Package ‘FEAR’

July 23, 2013

Version 2.0

License file LICENSE.

Description Routines for analyzing and estimating the productive efficiency of firms.

Title Frontier Efficiency Analysis with R

Author Paul W. wilson <pww@clemson.edu>

Maintainer Paul W. wilson <pww@clemson.edu>

Depends R (>= 3.0.0)

Date 18 July 2013

R topics documented:

---

ap *Outlier Detection for Non-parametric Frontier Models*

Description

This routine implements the Wilson (1993) outlier detection method.

Usage

```
ap(X, Y, NDEL = 3, err.check = TRUE)
```

Arguments

`X`	$p \times n$ matrix of observations on p inputs of n firms whose efficiency is to be estimated;
`Y`	$q \times n$ matrix of observations on q outputs of n firms whose efficiency is to be estimated;
`NDEL`	total number of observaitons to be deleted; i in Wilson(1993);
`err.check`	equals TRUE (default) for error-checking, or FALSE for no error-checking.

Details

In terms of the notation in Wilson (1993), this routine computes $R_{min}^{(i)}$ for $i = 1, \ldots, NDEL$. The QR decomposition is computed using Householder transformations. See Wilson (1993) for furter details, but note that results in Table 2 and Figure 2 of Wilson (1993) are incorrect; see Wilson (2010) for details.

Value

`r0`	a vector of length NDEL containing values $R_{min}^{(i)}$; the first element of r0 contains the value of $R_{min}^{(1)}$, the second element contains the value of $R_{min}^{(2)}$, etc.
`imat`	an NDEL by NDEL matrix containing indices of observations; indices in the i-th row of imat determine $R_{min}^{(i)}$.
`ratio`	an n by NDEL matrix containing data that can be used to produce log-ratio plots as in Wilson (1993); see the command `ap.plot`.

Author(s)

Paul W. Wilson

References

Wilson, P.W. (1993), Detecing outliers in deterministic nonparametric frontier models with multiple outputs, *Journal of Business and Economic Statistics* 11, 319-323.

Wilson, P.W. (2010), Detecing outliers in deterministic nonparametric frontier models with multiple outputs: Correction, unpublished working paper, Department of Economics, Clemson University, Clemson, South Carolina 29634 (available on-line at http://www.clemson.edu/economics/faculty/wilson/Papers/ap-corrected.pdf.

See Also

`ap.plot`

Examples

```
x=matrix(rnorm(25),nrow=1)
y=2+2*x+1.5*rnorm(25)
ap(X=x,Y=y,NDEL=4)
```

`ap.plot` *Produce log-ratio plot for outlier analysis*

Description

This routine produces log-ratio plots as in Wilson (1993) using data returned by the command `ap`.

Usage

```
ap.plot(RATIO, NLEN = 25, plot.options = list(pch = 19),
        ylim = NULL, ylab = "log-ratio", xlab = "i", main = "",
        err.check = TRUE)
```

Arguments

`RATIO`	n by NDEL matrix returned by `ap`;
`NLEN`	number of rows in RATIO to be used in plot; default value is 25, the number used in Wilson (1993);
`plot.options`	list of plotting options to be used;
`ylim`	vector of length 2 specifying minimum and maximum values on the vertical axis; default values are obtained when ylim=NULL;
`ylab`	label for vertical axis;
`xlab`	label for horizontal axis;
`main`	an overall title for the plot;
`err.check`	equals TRUE (default) for error-checking, or FALSE for no error-checking.

Author(s)

Paul W. Wilson

References

Wilson, P.W. (1993), Detecing outliers in deterministic nonparametric frontier models with multiple outputs, *Journal of Business and Economic Statistics* 11, 319-323.

See Also

ap, par, plot.

Examples

```
x=matrix(rnorm(25),nrow=1)
y=2+2*x+1.5*rnorm(25)
tmp=ap(X=x,Y=y,NDEL=4)
ap.plot(RATIO=tmp$ratio)
```

---

boot.sw98 *Homogeneous Bootstrap for Shephard (1970) Distance Functions*

Description

This routine implements the bootstrap method of Simar and Wilson (1998) for estimating confidence intervals for Shephard (1970) input and output distance functions as well as for hyperbolic graph distance functions.

Usage

```
boot.sw98(XOBS, YOBS, NREP = 2000, DHAT = NULL,
          RTS = 1, ORIENTATION = 1, alpha = 0.05, CI.TYPE=2,
  XREF = NULL, YREF = NULL, DREF = NULL,
  OUTPUT.FARRELL = FALSE, NOPRINT = FALSE, errchk = TRUE)
```

Arguments

XOBS	a $p \times n$ matrix of observations on p inputs of n firms for which confidence intervals are to be estimated.
YOBS	a $q \times n$ matrix of observations on q outputs of n firms for which confidence intervals are to be estimated.
NREP	number of bootstrap replications to be performed.
DHAT	(optional) vector of n estimates of Shephard distance functions corresponding to (XOBS, YOBS) for which confidence intervals are to be estimated; these efficiency estimates are relative to the technology supported by observations in (XREF, YREF) if different from the technology supported by (XOBS, YOBS).
RTS	indicator for returns to scale (equal 1 for variable returns to scale, 2 for non-increasing returns to scale, or 3 for constant returns to scale).
ORIENTATION	indicator input/output orientation (equals 1 for input orientation, 2 for output orientation, or 3 for hyperbolic graph orientation).

`alpha` either a scalar or a vector length n_a of values giving statistical sizes of the confidence intervals to be estimated.

`CI.TYPE` logical variable indicating method to be used to construct bootstrap confidence interval estimates; `CI.TYPE` can be: 1 - Efron percentile intervals; 2 - Hall percentile intervals based on differences; 3 - Efron's bias-corrected intervals; or 4 - percentile intervals based on ratios.

`XREF` (optional) $p \times n_r$ matrix of observations on p inputs of n_r firms that serve to define the technology.

`YREF` (optional) $q \times n_r$ matrix of observations on q outputs of n_r firms that serve to define the technology.

`DREF` (optional) vector of n_r Shephard distance function estimates corresponding to (XREF, YREF) giving estimates of technical efficiency of each observation in (XREF, YREF) relative to the technology defined by (XREF, YREF).

`OUTPUT.FARRELL` logical flag; if TRUE and ORIENTATON=2, Farrell output efficiency estimates are returned (these will be greater than or equal to 1) along with corresponding bias estimates, bias-corrected efficiency estimates, and confidence interval estimates. If FALSE and ORIENTATION=2, Shephard output distance function estimates are returned (these will be less than or equal to 1) along with corresponding bias estimates, bias-corrected efficiency estimates, and confidence interval estimates. If ORIENTATION=1 then the value of this argument has no effect.

`NOPRINT` logical flag; if TRUE, warning messages are suppressed. If FALSE, warning messages are printed on the console as appropriate.

`errchk` equals TRUE (default) for error-checking, or FALSE for no error-checking.

Details

The homogeneous bootstrap method described by Simar and Wilson (1998) is used to estimate confidence intervals for Shephard (1970) input or output distance functions, or for hyperbolic graph distance functions, corresponding to each observation in (XOBS, YOBS). Distance function estimates from previous computations may be supplied in DHAT (or DREF); if these are not supplied, they are computed automatically. Efficiency estimates are computed using the routine `dea`. Note that distance function estimates are biased; consequently, confidence intervals obtained using Efron's percentile method (`CI.TYPE=1`) should be corrected by subtracting 2 times the estimated bias from both the lower and upper bounds of confidence interval estimates when this method is used; see Simar and Wilson (1998) for details. This problem can be avoided by using `CI.TYPE=2`, 3, or 4. The method described in Simar and Wilson (2000b) corresponds to `CI.TYPE=2`.

Value

A list containing the following items is returned:

`bias` vector of bootstrap bias estimates corresponding to observations in (XOBS,YOBS).

`var` vector of bootstrap variance estimates corresponding to observations in (XOBS,YOBS).

`conf.int` An $(n \times (2n_a))$ matrix containing confidence inteval estimates corresponding to observations in (XOBS,YOBS).

`dhat` efficiency estimates for observations in (XOBS, YOBS) relative to the technology supported by (XREF, YREF), or the technology supported by (XOBS, YOBS) if XREF and YREF are not supplied or are not different from (XOBS, YOBS).

`dhat.bc` bias-corrected efficiency estimates for observations in (XOBS, YOBS) relative to the technology supported by (XREF, YREF), or the technology supported by (XOBS, YOBS) if XREF and YREF are not supplied or are not different from (XOBS, YOBS); equals dhat-bias.

`dref` efficiency estimates for observations in (XREF, YREF) relative to the technology supported by (XREF, YREF).

`bias.flag` logical variable; equals TRUE if any bias-corrected distance function estimates are less than or equal to zero, or FALSE otherwise.

`boot` an $n \times NREP$ matrix containing bootstrap estimates corresponding to the n observations in (XOBS, YOBS).

If alpha is scalar, columns 1:2 of conf.int contain bootstrap estimates of lower and upper bounds of $(1 - alpha)100$-percent confidence intervals for each observation in (XOBS, YOBS). If alpha is a vector, columns 1:2 contain similar information corresponding to the first element of alpha; columns 3:4 contain similar information corresponding tothe second element of alpha; *etc.*

Efficiency estimates returned in dhat and dref are estimated Shephard (1970) input or output distance functions, depending on whether ORIENTATION equals 1 or 2. Note if DHAT or DREF are passed as arguments when `boot.sw98` is called, then the same values are returned in dhat and dref.

Note

In some cases, use of Shephard output distance functions can result in bias-corrected distance function estimates that are negative. This will occur whenever the estimated bias is larger than the distance function estimate. In such cases, a warning message is printed (provided `NOPRINT-FALSE`), and the user is advised to use the Farrell output efficiency measure (which is the reciprocal of the Shephard output distance function) by setting `OUTPUT.FARRELL=TRUE` in the argument list. The problem is avoided since the Farrell output measures are weakly greater than one.

Author(s)

Paul W. Wilson

References

Hadley, G. (1962), *Linear Programming,* Reading, MA: Addison-Wesley Publishing Company, Inc.

Shephard, R.W. (1970), *Theory of Cost and Production Function*, Princeton: Princeton University Press.

Simar, L. and P.W. Wilson (1998), Sensitivity analysis of efficiency scores: How to bootstrap in nonparametric frontier models, *Management Science* 44, 49–61.

Simar, L. and Wilson, P.W. (2000a), Statistical inference in nonparametric frontier models: The state of the art, *Journal of Productivity Analysis* 13, 49-78.

Simar, L. and P.W. Wilson (2000b), A general methodology for bootstrapping in non-parametric frontier models, *Journal of Applied Statistics* 27, 779-802.

See Also

`dea`.

Examples

```
data(ccr)
x=matrix(c(ccr$x1,ccr$x2,ccr$x3,ccr$x4,ccr$x5),nrow=5,ncol=70)
y=matrix(c(ccr$y1,ccr$y2,ccr$y3),nrow=3,ncol=70)
nrep=100 # this is only an example; 100 replications are insufficient
         # to obtain reliable estimates of confidence intervals!
#
# here, estimate 70 CIs:
result=boot.sw98(XOBS=x,YOBS=y,NREP=nrep)
print(result)
#
# here, estimate CIs for only the first 5 observations:
x0=x[,1:5]
y0=y[,1:5]
dhat=result$dhat[1:5]
dref=result$dref
boot.sw98(XOBS=x0,YOBS=y0,DHAT=dhat,XREF=x,YREF=y,DREF=dref,NREP=nrep)
```

`bootstrap.ci` *Compute Bootstrap Confidence Interval Estimates from a Matrix of Bootstrap Estimates*

Description

This function computes several types of bootstrap confidence interval estimates, and is designed for use with efficiency estimates which may be bounded either from above or from below at one. The function can also be used to estimate confidence intervals for Malmquist productivity indices and components, as well as other quantities of interest.

