Package ‘SharpeR’

October 7, 2018

Maintainer Steven E. Pav <shabbychef@gmail.com>

Version 1.2.0

Date 2018-10-07

License LGPL-3

Title Statistical Significance of the Sharpe Ratio

BugReports https://github.com/shabbychef/SharpeR/issues

Description A collection of tools for analyzing significance of assets, 
funds, and trading strategies, based on the Sharpe ratio and overfit 
of the same. Provides density, distribution, quantile and random generation 
of the Sharpe ratio distribution based on normal returns, as well 
as the optimal Sharpe ratio over multiple assets. Computes confidence intervals 
on the Sharpe and provides a test of equality of Sharpe ratios based on 
the Delta method.

Depends R (>= 3.0.0)

Imports matrixcalc, methods, sadists (>= 0.2.0)

Suggests xtable, xts, timeSeries, quantmod, MASS, TTR, testthat, 
sandwich, knitr

URL https://github.com/shabbychef/SharpeR

VignetteBuilder knitr

Collate 'SharpeR.r' 'utils.r' 'distributions.r' 'sr.r' 'estimation.r' 
'sr_bias.r' 'tests.r' 'unified.r'

RoxygenNote 6.0.1

NeedsCompilation no

Author Steven E. Pav [aut, cre] (<https://orcid.org/0000-0002-4197-6195>)

Repository CRAN

Date/Publication 2018-10-07 21:00:13 UTC
Description

Inference on Sharpe ratio and Markowitz portfolio.
Sharpe Ratio

Suppose \( x_i \) are \( n \) independent draws of a normal random variable with mean \( \mu \) and variance \( \sigma^2 \). Let \( \bar{x} \) be the sample mean, and \( s \) be the sample standard deviation (using Bessel’s correction). Let \( c_0 \) be the ‘risk free’ or ‘disastrous rate’ of return. Then

\[
z = \frac{\bar{x} - c_0}{s}
\]

is the (sample) Sharpe ratio.

The units of \( z \) are \( \text{time}^{-1/2} \). Typically the Sharpe ratio is \emph{annualized} by multiplying by \( \sqrt{d} \), where \( d \) is the number of observations per year (or whatever the target annualization epoch.) It is \emph{not} common practice to include units when quoting Sharpe ratio, though doing so could avoid confusion.

The Sharpe ratio follows a rescaled non-central t distribution. That is, \( z/K \) follows a non-central t-distribution with \( m \) degrees of freedom and non-centrality parameter \( \zeta/K \), for some \( K, m \) and \( \zeta \).

We can generalize Sharpe’s model to APT, wherein we write

\[
x_i = \alpha + \sum_j \beta_j F_{j,i} + \epsilon_i,
\]

where the \( F_{j,i} \) are observed ‘factor returns’, and the variance of the noise term is \( \sigma^2 \). Via linear regression, one can compute estimates \( \hat{\alpha} \) and \( \hat{\sigma} \), and then let the ‘Sharpe ratio’ be

\[
z = \frac{\hat{\alpha} - c_0}{\hat{\sigma}}.
\]

As above, this Sharpe ratio follows a rescaled t-distribution under normality, \emph{etc}.

The parameters are encoded as follows:

- \( \text{df} \) stands for the degrees of freedom, typically \( n - 1 \), but \( n - J - 1 \) in general.
- \( \zeta \) is denoted by \text{zeta}.
- \( d \) is denoted by \text{ope} (‘Observations Per Year’)
- For the APT form of Sharpe, \( K \) stands for the rescaling parameter.

Optimal Sharpe Ratio

Suppose \( x_i \) are \( n \) independent draws of a \( q \)-variate normal random variable with mean \( \mu \) and covariance matrix \( \Sigma \). Let \( \bar{x} \) be the (vector) sample mean, and \( S \) be the sample covariance matrix (using Bessel’s correction). Let

\[
Z(w) = \frac{w^\top \bar{x} - c_0}{\sqrt{w^\top Sw}}
\]

be the (sample) Sharpe ratio of the portfolio \( w \), subject to risk free rate \( c_0 \).

