames=list(LETTERS[1:6],"Z"))
```

```
DEA (x,y,rts="crs", orientation="input", round=TRUE, slack=FALSE,
  second="min", z=z)

DEA (x,y,rts="crs", orientation="input", round=TRUE, slack=FALSE,
  second="max", z=z)
```

`fighter_jet` *Data: Fighter Jet technology forecasting data*

Description

Technical data from fighter jets for technology forecasting.

Usage

```
data(fighter_jet)
```

Format

A data frame containing historical jet fighter data for technology forecasting.

`CruiseSpeed` a numeric vector

`FirstFlight` a numeric vector

`Guns` a numeric vector

`InstantaneousTurnRate` a numeric vector

`MaintenanceHoursPerFlightHour` a numeric vector

`MaximumMachNumber` a numeric vector

`MeanFlightHoursBetweenFailure` a numeric vector

`Name` a numeric vector

`NumberOfBVRMissiles` a numeric vector

`NumberOfDogfightMissiles` a numeric vector

`NumberOfSimultaneousTargets` a numeric vector

`Payload` a numeric vector

`RadarRange` a numeric vector

`Range` a numeric vector

`RangeOfBVRMissiles` a numeric vector

`RangeOfDogfightMissiles` a numeric vector

`SeaLevelClimbRate` a numeric vector

`SustainedTurnRate` a numeric vector

`TakeoffRoll` a numeric vector

References

O. Inman, T.R. Anderson, R. Harmon, Predicting U.S. jet fighter aircraft introductions from 1944 to 1982: A dogfight between regression and TFDEA, Technol. Forecast. Soc. Change. 73 (2006) 1178~1187.

D.-J. Lim, T.R. Anderson, O.L. Inman, Choosing effective dates from multiple optima in Technology Forecasting using Data Envelopment Analysis (TFDEA), Technol. Forecast. Soc. Change. 88 (2014) 91~97.

Examples

```
# Reproduce the published results
data(fighter_jet)
attach(fighter_jet)
x <- data.frame(rep(1, 19)); row.names(x) <- Name
y <- data.frame(MeanFlightHoursBetweenFailure, Payload, MaximumMachNumber,
  RangeOfBVRMissiles);
row.names(y) <- Name
z <- data.frame(FirstFlight); row.names(z) <- Name
detach(fighter_jet)
TFDEA(x, y, z, 1960, rts="vrs", orientation="output", mode="dynamic",
  second="min", debug=0)
```

---

isEfficient *Check if DEA DMU is efficient*

Description

Cheks that based upon model orientation a DMU is efficient. For input orientation DMU's >= 0 are efficient, for output orientation DMU's <= 0.

Usage

```
isEfficient(eff, orientation)
```

Arguments

`eff`	Efficiency - numeric value.
`orientation`	Orientation is either `input` or `output`

Value

Returns a logical value.

SDEA *Super-efficiency Data Envelopment Analysis*

Description

Calculate Data Envelopment Analysis super-efficiency

Usage