Usage

```
bootstrap.ci(BOOT, alpha =c(0.1,0.05,0.01), BHAT = NULL, DEA = TRUE,
             METHOD = 2, errchk = TRUE)
```

Arguments

`BOOT`	$n \times B$ matrix of B bootstrap estimates corresponding to n (original) estimates.
`alpha`	vector of sizes for confidence intervals.
`BHAT`	optional) a vector of length n containing the original estimates of the quantities of interest.
`DEA`	equals TRUE if the quantities of interest are distance functions, Farrel-type efficiency scores, or other quantities bounded either above or below by 1.
`METHOD`	logical variable indicating method to be used to construct bootstrap confidence interval estimates; `METHOD` can be: 1 - Efron percentile intervals; 2 - Hall percentile intervals based on differences; 3 - Efron's bias-corrected intervals; or 4 - percentile intervals based on ratios.
`errchk`	equals TRUE (default) for error-checing, or FALSE for no error-checking.

Details

Typically, values of alpha will be between 0 and 0.1; the default is to use three values: 0.1, 0.05, and 0.01. Ordinary percentile intervals (METHOD==1) should be avoided when estimating confidence intervals based on Shephard (1970) distance functions or Farrell (1957) efficiency measures due to bias problems. Intervals based on ratios (METHOD==4) are used by Kneip et al. (2003).

Value

bootstrap.ci returns an $(n \times 2k)$ matrix, with each row giving confidence interval estimates corresponding to the n rows n BOOT, where k is the length of alpha. The first k columns of the returned matrix give lower bounds corresponding to the k sizes given in alpha, while the second group of k columns gives corresponding upper bounds.

Author(s)

Paul W. Wilson

References

Farrell, M.J. (1957), The measurement of productive efficiency, *Journal of the Royal Statistical Society Series A* 120, 253-281.

Hall, P. (1992), *The Bootstrap and Edgeworth Expansion*, New York: Springer-Verlag.

Kneip, A., L. Simar, and P.W. Wilson (2003), Asymptotics for DEA Estimators in Nonparametric Frontier Models, Discussion paper \#0317, Institut de Statistique, Universit\'e Catholique de Louvain, Louvain-la-Neuve, Belgium.

Shephard, R.W. (1970), *Theory of Cost and Production Function*, Princeton: Princeton University Press.

Simar, L. and Wilson, P.W. (2000), Statistical inference in nonparametric frontier models: The state of the art, *Journal of Productivity Analysis* 13, 49-78.

Examples

```
n=10
nrep=2000
xbar=rnorm(10)
boot=matrix(rnorm(n*nrep),nrow=n,ncol=nrep) #NOTE: not real bootstrap values!
bootstrap.ci(BOOT=boot,BHAT=xbar,DEA=FALSE,METHOD=1)

n=100
nrep=2000
dhat=rnorm(n)+1.5
dhat=ifelse(dhat<=1,1,dhat)
boot=rnorm(n*nrep)+1.3
boot=ifelse(boot<=1,1,boot)
boot=matrix(boot,nrow=n,ncol=nrep)
bootstrap.ci(BOOT=boot,BHAT=dhat,METHOD=4)
```

bounded.density *Kernel Estimation of Bounded Density*

Description

The generic function bounded.density computes, using the reflection method, kernel estimates bounded densities.

Usage

```
bounded.density(x, bw = "nrd0", lower = NULL, upper = NULL,
        kernel = "gaussian", n = 512, adjust = 1, weights = NULL,
        from, to, cut = 3, na.rm = FALSE)
```

Arguments

x	the data from which the estimate is to be computed.
bw	the smoothing bandwidth to be used. The kernels are scaled such that this is the standard deviation of the smoothing kernel. (Note this differs from the reference books cited below, and from S-PLUS.) bw can also be a character string giving a rule to choose the bandwidth. See bw.nrd. The default, "nrd0", has remained the default for historical and compatibility reasons, rather than as a general recommendation, where e.g., "SJ" would rather fit, see also V&R (2002). The specified (or computed) value of bw is multiplied by adjust.
lower,upper	lower and upper bounds of the support of the density to be estimated. Either one or both may be specified. If neither are specified, the support is assumed unbounded on both the left and the right.
kernel	a character string giving the smoothing kernel to be used. This must be one of "gaussian", "rectangular", "triangular", "epanechnikov", "biweight", "cosine" or "optcosine", with default "gaussian", and may be abbreviated to a unique prefix (single letter). "cosine" is smoother than "optcosine", which is the usual 'cosine' kernel in the literature and almost MSE-efficient. However, "cosine" is the version used by S.
n	the number of equally spaced points at which the density is to be estimated. When n > 512, it is rounded up to a power of 2 during the calculations (as fft is used) and the final result is interpolated by approx. So it almost always makes sense to specify n as a power of two.
adjust	the bandwidth used is actually adjust*bw. This makes it easy to specify values like 'half the default' bandwidth.
weights	numeric vector of non-negative observation weights, hence of same length as x. The default NULL is equivalent to weights = rep(1/nx, nx) where nx is the length of (the finite entries of) x[].
from,to	the left and right-most points of the grid at which the density is to be estimated; the defaults are cut * bw outside of range(x).

cut	by default, the values of from and to are cut bandwidths beyond the extremes of the data. This allows the estimated density to drop to approximately zero at the extremes.
na.rm	logical; if TRUE, missing values are removed from x. If FALSE any missing values cause an error.
...	further arguments for (non-default) methods.

Details

The function bounded.density computes kernel density estimates for densities with bounded support. Its default method does so with the given kernel and bandwidth for univariate observations, using the reflection method described by Silverman (1986) and Simar and Wilson (1998). The estimated density can have either upper, lower, or both upper and lower bounds.

Data are first reflected around the support boundaries (if given). A bandwidth is computed using the reflected data, and then this bandwidth is adjusted upward to reflect the original (unreflected) sample size. A kernel density estimate for the reflected data is computed, and then is reflected around the boundaries to give an estimate of the bounded density. If no boundaries are given, bounded.density computes estimates identical to those computed by density.

Value

An object with class "density" whose underlying structure is a list containing the following components.

x	the n coordinates of the points where the density is estimated.
y	the estimated density values. These will be non-negative, but can be zero.
bw	the bandwidth used.
n	the sample size after elimination of missing values.
call	the call which produced the result.
data.name	the deparsed name of the x argument.
has.na	logical, for compatibility (always FALSE).

The print method reports summary values on the x and y components.

References

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole (for S version).

Scott, D. W. (1992) *Multivariate Density Estimation. Theory, Practice and Visualization*. New York: Wiley.

Sheather, S. J. and Jones M. C. (1991) A reliable data-based bandwidth selection method for kernel density estimation. *J. Roy. Statist. Soc.* **B**, 683–690.

Silverman, B. W. (1986) *Density Estimation*. London: Chapman and Hall.

Simar, L. and P.W. Wilson (1998), Sensitivity analysis of efficiency scores: How to bootstrap in nonparametric frontier models, *Management Science* 44, 49–61.

Venables, W. N. and Ripley, B. D. (2002) *Modern Applied Statistics with S*. New York: Springer.

See Also

density, bw.nrd, plot.density, hist.

Examples

```
require(graphics)

x1=abs(rnorm(500))
plot(bounded.density(x1))

x2=runif(500)
plot(bounded.density(x2))
```

---

ccr *Charnes et al. (1981) Program Follow-Through Data*

Description

The ccr data frame has 70 rows and 9 columns. The data are from Charnes *et al.* (1982).

Usage

```
data(ccr)
```

Format

This data frame contains the following columns:

dmu decision-making unit number

x1 input no. 1: education level of mother as measured in terms of percentage of high school graduates among female parents;

x2 input no. 2: highest occupation of a family member according to a pre-arranged rating scale;

x3 input no. 3: parental visit index representing the number of visits to the school site;

x4 input no. 4: parent counseling index calculated from data on time spent with child on school-related topic such as reading together, etc.;

x5 input no. 5: number of teachers at a given site.

y1 output no. 1: total reading score as measured by the Metropolitan Achievement Test;

y2 output no. 2: total mathematics score as measured by the Metropolitan Achievement Test;

y3 output no. 3: Coopersmith Self-Esteem Inventory, intended as a measure of self-esteem.

Source

Charnes, A., W.W. Cooper, and E. Rhodes (1981), "Evaluating program and managerial efficiency: An application of data envelopment analysis to Program Follow Through, *Management Science* 27, 668–97.

Examples

```
data(ccr)
```

cost.min — *DEA Cost Minimization Problem — Deprecated*

Description

Given matrices of input and output vectors, and a matrix of input prices, this routine computes an estimate of optimal input levels for each of the given input price vectors. Note that this function is deprecated; use dea.cost.min instead.

Usage

```
cost.min(XREF, YREF, XPRICE, YOBS = NULL, RTS = 1, errchk = TRUE)
```

Arguments

XREF	(optional) $p \times n_r$ matrix of observations on p inputs of n_r firms that serve to define the technology.
YREF	(optional) $q \times n_r$ matrix of observations on q outputs of n_r firms that serve to define the technology.
XPRICE	a $p \times n_p$ matrix of input-price vectors, where n_p equals either 1 or n_0 (see description of YOBS below).
YOBS	(optional) $q \times n_0$ matrix of observations on q outputs of n_0 firms for which cost-minimizing inputs are to be estimated; if this argument is not passed, cost-minimizing inputs for the output vectors in YREF are estimated.
RTS	indicator for returns to scale (equal 1 for variable returns to scale, 2 for non-increasing returns to scale, or 3 for constant returns to scale).
errchk	equals TRUE (default) for error-checking, or FALSE for no error-checking.

Details

See Fare *et al.* (1985, pp. 104-105, eqn 4.7.7) for details. Linear programs are solved using the simplex method described by Hadley (1962).

Value

`cost.min` returns a matrix with estimates of cost-minimizing input vectors for each output vector in YOBS (or in YREF if YOBS is not passed as an argument to `cost.min`).

Author(s)

Paul W. Wilson

References

Fare, R., S. Grosskopf, and C.A.K. Lovell (1985), *The Measurement of Efficiency of Production,* Boston: Kluwer-Nijoff Publishing, Inc.

Hadley, G. (1962), *Linear Programming,* Reading, MA: Addison-Wesley Publishing Company, Inc.

Simar, L. and Wilson, P.W. (2000), Statistical inference in nonparametric frontier models: The state of the art, *Journal of Productivity Analysis* 13, 49-78.

See Also

dea profit.max revenue.max

Examples

```
input.prices=matrix(1,nrow=2,ncol=1)
y=matrix(2,nrow=1,ncol=7)
x=matrix(c(1,2,2,2,2,1,1,3,1,4,3,1.25,4,1.25),nrow=2,ncol=7)
cost.min(XREF=x,YREF=y,XPRICE=input.prices)
```

---

cquan *Nonparametric Conditional and Unconditional Alpha-Quantile Estimates*

Description

This function computes nonparametric, conditional input or output alpha-quantile estimates, or nonparametric, unconditional, hyperbolic alpha-quantile esitimates.

Usage

```
cquan(XOBS, YOBS, XREF = NULL, YREF = NULL, alpha = 0.95,
      ORIENTATION = 1, errchk = TRUE)
```

Arguments

XOBS	a $p \times n$ matrix of observations on p inputs of n firms whose efficiency is to be estimated;
YOBS	a $q \times n$ matrix of observations on q outputs of n firms whose efficiency is to be estimated;
XREF	(optional) $p \times n_r$ matrix of observations on p inputs of n_r firms that serve to define the technology;
YREF	(optional) $q \times n_r$ matrix of observations on q outputs of n_r firms that serve to define the technology;
alpha	quantile, in decimal form (must be greater than zero and less than or equal to one);
ORIENTATION	equals 1 for input orientation, 2 for output orientation, or 3 for hyperbolic graph direction;
errchk	equals TRUE (default) for error-checking, or FALSE for no error-checking.

Details

For either the input- or output-oreinted estimates (ORIENTATION=1 or ORIENTATION=2), an exact solution is computed as described by Daouia and Simar (2007). For the unconditional hyperbolic measure (ORIENTATION=3), a numerical solution is computed as described by Wheelock and Wilson (2008), who note that this method is faster than the corresponding exact solution for the hyperbolic orientation.