Let \( w_* \) be the solution to the portfolio optimization problem:

\[
\max_{w: 0 \leq w^\top w \leq R^2} Z(w),
\]

with maximum value \( z_* = Z(w_*) \). Then

\[
w_* = R \frac{S^{-1} \bar{x}}{\sqrt{\bar{x}^\top S^{-1} \bar{x}}}
\]
\[ z_* = \sqrt{\bar{x}^\top S^{-1} \bar{x} - \frac{c_0}{R}} \]

The variable \( z_* \) follows an *Optimal Sharpe ratio* distribution. For convenience, we may assume that the sample statistic has been annualized in the same manner as the Sharpe ratio, that is by multiplying by \( d \), the number of observations per epoch.

The Optimal Sharpe Ratio distribution is parametrized by the number of assets, \( q \), the number of independent observations, \( n \), the noncentrality parameter, \( \zeta_* = \sqrt{\mu^\top \Sigma^{-1} \mu} \), the 'drag' term, \( c_0/R \), and the annualization factor, \( d \). The drag term makes this a location family of distributions, and by default we assume it is zero.

The parameters are encoded as follows:
- \( q \) is denoted by \( \text{df1} \).
- \( n \) is denoted by \( \text{dfR} \).
- \( \zeta_* \) is denoted by \( \text{zetaNs} \).
- \( d \) is denoted by \( \text{ope} \).
- \( c_0/R \) is denoted by \( \text{drag} \).

**Spanning and Hedging**

As above, let

\[ Z(w) = \frac{w^\top \bar{x} - c_0}{\sqrt{w^\top S w}} \]

be the (sample) Sharpe ratio of the portfolio \( w \), subject to risk free rate \( c_0 \).

Let \( G \) be a \( g \times q \) matrix of 'hedge constraints'. Let \( w_* \) be the solution to the portfolio optimization problem:

\[ \max_{w:0<w^\top S w \leq R^2, GS w=0} Z(w), \]

with maximum value \( z_* = Z(w_*) \). Then \( z_*^2 \) can be expressed as the difference of two squared optimal Sharpe ratio random variables. A monotonic transform takes this difference to the LRT statistic for portfolio spanning, first described by Rao, and refined by Giri.

**Legal Mumbo Jumbo**

SharpeR is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

**Note**

The following are still in the works:

1. Corrections for standard error based on skew, kurtosis and autocorrelation.
2. Tests on Sharpe under positivity constraint. (c.f. Silvapulle)
3. Portfolio spanning tests.
4. Tests on portfolio weights.

This package is maintained as a hobby.

Author(s)
Steven E. Pav <shabbychef@gmail.com>

References

http://ideas.repec.org/a/ucp/jnlbus/v39y1965p119.html


as.del_sropt

Compute the Sharpe ratio of a hedged Markowitz portfolio.

Description

Computes the Sharpe ratio of the hedged Markowitz portfolio of some observed returns.

Usage

as.del_sropt(X, G, drag = 0, ope = 1, epoch = "yr")

## Default S3 method:
as.del_sropt(X, G, drag = 0, ope = 1, epoch = "yr")

## S3 method for class 'xts'
as.del_sropt(X, G, drag = 0, ope = 1, epoch = "yr")

Arguments

X  
matrix of returns, or xts object.

G  
an $g \times q$ matrix of hedge constraints. A garden variety application would have $G$ be one row of the identity matrix, with a one in the column of the instrument to be 'hedged out'.

drag  
the 'drag' term, $c_0/R$. defaults to 0. It is assumed that drag has been annualized, i.e. has been multiplied by $\sqrt{\text{ope}}$. This is in contrast to the $c_0$ term given to sr.

ope  
the number of observations per 'epoch'. For convenience of interpretation, the Sharpe ratio is typically quoted in 'annualized' units for some epoch, that is, 'per square root epoch', though returns are observed at a frequency of ope per epoch. The default value is 1, meaning the code will not attempt to guess what the observation frequency is, and no annualization adjustments will be made.

epoch  
the string representation of the 'epoch', defaulting to 'yr'.

Details

Suppose $x_i$ are $n$ independent draws of a $q$-variate normal random variable with mean $\mu$ and covariance matrix $\Sigma$. Let $G$ be a $g \times q$ matrix of rank $g$. Let $\bar{x}$ be the (vector) sample mean, and $S$ be the sample covariance matrix (using Bessel’s correction). Let

$$\zeta(w) = \frac{w^T \bar{x} - c_0}{\sqrt{w^T S w}}$$

be the (sample) Sharpe ratio of the portfolio $w$, subject to risk free rate $c_0$.

Let $w_*$ be the solution to the portfolio optimization problem:

$$\max_{w:0 \leq w^T S w \leq R^2, GS w = 0} \zeta(w),$$
with maximum value \( z^* = \zeta (w^*) \).