```
SDEA(x, y, rts="vrs", orientation="input", slack=TRUE, dual = FALSE,
  cook=FALSE,
  second="none", z=0,
  round=FALSE, debug=1)
```

Arguments

x	Inputs or resources used by each decision making unit
y	Outputs or products of each decision making unit
rts	Returns to scale for the application, production technology, or industry studied

vrs	Variable returns to scale, convexity and free disposability
drs	Decreasing returns to scale, convexity, down-scaling and free disposability
crs	Constant returns to scale, convexity and free disposability
irs	Increasing returns to scale, (up-scaling, but not down-scaling), convexity and free disposability

orientation	Orientation of the DEA model - primary emphasis on input-reduction `input` or output-augmentation `output`
slack	Optional: `slack=TRUE` indicates a secondary objective function of maximizing non-radial slacks
dual	Optional: `dual=TRUE` reports back the dual weights (multipliers) for the inputs and outputs
cook	Optional: `cook=TRUE` enables using Cook algo. to compute super efficiency of DMUs that are infeasible under the standard model.
round	Optional: `round=TRUE` rounds efficiency values to 0 and 1 if close.
second	Optional: Enables an alternate secondary objective function based on lambda and the z matrix. The default is `second=none`. Other options include `min` or `max` which will then minimize or maximize z*lambda while holding efficiency constant for each decision making unit. Note that this precludes slack maximization in the current implementation
z	Optional: a matrix with one column and the same number of rows (decision making units) as x and y, it is required when `second=min` or `max`
debug	Optional: Only for debugging. If debug is a integer greater then zero debug information is output.

Details

This function implements either the standard super-efficiency model or the Cook's extended model whereby each decision making unit is compared to all other decision making units but not itself. This allows for efficiency scores that are "better" than 1 for most efficient Decision Making Units. Note that the standard super-efficiency model can cause infeasibilities especially when the Varialbe Returns to Scale is assumed. (Ex. `rts=vrs`)

Value

`$status`	If the solver returned a non-zero status for each decision making unit
`$eff`	Efficiency score for each decision making unit
`$lambda`	Lambda values for each decision making unit
`$se.eff`	Tau(Gamma) in input(output)-orientated model, only returned when cook=TRUE
`$se.excess`	Beta(Delta) in input(output)-oriented model, only returned when cook=TRUE
`$vx`	Input weights used by each decision making unit, only returned when dual=TRUE
`$uy`	Output weights used by each decision making unit, only returned when dual=TRUE
`$w`	W value for each decision making unit, only returned when dual=TRUE
`$sx`	Radial input slacks, only returned when slack=TRUE
`$sy`	Radial output slacks, only returned when slack=TRUE

References

P. Andersen, N.C. Petersen, A Procedure for Ranking Efficient Units in Data Envelopment Analysis, Manage. Sci. 39 (1993) 1261~1264.

W.D. Cook, L. Liang, Y. Zha, J. Zhu, A modified super-efficiency DEA model for infeasibility, J. Oper. Res. Soc. 60 (2009) 276~281.

See Also

DEA Data Envelopment Analysis - provides a variety of standard DEA models for examining the efficiency of different decision making units.

TFDEA Technology Forecasting Using Data Envelopment Analysis - a method of technology forecasting using past data to predict future capabilities

Examples

```
x <- matrix(c(1,2,3,4),ncol=1,dimnames=list(LETTERS[1:4],"X"))
y <- matrix(c(1,3,4,3),ncol=1,dimnames=list(LETTERS[1:4],"Y"))

# Simple radial DEA super-efficiency
SDEA(x, y, rts="vrs", orientation="input")

# Simple radial DEA super-efficiency with slack maximization
SDEA(x, y, rts="vrs", orientation="input", slack=TRUE)

# Cook's super-efficiency with slack maximization
```

```
SDEA(x, y, rts="vrs", orientation="input", slack=TRUE, cook=TRUE)

# Example of secondary objective function
x <- matrix(c(8,2,4,7,10,12,2,8,4,3,2,2),ncol=2,dimnames=list(LETTERS[1:6]))
colnames(x) = c("X1", "X2")
y <- matrix(c(1,1,1,1,1,1),ncol=1,dimnames=list(LETTERS[1:6],"Y"))
z <- matrix(c(1:6),ncol=1,dimnames=list(LETTERS[1:6],"Z"))

SDEA (x, y, rts="crs", orientation="input", slack=FALSE, second="min", z=z)
```

TFDEA *Technology Forecasting Using DEA*

Description

Calculate the technology rate of change (ROC) which can then be used for predicting future product performance, estimate new product release dates, and other purposes.

Usage