Value

cquan returns a vector of length n containing estimates of Shephard-type distance functions. Note that these are the reciprocals of the corresponding Farrell measures. For a point contained within the convex hull of the reference obervations, the returned value will be weakly greater than one for ORIENTATION=1 (input orientation) when alpha=1, or weakly less than one for ORIENTATION=2 (output orientation) when alpha=1. For the hyperbolic graph direction, estimates will be weakly greater than one when alpha=1. In this case, a point of interest can be projected onto an estimated quantile by dividing input quantities by the distance function estimate while multiplying output quantities by the same distance function estimate. When alpha<1, returned values may be greater than, equal to, or less than one. In certain cases (e.g., where the point of interest has a zero value for an input or output quantity), a missing value (NA) may be returned if the distance function is not defined for such a point.

Author(s)

Paul W. Wilson

References

Daouia, A. and L. Simar (2007), Nonparametric efficiency analysis: A multivariate conditional quantile approach, *Journal of Econometrics* 140, 375–400.

Wheelock, D.C. and P.W. Wilson (2008), Non-parametric, Unconditional Quantile Estimation for Efficiency Analysis with an Application to Federal Reserve Check Processing Operations, *Journal of Econometrics* 145, 209–225.

See Also

orderm

Examples

```
x=matrix(c(1:100),nrow=1)
y=matrix(c(1:100),nrow=1)
tmp1=cquan(XOBS=x,YOBS=y,ORIENTATION=1)
summary(tmp1)
tmp2=cquan(XOBS=x,YOBS=y,ORIENTATION=2)
summary(tmp2)
tmp3=cquan(XOBS=x,YOBS=y,ORIENTATION=3)
summary(tmp3)
```

---

dea *Compute DEA Efficiency Estimates*

Description

This function computes Shephard (1970) input or output distance functions, under variable, non-increasing, or constant returns to scale.

Usage

```
dea(XOBS, YOBS, RTS = 1, ORIENTATION = 1, XREF = NULL, YREF = NULL,
    DISP = "strong", IS.EFF = NULL, errchk = TRUE)
```

Arguments

`XOBS`	a $p \times n$ matrix of observations on p inputs of n firms whose efficiency is to be estimated.
`YOBS`	a $q \times n$ matrix of observations on q outputs of n firms whose efficiency is to be estimated.
`RTS`	indicator for returns to scale (equal 1 for variable returns to scale, 2 for non-increasing returns to scale, 3 for constant returns to scale, or 4 for non-decreasing returns to scale).
`ORIENTATION`	indicates direction in which efficiency is to be evaluated (equals 1 for input orientation, 2 for output orientation, or 3 for hyperbolic graph orientation).
`XREF`	(optional) $p \times n_r$ matrix of observations on p inputs of n_r firms that serve to define the technology.
`YREF`	(optional) $q \times n_r$ matrix of observations on q outputs of n_r firms that serve to define the technology.
`DISP`	character string equal to either "strong" (default) or "weak", indicating either strong disposability of inputs and outputs or weak disposability of inputs and outputs.
`IS.EFF`	(optional) vector of length n indicating which columns of (XOBS, YOBS) or (XREF, YREF) are to be used to define the frontier estimate.
`errchk`	equals TRUE (default) for error-checking, or FALSE for no error-checking.

Details

For `ORIENTATION=1` or `ORIENTATION=2`, compute data envelopment analysis (DEA) efficiency estimates for each of n DMUs relative to the convex hull of the free-disposal hull of the same n DMUs. Efficiency is measured in terms of Shephard (1970) input or output distance functions, which are the reciprocals of the Farell (1957) input or output efficiency measures. Distance function estimates are computed using the simplex method described by Hadley (1962). For `ORIENTATION=3`, compute the hyperbolic graph measure of technical efficiency using a bisection method.

Value

For `ORIENTATION=1` or `ORIENTATION=2`, `dea` returns a vector of length n containing estimates of Shephard input or output distance functions. Note that these are the reciprocals of the corresponding Farrell measures. For a point contained within the convex hull of the reference obervations, the returned value will be weakly greater than one for ORIENTATION=1 (input orientation) or weakly less than one for ORIENTATION=2 (output orientation). For `ORIENTATION=3`, distance function values will be greater than one for a point contained within the convex hull of the reference observations. In cases where distance function estimates cannot be computed, NA is returned.

Author(s)

Paul W. Wilson

References

Charnes, A., Cooper, W.W. and Rhodes E. (1978), Measuring the inefficiency of decision making units, *European Journal of Operational Research* 2, 429-444.

Farrell, M.J. (1957), The measurement of productive efficiency, *Journal of the Royal Statistical Society Series A* 120, 253-281.

Hadley, G. (1962), *Linear Programming,* Reading, MA: Addison-Wesley Publishing Company, Inc.

Shephard, R.W. (1970), *Theory of Cost and Production Function*, Princeton: Princeton University Press.

Simar, L. and Wilson, P.W. (2000), Statistical inference in nonparametric frontier models: The state of the art, *Journal of Productivity Analysis* 13, 49-78.

Wilson, P.W. (2011), "Asymptotic Properties of some Non-parametric Hyperbolic Efficiency Estimators," in *Exploring Research Frontiers in Contemporary Statistics and Econometrics,* ed. by I. van Keilegom and P. W. Wilson, pp. 115–150, Berlin: Springer-Verlag.

See Also

`fdh`, `orderm`, `genxy`.

Examples

```
data(ccr)
x=t(matrix(c(ccr$x1,ccr$x2,ccr$x3,ccr$x4,ccr$x5),nrow=70,ncol=5))
y=t(matrix(c(ccr$y1,ccr$y2,ccr$y3),nrow=70,ncol=3))
dea(XOBS=x,YOBS=y)
```

`dea.cost.min` *DEA Estimates of Cost Efficiency, Etc.*

Description

Given matrices of input and output vectors, and a matrix of input prices, this routine computes an estimate of optimal input levels for each of the given input price vectors, as well as the cost efficiency of each observed decision-making unit.

Usage

```
dea.cost.min(XOBS, YOBS, XPRICE, RTS = 1, errchk = TRUE)
```

Arguments

`XOBS`	(optional) $p \times n$ matrix of observations on p inputs of n firms that serve to define the technology.
`YOBS`	(optional) $q \times n$ matrix of observations on q outputs of n firms that serve to define the technology.
`XPRICE`	a $p \times m$ matrix of input-price vectors, where m equals either 1 or n (if m equals 1, all decision-making units are assumed to face the same input-price vector).
`RTS`	indicator for returns to scale (equal 1 for variable returns to scale, 2 for non-increasing returns to scale, 3 for constant returns to scale, or 4 for non-decreasing returns to scale).
`errchk`	equals TRUE (default) for error-checking, or FALSE for no error-checking.

Details

See Fare *et al.* (1985, pp. 104-105, eqn 4.7.7) for details. Linear programs are solved using the simplex method described by Hadley (1962).

Value

`dea.cost.min` returns a list containing the following items:

`eff`	a vector of length n containing the cost efficiency estimates, computed as observed cost divided by estimated optimal cost.
`xopt`	matrix with estimates of cost-minimizing input vectors corresponding to each observation in XOBS, YOBS.

Author(s)

Paul W. Wilson

References

Fare, R., S. Grosskopf, and C.A.K. Lovell (1985), *The Measurement of Efficiency of Production,* Boston: Kluwer-Nijoff Publishing, Inc.

Hadley, G. (1962), *Linear Programming,* Reading, MA: Addison-Wesley Publishing Company, Inc.

Simar, L. and Wilson, P.W. (2000), Statistical inference in nonparametric frontier models: The state of the art, *Journal of Productivity Analysis* 13, 49-78.

See Also

`dea`, `dea.profit.max`, `dea.revenue.max`.

Examples

```
input.prices=matrix(1,nrow=2,ncol=1)
y=matrix(2,nrow=1,ncol=7)
x=matrix(c(1,2,2,2,2,1,1,3,1,4,3,1.25,4,1.25),nrow=2,ncol=7)
dea.cost.min(XOBS=x,YOBS=y,XPRICE=input.prices)
```

`dea.direc` *Compute Directional DEA Efficiency Estimates (general version)*

Description

This function computes DEA estimates of directional distance functions under variable, non-increasing, or constant returns to scale.

Usage

```
dea.direc(XOBS, YOBS, XDIREC=XOBS, YDIREC=YOBS,
    RTS = 1, XREF = NULL, YREF = NULL,
    DISP = "strong", IS.EFF = NULL, errchk = TRUE)
```

Arguments

`XOBS`	a $p \times n$ matrix of observations on p inputs of n firms whose efficiency is to be estimated.
`YOBS`	a $q \times n$ matrix of observations on q outputs of n firms whose efficiency is to be estimated.
`XDIREC`	a $p \times n$ matrix of direction vectors corresponding to the input vectors in `XOBS`.
`YDIREC`	a $q \times n$ matrix of direction vectors corresponding to the input vectors in `YOBS`.
`RTS`	indicator for returns to scale (equal 1 for variable returns to scale, 2 for non-increasing returns to scale, or 3 for constant returns to scale).
`XREF`	(optional) $p \times n_r$ matrix of observations on p inputs of n_r firms that serve to define the technology.
`YREF`	(optional) $q \times n_r$ matrix of observations on q outputs of n_r firms that serve to define the technology.
`DISP`	character string equal to either "strong" (default) or "weak", indicating either strong disposability of inputs and outputs or weak disposability of inputs and outputs.
`IS.EFF`	(optional) vector of length n indicating which columns of (XOBS, YOBS) or (XREF, YREF) are to be used to define the frontier estimate.
`errchk`	equals TRUE (default) for error-checking, or FALSE for no error-checking.

Details

This function computes data envelopment analysis (DEA) directional efficiency estimates for each of n DMUs relative to the convex hull of the free-disposal hull of the same n DMUs. Efficiency is measured in terms of the directional distance function defined by Cambers et al. (1996). Estimates are computed using the simplex method described by Hadley (1962).

Value

`dea.dist` returns a vector of length n containing estimates of the directional distance functions. For a point contained within the convex hull of the reference obervations, the returned value will be weakly greater than zero. In cases where distance function estimates cannot be computed, NA is returned; this can happen if direction vectors are not chosen appropriately.

Author(s)

Paul W. Wilson

References

Chambers, R.G., Y. Chung, and R. Fare (1995), "Benefit and Distance Functions," *Journal of Economic Theory* 70, 407-419.

Hadley, G. (1962), *Linear Programming,* Reading, MA: Addison-Wesley Publishing Company, Inc.

Simar, L., A. Vanhems, and P.W. Wilson (2012), "Statistical Inference for DEA Estimators of Directional Distances," *European Journal of Operational Research,* forthcoming.

See Also

`dea`, `fdh`, `orderm`, `genxy`.

Examples

```
data(ccr)
x=t(matrix(c(ccr$x1,ccr$x2,ccr$x3,ccr$x4,ccr$x5),nrow=70,ncol=5))
y=t(matrix(c(ccr$y1,ccr$y2,ccr$y3),nrow=70,ncol=3))
dea.direc(XOBS=x,YOBS=y)
```

`dea.profit.max` *DEA Estimates of Profit Efficiency, Etc.*

Description

Given matrices of input and output vectors, and matrices of input and output prices, this routine computes estimates of optimal input and output levels for each of given pair of input and output price vectors.

Usage

```
dea.profit.max(XOBS, YOBS, XPRICE, YPRICE, RTS = 1, errchk = TRUE)
```

Arguments

`XOBS`	(optional) $p \times n$ matrix of observations on p inputs of n firms that serve to define the technology.
`YOBS`	(optional) $q \times n$ matrix of observations on q outputs of n firms that serve to define the technology.
`XPRICE`	a $p \times m_x$ matrix of input-price vectors, where m_x equals either 1 or n (if m_x equals 1, all decision-making units are assumed to face the same input-price vector).
`YPRICE`	a $q \times m_y$ matrix of output-price vectors, where m_y equals either 1 or n (if m_y equals 1, all decision-making units are assumed to face the same output-price vector).
`RTS`	indicator for returns to scale (equal 1 for variable returns to scale, 2 for non-increasing returns to scale, 3 for constant returns to scale, or 4 for non-decreasing returns to scale).
`errchk`	equals TRUE (default) for error-checking, or FALSE for no error-checking.