Note that if ope and epoch are not given, the converter from xts attempts to infer the observations per epoch, assuming yearly epoch.

**Value**

An object of class del_sropt.

**Author(s)**

Steven E. Pav <shabbychef@gmail.com>

**See Also**

del_sropt, sropt, sr

Other del_sropt: del_sropt, is_del_sropt

**Examples**

```r
nfac <- 5
cy <- 10
ope <- 253
# simulations with no covariance structure.
# under the null:
set.seed(as.integer(charToRaw("be deterministic")))
Returns <- matrix(rnorm(ope*nyr*nfac, mean=0, sd=0.0125), ncol=nfac)
# hedge out the first one:
G <- matrix(diag(nfac)[1,], nrow=1)
asro <- as_del_sropt(Returns, G, drag=0, ope=ope)
print(asro)
G <- diag(nfac)[c(1:3),]
asro <- as_del_sropt(Returns, G, drag=0, ope=ope)
# compare to sropt on the remaining assets
# they should be close, but not exact.
asro.alt <- as.sropt(Returns[,4:nfac], drag=0, ope=ope)

## Not run:
# using real data.
if (require(quantmod)) {
  get.ret <- function(sym,...) {
    OHLCV <- getSymbols(sym,auto.assign=FALSE,...)
    lrets <- diff(log(OHLCV[,paste(c(sym,"Adjusted"),collapse=",",sep="")]))
    # chomp first NA!
    lrets[-1,]
  }
  get.rets <- function(syms,...) {
    some.rets <- do.call("cbind", lapply(syms, get.ret,...))
  }
  some.rets <- get.rets(c("IBM","AAPL","A","C","SPY","XOM"))
  # hedge out SPY
  G <- diag(dim(some.rets)[2])[5,]
asro <- as_del_sropt(some.rets, G)
}
as.sr  Compute the Sharpe ratio.

Description

Computes the Sharpe ratio of some observed returns.

Usage

as.sr(x, c0 = 0, ope = 1, na.rm = FALSE, epoch = "yr", higher_order = FALSE)

## Default S3 method:
as.sr(x, c0 = 0, ope = 1, na.rm = FALSE, epoch = "yr", higher_order = FALSE)

## S3 method for class 'matrix'
as.sr(x, c0 = 0, ope = 1, na.rm = FALSE, epoch = "yr", higher_order = FALSE)

## S3 method for class 'data.frame'
as.sr(x, c0 = 0, ope = 1, na.rm = FALSE, epoch = "yr", higher_order = FALSE)

## S3 method for class 'lm'
as.sr(x, c0 = 0, ope = 1, na.rm = FALSE, epoch = "yr", higher_order = FALSE)

## S3 method for class 'xts'
as.sr(x, c0 = 0, ope = 1, na.rm = FALSE, epoch = "yr", higher_order = FALSE)

## S3 method for class 'timeSeries'
as.sr(x, c0 = 0, ope = 1, na.rm = FALSE, epoch = "yr", higher_order = FALSE)

Arguments

x  vector of returns, or object of class data.frame, xts, or lm.
c0  the 'risk-free' or 'disastrous' rate of return. this is assumed to be given in the same units as x, not in 'annualized' terms.
the number of observations per 'epoch'. For convenience of interpretation, the Sharpe ratio is typically quoted in 'annualized' units for some epoch, that is, 'per square root epoch', though returns are observed at a frequency of ope per epoch. The default value is 1, meaning the code will not attempt to guess what the observation frequency is, and no annualization adjustments will be made.

na.rm
logical. Should missing values be removed?

epoch
the string representation of the 'epoch', defaulting to 'yr'.

higher_order
a Boolean. If true, we compute cumulants of the returns to leverage higher order accuracy formulae when possible.

Details
Suppose \( x_i \) are \( n \) independent returns of some asset. Let \( \bar{x} \) be the sample mean, and \( s \) be the sample standard deviation (using Bessel’s correction). Let \( c_0 \) be the 'risk free rate'. Then

\[
    z = \frac{\bar{x} - c_0}{s}
\]

is the (sample) Sharpe ratio.

The units of \( z \) are time\(^{-1/2}\). Typically the Sharpe ratio is *annualized* by multiplying by \( \sqrt{\text{ope}} \), where ope is the number of observations per year (or whatever the target annualization epoch.)

Note that if ope is not given, the converter from xts attempts to infer the observations per year, without regard to the name of the epoch given.

Value
a list containing the following components:

- \textbf{sr} the annualized Sharpe ratio.
- \textbf{df} the t-stat degrees of freedom.
- \textbf{c0} the risk free term.
- \textbf{ope} the annualization factor.
- \textbf{rescal} the rescaling factor.
- \textbf{epoch} the string epoch.

cast to class \textbf{sr}.

Author(s)
Steven E. Pav <shabbychef@gmail.com>

References

See Also

reannualize
sr-distribution functions, dsr, psr, qsr, rsr

Other sr: confint.sr, dsl, is.sr, plambdap, power.sr_test, predict.sr, reannualize, se, sr.equality_test, sr.test, sr_unpaired_test, sr_vcov, sr.summary.sr

Examples

# Sharpe's 'model': just given a bunch of returns.

asr <- as.sr(rnorm(253*3), ope=253)
# or a matrix, with a name

my.returns <- matrix(rnorm(253*3), ncol=1)
colnames(my.returns) <- c("my strategy")
asr <- as.sr(my.returns)
# given an xts object:

## Not run:
if (require(quantmod)) {
  IBM <- getSymbols("IBM", auto.assign=FALSE)
  lrets <- diff(log(IBM[, "IBM.Adjusted"]))
  asr <- as.sr(lrets, na.rm=TRUE)
}

## End(Not run)
# on a linear model, find the 'Sharpe' of the residual term

nfac <- 5
nyr <- 10
ope <- 253
set.seed(as.integer(charToRaw("deterministic")))
Factors <- matrix(rnorm(ope*nyr*nfac, mean=0, sd=0.0125), ncol=nfac)
Betals <- exp(0.1 * rnorm(dim(Factors)[2]))
Returns <- (Factors %*% Betals) + rnorm(dim(Factors)[1], mean=0.0005, sd=0.012)
APT_mod <- lm(Returns ~ Factors)
asr <- as.sr(APT_mod, ope=ope)
# try again, but make the Returns independent of the Factors.
Returns <- rnorm(dim(Factors)[1], mean=0.0005, sd=0.012)
APT_mod <- lm(Returns ~ Factors)
asr <- as.sr(APT_mod, ope=ope)

# compute the Sharpe of a bunch of strategies:

my.returns <- matrix(rnorm(253*3*4), ncol=4)
asr <- as.sr(my.returns)  # without sensible colnames?
colnames(my.returns) <- c("strat a", "strat b", "strat c", "strat d")
asr <- as.sr(my.returns)
Description

Computes the Sharpe ratio of the Markowitz portfolio of some observed returns.

Usage