```
TFDEA(x, y, dmu_date_rel, date_forecast, rts="vrs", orientation="output",
  second="min", mode="static", segroc=FALSE, debug=1)
```

Arguments

x	Inputs or resources used by each decision making unit
y	Outputs or products of each decision making unit
dmu_date_rel	Date of introduction (release) for decision making unit or product
date_forecast	Date of forecast
rts	Returns to scale for the application, production technology, or industry studied

vrs	Variable returns to scale, convexity and free disposability
drs	Decreasing returns to scale, convexity, down-scaling and free disposability
crs	Constant returns to scale, convexity and free disposability
irs	Increasing returns to scale, (up-scaling, but not down-scaling), convexity and free disposability

orientation	Orientation of the DEA model - primary emphasis on input-reduction `input` or output-augmentation `output`. Note that unlike the DEA functions, the default is `output` orientation.
second	Optional: Enables an alternate secondary objective function based on the product of lambda and dmu_date_rel. The default is `min`. Other options include `none` or `max` which will then skip this step or minimize date * lambda while holding the efficiency constant.
mode	Declares if the technology forecast is done using `static` or `dynamic` frontiers.

	A static frontier is where the forecast is made using a fixed date of date_forecast. A dynamic frontier allows for different based dates on the frontier based on the product of dmu_date and lambda. When a dynamic frontier is selected, a secondary objective function should be specified to avoid problems with multiple optima (`second=min` or `max`)
`segroc`	Uses segmented rate of change if `TURE`.
`debug`	Optional: Only for debugging. If `debug` is a integer greater then zero debug information will be output.

Value

TFDEA returns sets of values at three points in time, at the initial release of the product, `_rel`, at the forecast data specified by `date_forecast`, `_cur`, and for the date in the future when the product is forecast `for`.

The function returns a number of values per product (DMU). The standardized efficiency (all inefficiencies are between 0 and 1, for `input` and `output` orientation) `eff`, and the lambda values, `lambda`, are returned.

A rate of technology change `roc` is returned for products efficient at release. At current time a local rate of change and at forecast time an individualized rate of change is returned - `sroc`. If `segroc = FALSE` then the `sroc` is the average rate of change and is the same for every product.

Lastly a date for current and forecast is returned, `date`. If `mode = dynamic` then the current date is the current date adjusted by what products the product is being compared to. If `static` is used then the `date` is the `date_forecast` for all products. If the product release date is in the future then a forecast for the product is returned.

Not all values are calculated for all products at all points in time. For example a a product released in the past with that is inefficient at release would not have a `roc` or `sroc` calculated because the product is not used to calculate the overall rate of technology change.

`$date_soa`	List of unique release dates for conducting analyses
`$dmu_eff_rel`	Efficiency per product (DMU) at time of release
`$dmu_lambda_rel`	Lambdas per DMU at time of release
`$dmu_eff_cur`	Efficiency per product (DMU) at current time (date of forecast)
`$dmu_lambda_cur`	Lambdas per DMU at current time
`$dmu_date_cur`	Adjusted current date for DMU
`$dmu_roc_cur`	Rate of Change for product from date of release to current date
`$dmu_sroc_cur`	Local rate of change for the product
`$dmu_eff_for`	Superefficiency per product (DMU)
`$dmu_lambda_for`	Lambda per DMU at forecast time
`$dmu_date_for`	Date forecast for product based upon supereffeciency and technology rate of change
`$dmu_sroc_for`	Individualized rate of change for product
`$roc`	Average Rate of Technology Change

See Also

DEA

SDEA

Examples

```
# Example from Inman (2004) p. 93-104, predicting flash drive introduction dates

drive <- c("A", "B", "C", "D", "E", "F", "G")

x           <- data.frame(c(16, 14, 8, 25, 40, 30, 40))
rownames(x) <- drive
colnames(x) <- c("Cost")

y           <- data.frame(c(16, 32, 32, 128, 32, 64, 256))
rownames(y) <- drive
colnames(y) <- c("Capacity")


z           <- data.frame(c(2001, 2002, 2003, 2004, 2001, 2002, 2004))
rownames(z) <- drive
colnames(z) <- c("Date_Intro")

# Calc intro date for products using forecast year 2003
results <- TFDEA(x, y, z, 2003, rts="vrs", orientation="output", mode="dynamic")

# Examine what dates are forecast for DMU D & G
print(results$dmu_date_for)
```

---

wireless_2012 *Data: Wireless Forecasting Data*

Description

Technical data from wireless protocol for technology forecasting.

Usage

```
data(wireless_2012)
```

Format

A data frame containing historical wireless data for technology forecasting.

`Date` a numeric vector

`Bandwidth` a numeric vector

`Bitrate` a numeric vector

References

D.-J. Lim, T.R. Anderson, J. Kim, Forecast of wireless communication technology: A comparative study of regression and TFDEA Model, in: Technol. Manag. Emerg. Technol., PICMET, Vancouver, Canada, 2012: pp. 1247~1253.

Examples

```
# Reproduce the published results
data(wireless_2012)
attach(wireless_2012)
x <- data.frame(Bandwidth); row.names(x)  <- rownames(wireless_2012)
y <- data.frame(Bitrate); row.names(y) <- rownames(wireless_2012)
z <- data.frame(Date); row.names(z) <- rownames(wireless_2012)
detach(wireless_2012)
TFDEA(x, y, z, 2001, rts="crs", orientation="input", second="none",
  mode="static", debug=0)
```

Index