Details

See Fare *et al.* (1985, pp. 129-130, eqn 5.8.7) for details. Linear programs are solved using the simplex method described by Hadley (1962).

Value

`dea.profit.max` returns a list containing the following items:

`eff`	A vector of length n containing the profit efficiency estimates, computed as observed profit divided by estimated optimal profit. If estimated optimal profit is zero, or if either observed or estimated optimal profits are negative, a zero value is returned.
`profit.obs`	Observed profits for each decision making unit represented in XOBS, YOBS.

`profit.opt`	Estimated optimal (i.e., maximum) profits for each decision making unit represented in XOBS, YOBS.
`xopt`	Matrix of profit-maximizing input vectors corresponding to observations in XOBS, YOBS.
`yopt`	Matrix of profit-maximizing output vectors ocrresponding to observations in XOBS, YOBS.

Note that the optimal input-output vectors returned in xopt, yopt are not necessarily unique.

Author(s)

Paul W. Wilson

References

Fare, R., S. Grosskopf, and C.A.K. Lovell (1985), *The Measurement of Efficiency of Production,* Boston: Kluwer-Nijoff Publishing, Inc.

Hadley, G. (1962), *Linear Programming,* Reading, MA: Addison-Wesley Publishing Company, Inc.

Simar, L. and Wilson, P.W. (2000), Statistical inference in nonparametric frontier models: The state of the art, *Journal of Productivity Analysis* 13, 49-78.

See Also

`dea`, `dea.cost.min`, `dea.revenue.max`.

Examples

```
input.prices=matrix(3,nrow=1,ncol=1)
output.prices=matrix(c(2,1),nrow=2,ncol=1)
x=matrix(c(2,2,4,6,7,8,9),nrow=1,ncol=7)
y=matrix(c(1.5,1,2,2,3,2,6,6,6,6,7,4,7,4),nrow=2,ncol=7)
dea.profit.max(XOBS=x,YOBS=y,XPRICE=input.prices,YPRICE=output.prices)
```

`dea.resample` *Resample efficiency scores from a smooth density estimate*

Description

Given a set of n efficiency estimates (bounded either above or below at unity) and a bandwidth, this function returns a vector of n values drawn from a kernel estimate of the (bounded) density of the efficiency estimates. The returned values are needed for the homogeneous bootstrap method described by Simar and Wilson (1998). The reflection method is used to avoid the well-known problems of bias and inconsistency of kernel density estimates near boundaries of support.

Usage

```
dea.resample(dist, bw, m = NULL, omit.ones=FALSE)
```

Arguments

- `dist` — vector of n efficiency estimates, bounded either above or below at 1;
- `bw` — bandwith for the kernel density estimate.
- `m` — number of draws to be taken from elements of `dist`; if NULL (the default), then the number of draws taken is n, the length of `dist`.
- `omit.ones` — if FALSE (the default), and elements of `dist` that are identically equal to one are not removed; if TRUE, such values are removed before sampling from `dist`.

Details

See Simar and Wilson (1998) for details.

Value

A vector of n random efficiency scores is returned.

Note

A suitable bandwidth may be obtained by a call to `eff.bw`.

Author(s)

Paul W. Wilson

References

Simar, L. and P.W. Wilson (1998), Sensitivity analysis of efficiency scores: How to bootstrap in nonparametric frontier models, *Management Science* 44, 49–61.

Simar, L. and Wilson, P.W. (2000), Statistical inference in nonparametric frontier models: The state of the art, *Journal of Productivity Analysis* 13, 49-78.

See Also

`eff.bw`.

Examples

```
dhat=rnorm(100)+1
dhat=ifelse(dhat>=1,dhat,1)
h=eff.bw(dhat)
dstar=dea.resample(dhat,h)
```

`dea.revenue.max` *DEA Estimates of Revenue Efficiency, Etc.*

Description

Given matrices of input and output vectors, and a matrix of output prices, this routine computes an estimate of optimal output levels for each of the given output price vectors, as well as the estimated revenue efficiency of each observed decision-making unit.

Usage

```
dea.revenue.max(XOBS, YOBS, YPRICE, RTS = 1, errchk = TRUE)
```

Arguments

XOBS	(optional) $p \times n$ matrix of observations on p inputs of n firms that serve to define the technology.
YOBS	(optional) $q \times n$ matrix of observations on q outputs of n firms that serve to define the technology.
YPRICE	a $q \times m$ matrix of output-price vectors, where m equals either 1 or n (if m equals 1, all decision-making units are assumed to face the same output-price vector).
RTS	indicator for returns to scale (equal 1 for variable returns to scale, 2 for non-increasing returns to scale, 3 for constant returns to scale, or 4 for non-decreasing returns to scale).
errchk	equals TRUE (default) for error-checking, or FALSE for no error-checking.

Details

See Fare *et al.* (1985, pp. 104-105, eqn 4.7.7) for details. Linear programs are solved using the simplex method described by Hadley (1962).

Value

`dea.revenue.max` returns a list containing the following items:

eff	a vector of length n containing the revenue efficiency estimates, computed as observed reveneue divided by estimated optimal revenue.
yopt	a matrix with estimates of revenue-maximizing output vectors corresponding to each observation in XOBS, YOBS.

Author(s)

Paul W. Wilson

References

Fare, R., S. Grosskopf, and C.A.K. Lovell (1985), *The Measurement of Efficiency of Production,* Boston: Kluwer-Nijoff Publishing, Inc.

Hadley, G. (1962), *Linear Programming,* Reading, MA: Addison-Wesley Publishing Company, Inc.

Simar, L. and Wilson, P.W. (2000), Statistical inference in nonparametric frontier models: The state of the art, *Journal of Productivity Analysis* 13, 49-78.

See Also

dea, dea.cost.min, dea.profit.max.

Examples

```
output.prices=matrix(c(1,2),nrow=2,ncol=1)
x=matrix(5,nrow=1,ncol=7)
y=matrix(c(2,5,3,5,4,6,5,4,6,5,6,4,6,2),nrow=2,ncol=7)
dea.revenue.max(XOBS=x,YOBS=y,YPRICE=output.prices)
```

---

dea.subsample — *Estimate confidence intervals for efficiencies estimated by DEA estimators using subsampling*

Description

Given a vector of DEA efficiency estimates and corresponding matrices of observed input and output vecotors, use bootstrap subsampling to estimate confidence intervals for the corresponding true efficiency levels.

Usage

```
  dea.subsample(DHAT, XOBS, YOBS, XREF = NULL, YREF = NULL,
                XDIREC = NULL, YDIREC = NULL,
                RTS = 1, ORIENTATION = 1, DISP = "strong",
itype = 2, nrep = 1000, subsample.sizes = NULL,
seed = NULL, errchk = TRUE)
```

Arguments

DHAT	vector of DEA efficiency estimates corresponding to observations in XOBS and YOBS.
XOBS	a $p \times n$ matrix of n observed input vectors.
YOBS	a $q \times n$ matrix of n observed output vectors.
XREF	(optional) $p \times n_r$ matrix of observations on p inputs of n_r firms that serve to define the technology.
YREF	(optional) $q \times n_r$ matrix of observations on q outputs of n_r firms that serve to define the technology.
XDIREC	(optional) $p \times n$ matrix of direction vectors corresponding to the input vectors in XOBS.
YDIREC	(optional) $q \times n$ matrix of direction vectors corresponding to the input vectors in YOBS.
RTS	indicator for returns to scale (equal 1 for variable returns to scale, 2 for non-increasing returns to scale, 3 for constant returns to scale, or 4 for non-decreasing returns to scale).
ORIENTATION	indicates direction in which efficiency is to be evaluated (equals 1 for input orientation, 2 for output orientation, or 3 for hyperbolic graph orientation).

`DISP` character string equal to either "strong" (default) or "weak", indicating either strong disposability of inputs and outputs or weak disposability of inputs and outputs.

`itype` equal to 1, 2, 3, or 4 to indicate the type of resampling to be done. If 1, use unbalanced resampling with replacement; if 2, use unbalanced resampling without replacement; if 3, use balanced resampling with replacement; if 4, use balanced resampling without replacement. See Simar and Wilson (2011) for details.

`nrep` number of bootstrap replications.

`subsample.sizes` (optional) vector of subsample sizes to be used in estimating confidence intervals.

`seed` (optional) seed value for random number generator (must be a number in the open interval (1,2147483647).

`errchk` equals TRUE (default) for error-checking, or FALSE for no error-checking.

Details

See Simar and Wilson (2011) for details. Arguments `XOBS, YOBS, XREF, YREF, RTS, ORIENTATION`, and `DISP` should be specified exactly as in the call to the `dea` function used to compute the efficiency estimates in `DHAT`. Note that depending on the numbers of inputs, outputs, and observations, computation can take a very long time. For example, in a problem with 5,000 observations, the problem of estimating confidence intervals for each of 5,000 efficiencies can be split into parts by putting subsets of observations in `XOBS, YOBS`, specifying `XREF, YREF` as the full set of 5,000 observed input and output vectors, and running the subparts of the problem on different processors. If `XDIREC` and `YDIREC` are specified, `dea.subsample` assumes that the distance function estimates in `DHAT` are directional distance function estimates (see dea.direc), and the `ORIENTATION` argument is not used. The direction vectors in `XDIREC` and `YDIREC` should be the same as those used to compute the original directional distance function estimates in `DHAT`.

Value

A list containing the following elements is returned:

ci $6 \times n$ matrix of n estimated confidence intervals corresponding to the n efficiency estimates in `DHAT`; the first three rows give lower bounds for confidence interval estimates at .90, .95, and .99 significance, while the last three rows give upper bounds for conficence interval estimates at the same significance levels.

msize $3 \times n$ matrix holding the sub-sample sizes used to compute confidence interval estimates returned in `msize`; the first row corrsponds to .90 significance, the second row to .95 significance, and the last row to .99 significance.

nm Number of different subsample sizes used in estimating confidence intervals.

subsample.sizes vector of subsample sizes used in estimating confidence intervals.

seed value of random number generator seed, to be (optionally) passed as an argument on subsequent calls.

Author(s)

Paul W. Wilson

References

Kneip, A., L. Simar, and P.W. Wilson (2008), Asymptotics and Consistent Bootstraps for DEA Estimators in Non-parametric Frontier Models, *Econometric Theory* 24, 1663–1697.

Simar, L. and P.W. Wilson (2011), Inference by the m out of n Bootstrap in Nonparametric Frontier Models, *Journal of Productivity Analysis* 36, 33–53.

Examples

```
## --not run--##
#
# data(ccr)
# x=t(ccr[,2:6])
# y=t(ccr[,7:9])
# x0=matrix(x[,1],ncol=1)
# y0=matrix(y[,1],ncol=1)
# result=dea.subsample(DHAT=dhat,XOBS=x0,YOBS=y0,XREF=x,YREF=y,ORIENTATION=3)
```

`eff.bw` *Find bandwidth for kernel estimate of bounded density*

Description

This function computes a bandwidth suitable for use in kernel estimates of densities of efficiency estimates that are bounded either above or below at one.

Usage

```
eff.bw(dist, method = 1)
```

Arguments

`dist`	vector of efficiency estimates, bounded (above or below) at 1;
`method`	if equal to 1 (default), use two-stage plug-in method (Sheather and Jones, 1991), or if equal to 2, use unbaised cross-validation.

Details

It is anticipated that bandwidths selected using `eff.bw` will be used in kernel density estimates that rely on the reflection method to avoid problems of bias and inconsistency at the boundary of support. Consequently, the data are reflected around unity before applying either of the bandwidth-selection methods. The resulting bandwidth is then scale up to reflect the fact that only n of the $2n$ observations after reflection are real. The distance function values passed to `eff.bw` are first rounded to four decimal places; the rounded values are then checked to see whether at least one value is less than 1 and at least one value is greater than 1. If so, an error message is printed on the console and execution stops.

Value

A scalar bandwidth is returned.

Author(s)

Paul W. Wilson

References

Scott, D. (1992), *Multivariate Density Estimation: Theory, Practice, and Visualization,* New York: John Wiley & Sons, Inc.

Sheather, S.J., and M.C. Jones (1991), A reliable data-based bandwidth selection method for kernel density estimation, *Journal of the Royal Statistical Society* B, 53, 684–690.

Silverman, B.W. (1986), *Density Estimation for Statistics and Data Analysis,*

See Also

bw.ucv, dpik.

Examples

```
dhat=rnorm(100)+1
dhat=ifelse(dhat>=1,dhat,1)
h=eff.bw(dhat)
```

---

fdh — *Compute FDH efficiency estimates — Deprecated*

Description

This function computes technical efficiency for a set of observed input/output data relative to the Free Disposal Hull (FDH) of a set of reference observations. The set of reference observations may be the same as the observations for which efficiency estimates are desired, or a different set of observations.

Usage

```
fdh(XOBS, YOBS, ORIENTATION = 1, XREF = NULL, YREF = NULL, errchk = TRUE)
```

Arguments

XOBS	a $p \times n$ matrix of observations on inputs of n firms whose efficiency is to be estimated.
YOBS	a $q \times n$ matrix of observations on outputs of n firms whose efficiency is to be estimated.
ORIENTATION	indicator input/output orientation (equals 1 for input orientation, 2 for output orientation, 3 for both input and output orientations, or 4 for hyperbolic orientation).
XREF	(optional) $p \times n_r$ matrix of observations on inputs of n_r firms that serve to define the technology.
YREF	(optional) $q \times n_r$ matrix of observations on outputs of n_r firms that serve to define the technology.
errchk	equals TRUE (default) for error-checking, or FALSE for no error-checking.

Details

`fdh` computes efficiency estimates for each of n DMUs relative to the free-disposal hull of the same n DMUs if `XREF` and `YREF` are not passed as arguments, or alternatively compute efficiency estimates for each of the n DMUs represented in `XOBS` and `YOBS` relative to the free-disposal hull of the observations in `XREF` and `YREF` when these are passed as arguments. When `XREF` and `YREF` are not passed as arguments, the efficiency estimates will be weakly greater than one in the case of input orientation, or weakly less than one in the case of output orientation.

Value

When `ORIENTATION` equals 1 or 2, `fdh` returns a $2 \times n$ matrix; the first row contains the input or output oriented efficiency estimates for the n observations, and the second row contains the number of observations in the reference set that weakly dominate each of the n observations in `XOBS, YOBS`. When `ORIENTATION` equals 3, `fdh` returns a $3 \times n$ matrix; the first row contains the input-oriented efficiency estimates for the n observations, the second row contains the output-oriented efficiency estimates, and the third row contains the number of observations in the reference set that weakly dominate each of the n observations in `XOBS, YOBS`. When `ORIENTATION` equals 4, `FDH` returns (FDH estimate of) the hyperbolic graph measure of technical efficiency in a $1 \times n$ matrix.

Author(s)

Paul W. Wilson

References

Deprins, D., L. Simar, and H. Tulkens (1984), Measuring labor inefficiency in post offices, in *The Performance of Public Enterprises: Concepts and measurements*, ed. by M. Marchand, P. Pestieau and H. Tulkens, Amsterdam, North-Holland, 243-267.

Simar, L. and Wilson, P.W. (2000), Statistical inference in nonparametric frontier models: The state of the art, *Journal of Productivity Analysis* 13, 49-78.

Wheelock, D.C. and P.W. Wilson (2008), Non-parametric, Unconditional Quantile Estimation for Efficiency Analysis with an Application to Federal Reserve Check Processing Operations, *Journal of Econometrics* 145, 209–225.

Wilson (2011), "Asymptotic Properties of some Non-parametric Hyperbolic Efficiency Estimators," in *Exploring Research Frontiers in Contemporary Statistics and Econometrics,* ed. by I. van Keilegom and P. W. Wilson, pp. 115–150, Berlin: Springer-Verlag.

See Also

`dea`, `orderm`.

Examples

```
tmp=genxy(90001,100,2,2)
x=tmp$x
y=tmp$y
rm(tmp)
dhat=fdh(XOBS=x,YOBS=y)
```

fdh.eff *Compute FDH efficiency estimates*

Description

This function computes technical efficiency for a set of observed input/output data relative to the Free Disposal Hull (FDH) of a set of reference observations. The set of reference observations may be the same as the observations for which efficiency estimates are desired, or a different set of observations.

Usage

```
fdh.eff(XOBS, YOBS, ORIENTATION = 1, XREF = NULL, YREF = NULL,
        IS.EFF = NULL, errchk = TRUE)
```

Arguments

XOBS	a $p \times n$ matrix of observations on inputs of n firms whose efficiency is to be estimated.
YOBS	a $q \times n$ matrix of observations on outputs of n firms whose efficiency is to be estimated.
ORIENTATION	indicator input/output orientation (equals 1 for input orientation, 2 for output orientation, 3 for hyperbolic orientation).
XREF	(optional) $p \times n_r$ matrix of observations on inputs of n_r firms that serve to define the technology.
YREF	(optional) $q \times n_r$ matrix of observations on outputs of n_r firms that serve to define the technology.
IS.EFF	(optional) vector of length n indicating which columns of (XOBS, YOBS) or (XREF, YREF) are to be used to define the frontier estimate.
errchk	equals TRUE (default) for error-checking, or FALSE for no error-checking.

Details

`fdh.eff` computes efficiency estimates for each of n DMUs relative to the free-disposal hull of the same n DMUs if `XREF` and `YREF` are not passed as arguments, or alternatively compute efficiency estimates for each of the n DMUs represented in `XOBS` and `YOBS` relative to the free-disposal hull of the observations in `XREF` and `YREF` when these are passed as arguments. When `XREF` and `YREF` are not passed as arguments, the efficiency estimates will be weakly greater than one in the case of input orientation, or weakly less than one in the case of output orientation.

Value

`fdh.eff` returns a vector of n FDH efficiency estimates corresponding to observations in `XOBS` and `YOBS`, where n is the number of columns in `XOBS` and `YOBS`.

Author(s)

Paul W. Wilson

References

Deprins, D., L. Simar, and H. Tulkens (1984), Measuring labor inefficiency in post offices, in *The Performance of Public Enterprises: Concepts and measurements*, ed. by M. Marchand, P. Pestieau and H. Tulkens, Amsterdam, North-Holland, 243-267.

Simar, L. and Wilson, P.W. (2000), Statistical inference in nonparametric frontier models: The state of the art, *Journal of Productivity Analysis* 13, 49-78.

Wheelock, D.C. and P.W. Wilson (2008), Non-parametric, Unconditional Quantile Estimation for Efficiency Analysis with an Application to Federal Reserve Check Processing Operations, *Journal of Econometrics* 145, 209–225.

Wilson (2011), "Asymptotic Properties of some Non-parametric Hyperbolic Efficiency Estimators," in *Exploring Research Frontiers in Contemporary Statistics and Econometrics,* ed. by I. van Keilegom and P. W. Wilson, pp.115–150, Berlin: Springer-Verlag.

See Also

`hquan`, `dea`, `orderm`.

Examples

```
tmp=genxy(n=100,p=2,q=2)
x=tmp$x
y=tmp$y
rm(tmp)
dhat=fdh.eff(XOBS=x,YOBS=y)
```

`fear.cite` *Print FEAR Citation Information*

Description

This function displays the proper citation for use of, or reference to, the FEAR software package.

Usage

```
fear.cite(echo = TRUE)
```

Arguments

`echo`	logical; if TRUE, print the citation information; otherwise, return the citation information in a character variable.

Value

For `echo=TRUE`, fear.cite returns NUll; for `echo=FALSE`, fear.cite returns a character vector.

Author(s)

Paul W. Wilson

See Also

`fear.license`.

Examples

```
fear.cite()
```

`fear.license` *Print FEAR License*

Description

This function displays the license governing legal use of the FEAR software package.

Usage

```
fear.license(echo = TRUE)
```

Arguments

`echo`	logical; if TRUE, print software license; otherwise, return the license information in a character variable.

Value

For `echo=TRUE`, fear.license returns NUll; for `echo=FALSE`, fear.license returns a character vector.

Author(s)

Paul W. Wilson

See Also

`fear.cite`.

Examples

```
fear.license()
```

`gen.unif` *Alternative uniform random number generator*

Description

This function generates pseudo-random uniform numbers on the interval (0,1) using the multiplicative congruential method.

Usage

```
gen.unif(seed=NULL, n)
```

Arguments

`seed`	(optional) seed for generator (must be a number in the open interval (1,2147483647);
`n`	number of random deviates to be generated

Details

gen.unif generates uniform (0,1) deviates using the multiplicative congruential method with modulus $2^{31} - 1$ and multiplier 7^5. This routine is provided as an alternative to R's generators in the base package in order to make it easy to retain the seed value. This is required in order to replicate experiments exactly.

Value

A LIST is returned containing the following elements:

ran	vector of n generated deviates;
seed	value of the generator's seed to be used on the next call.

Author(s)

Paul W. Wilson

References

Lewis, P.A., A.S. Goodman, and J.M. Miller (1969), A pseudo-random number generator for the System/360, *IBM Systems Journal* 8, 136–146.

Press, W.H., B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling (1986), *Numerical Recipes,* Cambridge University Press, Cambridge.

Examples

```
seed=900001
n=20
gen.unif(seed,n)
```

---

genxy — *Generate artificial input-output observations*

Description

This function generates artifical observations on inputs and outputs from a known, simulate data-generating process based on microeconomic theory of the firm.

Usage

```
genxy(seed=NULL, n, p, q)
```

Arguments

seed	(optional) seed value for random number generator (must be a number in the open interval (1,2147483647).
n	number of observations to be generated.
p	number of inputs.
q	number of outputs.

Details

The p inputs for each observation are generated as independent uniform deviates on the interval (10,20) by a call to gen.unif. Next, the inputs $x_1, \ldots, x_p$ for a given observation are used to compute $Y = \prod_{j=1}^{p} x_j^{0.8/p}$. In the case of one output, Y is the output value. In the case of multiple outputs $(q > 1)$, $(q-1)$ uniform $(0, \pi/2)$ deviates $\phi_1, \ldots, \phi_{q-1}$ are generated. Then, y_q is set equal to $\sqrt{Y^2/(\sum_{j=1}^{q-1}\tan^2(\phi_j)+1)}$, and $y_j = y_q \tan\phi_j$ for $j = 1, \ldots, q-1$. In other words, in the case of multiple outputs, the outputs are assumed to be randomly distributed on the surface contained in the positive orthant of a (hyper)sphere centered at the origin and with radius y.

Value

A list is returned, containing the following elements:

x	a $p \times n$ matrix of simulated input values
y	a $q \times n$ matrix of simulated output values
seed	seed value for random number generator

Author(s)

Paul W. Wilson

Examples

```
n=10
p=3
q=2
tmp=genxy(9001,n,p,q)
```

---

genxy.sphere *Generate artificial input-output observations*

Description

This function generates artifical observations on inputs and outputs from a known, simulate data-generating process based on microeconomic theory of the firm.

Usage

```
genxy.sphere(seed=NULL, n, p, q)
```

Arguments

seed	(optional) seed value for random number generator (must be a number in the open interval (1,2147483647)
n	number of observations to be generated
p	number of inputs
q	number of outputs

Details

Simulates draws of n observations on p inputs and q outputs uniformly distributed on the part of a unit sphere centered at (1, ..., 1, 0, ..., 0) (i.e, p 1s and q 0s) lying in the positive orthant. See Wilson (2011) for details on the simulation method.

Value

A list is returned, containing the following elements:

x	a $p \times n$ matrix of simulated input values
y	a $q \times n$ matrix of simulated output values
seed	seed value for random number generator

Author(s)

Paul W. Wilson

References

Wilson, P.W. (2011), "Asymptotic Properties of some Non-parametric Hyperbolic Efficiency Estimators," in *Exploring Research Frontiers in Contemporary Statistics and Econometrics,* ed. by I. van Keilegom and P. W. Wilson, pp. 115–150, Berlin: Springer-Verlag.

Examples

```
n=10
p=3
q=2
tmp=genxy.sphere(9001,n,p,q)
x=tmp$x
y=tmp$y
test=apply((x-1)**2,2,sum)+apply(y**2,2,sum)
test=round(test,6)
which(test!=1)
```

---

hquan — *Nonparametric Conditional and Unconditional Alpha-Quantile Estimates*

Description

This function computes nonparametric, conditional input or output alpha-quantile estimates, or nonparametric, unconditional, hyperbolic alpha-quantile esitimates.

Usage

```
hquan(XOBS, YOBS, XREF = NULL, YREF = NULL, alpha = 0.95,
      ORIENTATION = 1, errchk = TRUE)
```

Arguments

- `XOBS` a $p \times n$ matrix of observations on p inputs of n firms whose efficiency is to be estimated;
- `YOBS` a $q \times n$ matrix of observations on q outputs of n firms whose efficiency is to be estimated;
- `XREF` (optional) $p \times n_r$ matrix of observations on p inputs of n_r firms that serve to define the technology;
- `YREF` (optional) $q \times n_r$ matrix of observations on q outputs of n_r firms that serve to define the technology;
- `alpha` quantile, in decimal form (must be greater than zero and less than or equal to one);
- `ORIENTATION` equals 1 for input orientation, 2 for output orientation, or 3 for hyperbolic graph direction;
- `errchk` equals TRUE (default) for error-checking, or FALSE for no error-checking.

Details

This function is identical to the `cquan` function.

For either the input- or output-oreinted estimates (ORIENTATION=1 or ORIENTATION=2), an exact solution is computed as described by Daouia and Simar (2007). For the unconditional hyperbolic measure (ORIENTATION=3), a numerical solution is computed as described by Wheelock and Wilson (2008), who note that this method is faster than the corresponding exact solution for the hyperbolic orientation.

Value

`hquan` returns a vector of length n containing estimates of Shephard-type distance functions. Note that these are the reciprocals of the corresponding Farrell measures. For a point contained within the convex hull of the reference obervations, the returned value will be weakly greater than one for ORIENTATION=1 (input orientation) when alpha=1, or weakly less than one for ORIENTATION=2 (output orientation) when alpha=1. For the hyperbolic graph direction, estimates will be weakly greater than one when alpha=1. In this case, a point of interest can be projected onto an estimated quantile by dividing input quantities by the distance function estimate while multiplying output quantities by the same distance function estimate. When alpha<1, returned values may be greater than, equal to, or less than one. In certain cases (e.g., where the point of interest has a zero value for an input or output quantity), a missing value (NA) may be returned if the distance function is not defined for such a point.

Author(s)

Paul W. Wilson

References

Daouia, A. and L. Simar (2007), Nonparametric efficiency analysis: A multivariate conditional quantile approach, *Journal of Econometrics* 140, 375–400.

Wheelock, D.C. and P.W. Wilson (2008), Non-parametric, Unconditional Quantile Estimation for Efficiency Analysis with an Application to Federal Reserve Check Processing Operations, *Journal of Econometrics* 145, 209–225.

See Also

orderm

Examples

```
x=matrix(c(1:100),nrow=1)
y=matrix(c(1:100),nrow=1)
tmp1=hquan(XOBS=x,YOBS=y,ORIENTATION=1)
summary(tmp1)
tmp2=hquan(XOBS=x,YOBS=y,ORIENTATION=2)
summary(tmp2)
tmp3=hquan(XOBS=x,YOBS=y,ORIENTATION=3)
summary(tmp3)
```

---

malmquist — *Malmquist Productivity Indices*

Description

This function computes Malmquist productivity indices and various decompositions from component distance function estimates.

Usage

```
malmquist(LIST, alpha = c(0.1,0.05,0.01), CI.TYPE = 2)
```

Arguments

`LIST`	a list of estimates for n firms returned by malmquist.components.
`alpha`	vector of length K containing sizes of confidence intervals to be estimated from bootstrap estimates included in LIST.
`CI.TYPE`	logical variable indicating method to be used to construct bootstrap confidence interval estimates; `CI.TYPE` can be: 1 - Efron percentile intervals; 2 - Hall percentile intervals based on differences; 3 - Efron's bias-corrected intervals; or 4 - percentile intervals based on ratios.

Details

This routine processes the results returned by malmquist.components to computed estimates of Malmquist indices of productivity change, as well as components of productivity change under several decompositions that have been proposed in the literature. If the list returned by malmquist.components includes bootstrap estimates (i.e., if `NREP` > 0 in the call to malmquist.components), then confidence intervals for the various quantities are also estimated. If the list returned by malmquist.components does not include bootstrap estimates, then the arguments `alpha` and `CI.TYPE` are not used.

For a given firm, define the following terms:

- V11variable returns efficiency at time 1 relative to technology at time 1;
- V12variable returns efficiency at time 1 relative to technology at time 2;
- V21variable returns efficiency at time 2 relative to technology at time 1;
- V22variable returns efficiency at time 2 relative to technology at time 2;

- C11constant returns efficiency at time 1 relative to technology at time 1;
- C12constant returns efficiency at time 1 relative to technology at time 2;
- C21constant returns efficiency at time 2 relative to technology at time 1;
- C22constant returns efficiency at time 2 relative to technology at time 2.

The Malmquist productivity index is defined by $(\frac{C21}{C11} \times \frac{C22}{C12})^{1/2}$. Fare et al. (1992) (FGLR) decomposed the Malmquist productivity index into a measure of efficiency change given by $\frac{C22}{C11}$ and change in technology given by $((\frac{C21}{C22} \times (\frac{C11}{C12}))^{1/2}$. Fare et al. (1994) (FGNZ) decomposed the Malmquist productivity index into three parts, representing (i) change in pure efficiency given by $\frac{V22}{V11}$, (ii) change in scale efficiency given by $\frac{C22/V22}{C11/V11}$, and (iii) change in technology (this term is identical to the one proposed by FGLR). Ray and Desli (1997) decomposed the Malmquist productivity index into (i) change in pure efficiency, (ii) pure technical change given by $(\frac{V21}{V22} \times \frac{V11}{V12})^{1/2}$ and a term they labelled SCH, given by $(\frac{C21/V21}{C12/V12} \times \frac{C22/V22}{C11/V11})^{1/2}$. Simar and Wilson (1998) (SW) and Wheelock and Wilson (1999) (WW) decomposed the Malmquist productivity index into four parts, including (i) change in pure efficiency, (ii) pure technical change, (iii) change in scale efficiency (identical to the term appearing in the FGLR and FGNZ decompositions), and (iv) a term labelled change in scale of technology, given by $(\frac{C21/V21}{C22/V22} \times \frac{C11/V11}{C12/V12})^{1/2}$ the square root of (((C21/V21)/(C22/V22))((C21/V21)/(C22/V22))). This term is the product of the FGLR change in efficiency term and pure technical change, while SCH is the product of change in scale efficiency and change in scale of technology.

Value

A list is returned. If input argument LIST was obtained by a call to malmquist.components with `NREP=0`, then the returned list includes the following items:

`id` a vector of length n containing identifiers for each firm; these are the same as in LIST.

`malm` a vector of length n containing estimates of the Malmquist productivity index for each firm.

`eff` a vector of length n containing estimates of efficiency change as defined by FGLR for each firm.

`tech` a vector of length n containing estimates of technical change as defined by FGLR for each firm.

`pure.eff` a vector of length n containing estimates of pure efficiency change as defined by FGNZ for each firm.

`scale` a vector of length n containing estimates of change in scale efficiency as defined by FGNZ for each firm.

`pure.tech` a vector of length n containing estimates of pure technical change as defined by RD, SW, and WW for each firm.

`scale.tech` a vector of length n containing estimates of changes in scale of technology as defined by SW and WW for each firm.

`sch` a vector of length n containing estimates of the residual SCH term as defined by RD for each firm.

If the input argument LIST was obtained by a call to malmquist.components with `NREP` > 0, then the returned list includes the following items in addition to those listed above:

`ci.malm` an $(n \times 2K)$ matrix of confidence interval estimates for the Malmquist productivity indices for each of n firms.

`ci.eff` an $(n \times 2K)$ matrix of confidence interval estimates for efficiency change (FGLR definition) for each of n firms.

`ci.tech` an $(n \times 2K)$ matrix of confidence interval estimates for technical change (FGLR definition) for each of n firms.

`ci.pure.eff` an $(n \times 2K)$ matrix of confidence interval estimates for pure efficiency change (FGNZ definition) for each of n firms.

`ci.scale` an $(n \times 2K)$ matrix of confidence interval estimates for change in scale efficiency (FGNZ definition) for each of n firms.

`ci.pure.tech` an $(n \times 2K)$ matrix of confidence interval estimates for pure technical change (RD, SW, WW definition) for each of n firms.

`ci.scale.tech` an $(n \times 2K)$ matrix of confidence interval estimates for change in scale of technology (SW, WW definition) for each of n firms.

`ci.sch` an $(n \times 2K)$ matrix of confidence interval estimates for residual term defined by RD.

Note

Bootstrap confidence interval estimates are obtained by calls to bootstrap.ci. Typically, values of `alpha` will be between 0 and 0.1; the default is to use three values: 0.1, 0.05, and 0.01. Returned matrices containing confidence interval estimates will have dimension $n \times 2k$; the first k columns give lower bounds corresponding to the k sizes given in `alpha`, while the second group of k columns gives corresponding upper bounds.

Author(s)

Paul W. Wilson

References

Fare, R., S. Grosskopf, B. Lindgren, and P. Roos (1992), Productivity changes in Swedish pharmacies 1980–1989: A non-parametric Malmquist approach, *Journal of Productivity Analysis* 3, 85-101.

Fare, R., S. Grosskopf, M. Norris, and Z. Zhang (1994), Productivity growth, technical progress, and efficiency change in industrialized countries, *American Economic Review* 84, 66-83.

Ray, S.C. and E. Desli (1997), Productivity growth, technical progress, and efficiency change in industrialized countries: Comment, *American Economic Review* 87, 1033-1039.

Simar, L. and P.W. Wilson (1998), "Productivity Growth in Industrialized Countries," Discussion Paper \#9810, Institut de Statistique, Universit\'e Catholique de Louvain, Louvain-la-Neuve, Belgium.

Simar, L. and Wilson, P.W. (1999), Estimating and bootstrapping Malmquist indices, *European Journal of Operational Research* 115, 459-471.

Wheelock, D.C. and P.W. Wilson (1999), Technical progress, inefficiency, and productivity change in U.S. banking, 1984–1993, *Journal of Money, Credit, and Banking* 31, 212-234.

See Also

dea malmquist.components

Examples

```
#
# input orientation:
tmp=genxy(seed=900001,n=100,p=1,q=1)
x1=matrix(tmp$x*exp(0.2*abs(rnorm(100))),nrow=1)
y1=matrix(tmp$y,nrow=1)
tmp=genxy(seed=900001,n=90,p=1,q=1)
x2=matrix(tmp$x*exp(0.2*abs(rnorm(90))),nrow=1)
y2=matrix(tmp$y*(1+runif(90)*0.2),nrow=1)
id1=c(1:100)
id2=c(11:80,101:120)
m1=malmquist.components(X1=x1,Y1=y1,ID1=id1,X2=x2,Y2=y2,ID2=id2,
                        ORIENTATION=1,NREP=200)
tmp1=malmquist(LIST=m1,alpha=c(0.1,0.05,0.01),CI.TYPE=1)
tmp2=malmquist(LIST=m1,alpha=c(0.1,0.05,0.01),CI.TYPE=2)
tmp3=malmquist(LIST=m1,alpha=c(0.1,0.05,0.01),CI.TYPE=3)
tmp4=malmquist(LIST=m1,alpha=c(0.1,0.05,0.01),CI.TYPE=4)
#
# output orientation:
tmp=genxy(seed=900001,n=100,p=1,q=1)
x1=matrix(tmp$x,nrow=1)
y1=matrix(tmp$y*exp(-0.2*abs(rnorm(100))),nrow=1)
tmp=genxy(seed=900001,n=90,p=1,q=1)
x2=matrix(tmp$x,nrow=1)
y2=matrix(tmp$y*(1+runif(90)*0.2)*exp(-0.2*abs(rnorm(90))),nrow=1)
id1=c(1:100)
id2=c(11:80,101:120)
m2=malmquist.components(X1=x1,Y1=y1,ID1=id1,X2=x2,Y2=y2,ID2=id2,
                        ORIENTATION=2,NREP=20)
tmp=malmquist(LIST=m2,alpha=c(0.1,0.05,0.01))
#
# hyperbolic orientation:
# tmp=genxy(seed=900001,n=100,p=1,q=1)
# d1=exp(0.2*abs(rnorm(100)))
# x1=matrix(tmp$x*d1,nrow=1)
# y1=matrix(tmp$y/d1,nrow=1)
# tmp=genxy(seed=900001,n=90,p=1,q=1)
# d2=exp(0.2*abs(rnorm(90)))
# x2=matrix(tmp$x*d2,nrow=1)
# y2=matrix(tmp$y*(1+runif(90)*0.2)/d2,nrow=1)
# id1=c(1:100)
# id2=c(11:80,101:120)
# m3=malmquist.components(X1=x1,Y1=y1,ID1=id1,X2=x2,Y2=y2,ID2=id2,
#                         ORIENTATION=3,NREP=20)
# tmp=malmquist(LIST=m3,alpha=c(0.1,0.05,0.01))
```

`malmquist.components` *Compute Components of Malmquist Productivity Indices*

Description

This function computes distance function estimates needed to construct Malmquist productivity indices, using either balanced or unbalanced panel data.

Usage

```
malmquist.components(X1, Y1, ID1, X2, Y2, ID2, ORIENTATION = 1, NREP = 0,
                     errchk = TRUE)
```

Arguments

`X1`	$p \times n_1$ matrix of observations on p inputs of n_1 firms operating in period 1.
`Y1`	$q \times n_1$ matrix of observations on q outputs of n_1 firms operating in period 1.
`ID1`	a vector of length n_1 containing unique, identifying labels for the n firms represented by columns of `X1` and `Y1`.
`X2`	$p \times n_2$ matrix of observations on p inputs of n_2 firms operating in period 2.
`Y2`	$q \times n_2$ matrix of observations on q outputs of n_2 firms operating in period 2.
`ID2`	a vector of length n_2 containing unique, identifying labels for the n firms represented by columns of `X2` and `Y2`.
`ORIENTATION`	indicates direction in which efficiency is to be evaluated for purposes of constructing Malmquist indices (equals 1 for input orientation, 2 for output orientation, or 3 for hyperbolic graph orientation).
`NREP`	(optional) Number of bootstrap replications to be performed for inference-making purposes.
`errchk`	(optional) Perform error checking if TRUE; do not check for errors if FALSE.

Details

Calls are made to `dea` to compute estimates of efficiency under both constant as well as variable returns to scale for (i) each of the n_1 firms in period 1, relative to estimates of the technology at time 1; (ii) each of the n_1 firms in period 1, relative to estimates of the technology at time 2; (iii) each of the n_2 firms in period 2, relative to estimates of the technology at time 1; and (iv) each of the n_2 firms in period 2, relative to estimates of the technology at time 2, For each firm, as many as 8 estimates are computed (fewer estimates will be available in some cases, as distance function estimates may not be feasible for some of the cross-period cases. In all cases, all of the n_1 observations available for period 1 are used to estimate the technology in period 1. Similarly, all of the n_2 observatiosn available for period 2 are used to estimate the technology in period 2. Note that some firms that appear in period 1 may be absent in period 2, and some that appear in period 2 may be absent in period 1; for such firms, Malmquist indices cannot be estimated. If `NREP` > 0, bootstrap estimates are computed using the algorithm described by Simar and Wilson (1999), with some small modifications (see below).

Value

Let S be the set of firms that appear in both periods 1 and 2. If `NREP` $= 0$, a list containing the following elements is returned:

`id`	a vector of length n containing unique identifiers from input arguments `ID1` and `ID2` representing those n firms that appear in both periods 1 and 2.
`v11`	a vector of length n containing distance function estimates (under varying returns to scale) for the n firms in period 1 and listed in `ID` relative to estimated technology in period 1.
`v22`	a vector of length n containing distance function estimates (under varying returns to scale) for the n firms in period 2 and listed in `ID` relative to estimated technology in period 2.

v12 a vector of length n containing distance function estimates (under varying returns to scale) for the n firms in period 1 and listed in ID relative to estimated technology in period 2.

v21 a vector of length n containing distance function estimates (under varying returns to scale) for the n firms in period 2 and listed in ID relative to estimated technology in period 1.

c11 a vector of length n containing distance function estimates (under constant returns to scale) for the n firms in period 1 and listed in ID relative to estimated technology in period 1.

c22 a vector of length n containing distance function estimates (under constant returns to scale) for the n firms in period 2 and listed in ID relative to estimated technology in period 2.

c12 a vector of length n containing distance function estimates (under constant returns to scale) for the n firms in period 1 and listed in ID relative to estimated technology in period 2.

c21 a vector of length n containing distance function estimates (under constant returns to scale) for the n firms in period 2 and listed in ID relative to estimated technology in period 1.

If NREP > 0, a list is returned containing the above items as well as the following items:

bv11 an $n \times$ NREP matrix of bootstrap estimates corresponding to the n estimates in v11.

bv22 an $n \times$ NREP matrix of bootstrap estimates corresponding to the n estimates in v22.

bv12 an $n \times$ NREP matrix of bootstrap estimates corresponding to the n estimates in v12.

bv21 an $n \times$ NREP matrix of bootstrap estimates corresponding to the n estimates in v21.

bc11 an $n \times$ NREP matrix of bootstrap estimates corresponding to the n estimates in c11.

bc22 an $n \times$ NREP matrix of bootstrap estimates corresponding to the n estimates in c22.

bc12 an $n \times$ NREP matrix of bootstrap estimates corresponding to the n estimates in c12.

bc21 an $n \times$ NREP matrix of bootstrap estimates corresponding to the n estimates in c21.

Note

In Simar and Wilson (1999), data are projected onto constant-returns estimates of production-set boundaries and then projected away from the boundaries to obtain bootstrap pseudo-data. In malmquist.components, data are first projected onto variable-returns estimates of production-set boundaries, and then projected away from the boundaries using draws from a nonparametric estimate of the joint density of variable-returns distance function estimates.

Author(s)

Paul W. Wilson

References

Simar, L. and Wilson, P.W. (1999), Estimating and bootstrapping Malmquist indices, *European Journal of Operational Research* 115, 459-471.

See Also

dea malmquist

Examples

```
tmp=genxy(seed=900001,n=100,p=1,q=1)
x1=matrix(tmp$x*exp(0.2*abs(rnorm(100))),nrow=1)
y1=matrix(tmp$y,nrow=1)
tmp=genxy(seed=900001,n=90,p=1,q=1)
x2=matrix(tmp$x*exp(0.2*abs(rnorm(90))),nrow=1)
y2=matrix(tmp$y*(1+runif(90)*0.2),nrow=1)
id1=c(1:100)
id2=c(11:80,101:120)
m1=malmquist.components(X1=x1,Y1=y1,ID1=id1,X2=x2,Y2=y2,ID2=id2,
                        ORIENTATION=1)
```

---

orderm — *Compute Order-m Efficiency Estimates*

Description

This function computes estimates of the Cazals *et al.* (2002) order-m efficiency measure for a set of observed input/output data relative to the order-*m* frontier based on a set of reference observations. The set of reference observations may be the same as the observations for which efficiency estimates are desired, or a different set of observations.

Usage

```
orderm(XOBS, YOBS, ORIENTATION = 1, M = 25, NREP = 200,
       XREF = NULL, YREF = NULL, errchk = TRUE)
```

Arguments

XOBS	a $p \times n$ matrix of observations on p inputs of n firms whose efficiency is to be estimated;
YOBS	a $q \times n$ matrix of observations on q outputs of n firms whose efficiency is to be estimated;
ORIENTATION	indicator for input or output orientation (equals 1 for input orientation, 2 for output orientation, or 3 for both orientations);
M	order of the reference frontier;
NREP	number of Monte Carlo replications used in computing the order-m estimates;
XREF	(optional) $p \times n_r$ matrix of observations on p inputs of n_r firms that serve to define the technology;
YREF	(optional) $q \times n_r$ matrix of observations on q outputs of n_r firms that serve to define the technology;
errchk	equals TRUE (default) for error-checking, or FALSE for no error-checking.

Details

See Cazals *et al.* (2002) for detailed description of this estimator. `orderm` computes efficiency estimates for each of n DMUs relative to the order-m frontier estimated from a reference set of observations. If `XREF,YREF` are not passed as arguments, the reference set of observations are those in `XOBS,YOBS`. Alternatively, a different reference set of observations may be specified in `XREF,YREF`.

Value

When `ORIENTATION` equals 1 or 2, `orderm` returns a $2 \times n$ matrix; the first row contains the input or output oriented order-m efficiency estimates for the n observations, and the second row contains corresponding estimates of standard error. When `ORIENTATION` equals 3, `orderm` returns a $4 \times n$ matrix; the first row contains the input-oriented order-m efficiency estimates for the n observations, the second row contains the output-oriented efficiency estimates, the third row contains the standard error estimates corresponding to the input-oriented efficiency estimates, and the fourth row contains the the standard error estimates corresponding to the output-oriented efficiency estimates.

Author(s)

Paul W. Wilson

References

Cazals, C., J.P. Florens, and L. Simar (2002), Nonparametric frontier estimation: A robust approach, *Journal of Econometrics* 106, 1–25.

Wheelock, D.C. and P.W. Wilson (2003), "Robust Nonparametric Estimation of Efficiency and Technical Change in U.S. Commercial Banking," unpublished working paper, Department of Economics, University of Texas, Austin, Texas 78712 USA.

See Also

`dea`, `fdh`.

Examples

```
tmp=genxy(90001,100,2,2)
x=tmp$x
y=tmp$y
rm(tmp)
dhat=orderm(XOBS=x,YOBS=y)
```

`profit.max` *DEA Profit Maximization Problem — Deprecated*

Description

Given matrices of input and output vectors, and matrices of input and output prices, this routine computes estimates of optimal input and output levels for each of given pair of input and output price vectors. Note that this function is deprecated; use dea.profit.max instead.

Usage

```
profit.max(XREF, YREF, XPRICE, YPRICE, RTS = 1, errchk = TRUE)
```

Arguments

XREF	(optional) $p \times n_r$ matrix of observations on p inputs of n_r firms that serve to define the technology.
YREF	(optional) $q \times n_r$ matrix of observations on q outputs of n_r firms that serve to define the technology.
XPRICE	a $p \times n_p$ matrix of input-price vectors, where n_p equals either 1 or n_0.
YPRICE	a $q \times n_p$ matrix of output-price vectors, where n_p equals either 1 or n_0.
RTS	indicator for returns to scale (equal 1 for variable returns to scale, 2 for non-increasing returns to scale, or 3 for constant returns to scale).
errchk	equals TRUE (default) for error-checking, or FALSE for no error-checking.

Details

See Fare *et al.* (1985, pp. 129-130, eqn 5.8.7) for details. Linear programs are solved using the simplex method described by Hadley (1962).

Value

profit.max returns a list:

x	matrix of profit-maximizing input vectors;
y	matrix of profit-maximizing output vectors.

Author(s)

Paul W. Wilson

References

Fare, R., S. Grosskopf, and C.A.K. Lovell (1985), *The Measurement of Efficiency of Production,* Boston: Kluwer-Nijoff Publishing, Inc.

Hadley, G. (1962), *Linear Programming,* Reading, MA: Addison-Wesley Publishing Company, Inc.

Simar, L. and Wilson, P.W. (2000), Statistical inference in nonparametric frontier models: The state of the art, *Journal of Productivity Analysis* 13, 49-78.

See Also

cost.min dea revenue.max

Examples

```
  input.prices=matrix(3,nrow=1,ncol=1)
output.prices=matrix(c(2,1),nrow=2,ncol=1)
x=matrix(c(2,2,4,6,7,8,9),nrow=1,ncol=7)
y=matrix(c(1.5,1,2,2,3,2,6,6,6,6,7,4,7,4),nrow=2,ncol=7)
profit.max(XREF=x,YREF=y,XPRICE=input.prices,YPRICE=output.prices)
```

resample.xy *Draw a boostrap (sub)sample from input/output matrices*

Description

Given a matrix x of observed input vectors and a matrix y of observed output vectors, each with n columns where n is the number of observations, draw a bootstrap sample of size m by drawing independently, either with or without replacement, from columns of x and y such that on each draw, each column of x and its corresponding column in y have equal probability of selection.

Usage

```
resample.xy(x, y, x0 = NULL, y0 = NULL, n = NULL, m, itype,
            seed = NULL, errchk = TRUE)
```

Arguments

x	a $p \times n$ matrix of n observed input vectors.
y	a $q \times n$ matrix of n observed output vectors.
x0	input vector of length p used to divide the sample for balanced resampling.
y0	output vector of length q used to divide the sample for balanced resampling.
n	(optional) the number of observations in x and y; if NULL (the default), then n is set equal to the number of columns in x and y.
m	number of draws to be taken from x and y.
itype	equal to 1, 2, 3, or 4 to indicate the type of resampling to be done. If 1, use unbalanced resampling with replacement; if 2, use unbalanced resampling without replacement; if 3, use balanced resampling with replacement; if 4, use balanced resampling without replacement. See Simar and Wilson (2011) for details.
seed	(optional) seed value for random number generator (must be a number in the open interval (1,2147483647).
errchk	equals TRUE (default) for error-checking, or FALSE for no error-checking.

Details

See Simar and Wilson (2011) for details. Note that m can be less than, equal to, or greater than n. The arguments x0 and y0 are only used if itype equals 3 or 4.

Value

A list containing the following elements is returned:

xstar $p \times m$ matrix of n resampled input vectors.

ystar $p \times m$ matrix of n resampled output vectors.

seed value of random number generator seed, to be (optionally) passed as an argument on subsequent calls.

Author(s)

Paul W. Wilson

References

Kneip, A., L. Simar, and P.W. Wilson (2008), Asymptotics and Consistent Bootstraps for DEA Estimators in Non-parametric Frontier Models, *Econometric Theory* 24, 1663–1697.

Simar, L. and P.W. Wilson (2011), Inference by the m out of n Bootstrap in Nonparametric Frontier Models, *Journal of Productivity Analysis* 36, 33–53.

Examples

```
x=matrix(runif(50),nrow=3,ncol=25)
y=matrix(runif(50),nrow=2,ncol=25)
xystar=resample.xy(x=x,y=y,m=15,itype=2)
```

`revenue.max` *DEA Revenue Maximization Problem — Deprecated*

Description

Given matrices of input and output vectors, and a matrix of output prices, this routine computes an estimate of optimal output levels for each of the given output price vectors. Note that this function is deprecated; use dea.revenue.max instead.

Usage

```
revenue.max(XREF, YREF, XOBS = NULL, YPRICE, RTS = 1, errchk = TRUE)
```

Arguments

`XREF`	(optional) $p \times n_r$ matrix of observations on p inputs of n_r firms that serve to define the technology.
`YREF`	(optional) $q \times n_r$ matrix of observations on q outputs of n_r firms that serve to define the technology.
`YPRICE`	a $q \times n_p$ matrix of output-price vectors, where n_p equals either 1 or n_0 (see description of XOBS below).
`XOBS`	(optional) $p \times n_0$ matrix of observations on p inputs of n_0 firms for which revenue-maximizing outputs are to be estimated; if this argument is not passed, revenue-maximizing outputs for the input vectors in XREF are estimated.
`RTS`	indicator for returns to scale (equal 1 for variable returns to scale, 2 for non-increasing returns to scale, or 3 for constant returns to scale).
`errchk`	equals TRUE (default) for error-checking, or FALSE for no error-checking.

Details

See Fare *et al.* (1985, pp. 104-105, eqn 4.7.7) for details. Linear programs are solved using the simplex method described by Hadley (1962).

Value

`revenue.max` returns a matrix with estimates of revenue-maximizing output vectors for each input vector in XOBS (or in XREF if XOBS is not passed as an argument to `revenue.max`).

Author(s)

Paul W. Wilson

References

Fare, R., S. Grosskopf, and C.A.K. Lovell (1985), *The Measurement of Efficiency of Production,* Boston: Kluwer-Nijoff Publishing, Inc.

Hadley, G. (1962), *Linear Programming,* Reading, MA: Addison-Wesley Publishing Company, Inc.

Simar, L. and Wilson, P.W. (2000), Statistical inference in nonparametric frontier models: The state of the art, *Journal of Productivity Analysis* 13, 49-78.

See Also

`cost.min` `dea` `revenue.max`

Examples

```
output.prices=matrix(c(1,2),nrow=2,ncol=1)
x=matrix(5,nrow=1,ncol=7)
y=matrix(c(2,5,3,5,4,6,5,4,6,5,6,4,6,2),nrow=2,ncol=7)
revenue.max(XREF=x,YREF=y,YPRICE=output.prices)
```

`rnorm.trunc` *Truncated Normal Distribution*

Description

This function generates random deviates from a normal distribution with mean `mean` and standard deviation `sigma`, and with left-truncation at `t.left` and right-truncation at `t.right`.

Usage

```
rnorm.trunc(n, t.left = 0, t.right = Inf, mean = 0, sigma = 1)
```

Arguments

`n`	number of deviates to be generated.
`t.left`	left truncation point(s); may be either a scalar or a vector of lenght `n`.
`t.right`	right truncation point(s); may be either a scalar or a vector of length `n`.
`mean`	mean parameter of the original distribution before truncation.
`sigma`	standard deviation of the original distribution before truncation.

Details

If `mean` or `sigma` are not specified, they assume default values of 0 and 1, respectively. If `t.left` and `t.right` are not specified, they assume values of 0 and positive infinity, respectively; in this case, draws are taken from a half-normal distribution with non-negative support. Deviates are generated using a transformation method, which is implemented with a call to `runif`.

Value

Random deviates are returned in a vector of length n.

Author(s)

Paul W. Wilson

See Also

runif, set.seed.

Examples

```
rnorm.trunc(n=20)
```

---

sexton *Modified Program Follow-Through Data from Sexton et al. (1986)*

Description

The sexton data frame has 70 rows and 9 columns. The data are identical to those in ccr, except that observation 28 for the third output has been deliberately miscoded as discussed by Sexton et al.

Usage

```
data(sexton)
```

Format

This data frame contains the following columns:

dmu decision-making unit number

x1 input no. 1: education level of mother as measured in terms of percentage of high school graduates among female parents;

x2 input no. 2: highest occupation of a family member according to a pre-arranged rating scale;

x3 input no. 3: parental visit index representing the number of visits to the school site;

x4 input no. 4: parent counseling index calculated from data on time spent with child on school-related topic such as reading together, etc.;

x5 input no. 5: number of teachers at a given site.

y1 output no. 1: total reading score as measured by the Metropolitan Achievement Test;

y2 output no. 2: total mathematics score as measured by the Metropolitan Achievement Test;

y3 output no. 3: Coopersmith Self-Esteem Inventory, intended as a measure of self-esteem.

Source

Chames, A., W.W. Cooper, and E. Rhodes (1981), "Evaluating program and managerial efficiency: An application of data envelopment analysis to Program Follow Through, *Management Science* 27, 668–97.

Sexton, T.R., R.H. Silkman, and A.J. Hogan (1986), Data envelopment analysis: Critique and extensions, in *Measuring Efficiency: An Assessment of Data Envelopment Analysis,* R.H. Silkman (ed.), Jossey-Bass Co., San Francisco.

Examples

```
data(sexton)
```

`show.dens` *Plot estimated density of bounded efficiency estimates*

Description

This function plots a kernel estimate of the density of a set of efficiency estimates bounded either above or below at unity.

Usage

```
show.dens(dist, bw, XLIM = NULL, YLIM = NULL, show.plot = TRUE,
          XLAB = "distance function estimate", YLAB = "density")
```

Arguments

`dist`	vector of efficiency estimates, bounded either above or below at 1;
`bw`	bandwith for the kernel density estimate;
`XLIM`	(optional) range of efficiency estimates over which the density estimate is to be plotted (defaults to the range of the estimates in `dist`);
`YLIM`	(optional) range for density that is to be plotted (defaults to range of the estimated density);
`show.plot`	display plot on screen if true; otherwise, return values that can be passed to a plotting command;
`XLAB`	label for the horizontal axis;
`YLAB`	lable for the vertical axis.

Details

Given a set of (bounded at unity) estimates in `dist` and a bandwidth, `show.dens` computes a kernel density estimate and plots the estimate. It is well known that kernel density estimates are biased and inconsistent near boundaries of support; to avoid this problem, the `show.dens` employs the reflection method described by Silverman (1986) and Scott (1992).

Value

A list (x,y) is returned, with both equal to `NULL` if `show.dens` is TRUE. Otherwise, x is a vector of length 256 giving abcissa values and y is a vector of length 256 giving ordinate values that may be passed to a plotting routine to render a plot of the kernel density estimate computed by `show.dens`.

Author(s)

Paul W. Wilson

References

Scott, D. (1992), *Multivariate Density Estimation: Theory, Practice, and Visualization,* New York: John Wiley & Sons, Inc.

Silverman, B.W. (1986), *Density Estimation for Statistics and Data Analysis,* Chapman and Hall, London.

See Also

`dea`, `fdh`, `eff.bw`, `plot`.

Examples

```
dhat=rnorm(100)+1
dhat=ifelse(dhat>=1,dhat,1)
h=eff.bw(dhat)
show.dens(dhat,h)
```

---

treg *Truncated Regression*

Description

Estimate a truncated normal regression equation using the method of maximum likelihood.

Usage

```
treg(Y, X, TPOINT = 1, tol = 1e-05, maxit = 100, iovar = 0,
          err.check = TRUE)
```

Arguments

`Y`	vector of length n containing observations on the truncated dependent variable.
`X`	matrix of regressors with n rows.
`TPOINT`	scalar truncation point such that $Y >$ `TPOINT`.
`tol`	convergence tolerance.
`maxit`	maximum number of iterations.
`iovar`	if 0, information about each iteration is suppressed; if 1, values of the gradient and step size are written to the console on each iteration.
`err.check`	if TRUE, check for errors in arguments; if FALSE, do not check for errors.

Details

The log-likelihood is maximized using a Newton method. The right-hand side data in X should be scaled so that the data do not differ by too many orders of magnitude from 1; otherwise, acheiving convergence may be difficult.

Value

A list is returned containing the following components:

bhat	vector of estimates of intercept and slope parameters; length is same as number of columns in X.
sighat	estimate of square-root of the variance parameter.
cov	estimated variance-covariance matrix.
se	estimated standard errors of (bhat, sighat).
grad	gradient vector for the log-likelihood at the last iteration.
iter	number of iterations performed.
ier	error code; equals 0 if estimation was successful; equals 1 otherwise.

Author(s)

Paul W. Wilson

References

Olsen, R. (1978), A note on the uniqueness of the maximum likelihood estimator in the tobit model, *Econometrica* 46, 1211–1215.

Simar, L. and P.W. Wilson (2007), "Estimation and Inference in Two-Stage, Semi-Parametric Models of Production Processes," *Journal of Econometrics* 137, 31–64.

Examples

```
set.seed(900001)
n=100
k=5
km1=4
z=matrix(c(rep(1,n),rnorm(km1*n)),nrow=n,ncol=k)
z[,5]=100 + 10*z[,5]
b=matrix(c(rep(1,k)),nrow=k)
zb=z %*% b
t1=1-zb
eps=rnorm.trunc(n,t.left=t1)
y=zb+eps
tmp=treg(Y=y,X=z,TPOINT=1)
```

---

wood *Wood (1973) refinery data*

Description

The wood data frame contains 82 observations from a process variable study of an oil refinery unit; see Wood (1973) for details.

Usage

```
data(wood)
```

Format

This data frame contains the following variables:

dmu observation number

x1 input no. 1

x2 input no. 2

x3 input no. 3

x4 input no. 4

y output; octane number of the product produced.

The first three inputs represent feed compositions, while the fourth input is the log of a combination of process conditions.

Source

Wood, F.S. (1973), The use of individual effects and residuals in fitting equaitons to data, *Technometrics* 15, 677–694.

Examples

```
data(wood)
```

Index