```r
as.sropt(X, drag = 0, ope = 1, epoch = "yr")
```

## Default S3 method:

```r
as.sropt(X, drag = 0, ope = 1, epoch = "yr")
```

## S3 method for class 'xts'

```r
as.sropt(X, drag = 0, ope = 1, epoch = "yr")
```

Arguments

- `X`: matrix of returns, or `xts` object.
- `drag`: the 'drag' term, $c_0/R$. defaults to 0. It is assumed that `drag` has been annualized, i.e. has been multiplied by $\sqrt{\text{ope}}$. This is in contrast to the $c_0$ term given to `sr`.
- `ope`: the number of observations per 'epoch'. For convenience of interpretation, the Sharpe ratio is typically quoted in 'annualized' units for some epoch, that is, 'per square root epoch', though returns are observed at a frequency of `ope` per epoch. The default value is 1, meaning the code will not attempt to guess what the observation frequency is, and no annualization adjustments will be made.
- `epoch`: the string representation of the 'epoch', defaulting to 'yr'.

Details

Suppose $x_i$ are $n$ independent draws of a $q$-variate normal random variable with mean $\mu$ and covariance matrix $\Sigma$. Let $\bar{x}$ be the (vector) sample mean, and $S$ be the sample covariance matrix (using Bessel’s correction). Let

$$
\zeta(w) = \frac{w^\top \bar{x} - c_0}{\sqrt{w^\top Sw}}
$$

be the (sample) Sharpe ratio of the portfolio $w$, subject to risk free rate $c_0$.

Let $w_\star$ be the solution to the portfolio optimization problem:

$$
\max_{w:0<w^\top Sw\leq R^2} \zeta(w),
$$

with maximum value $z_\star = \zeta(w_\star)$. Then

$$
w_\star = R \frac{S^{-1} \bar{x}}{\sqrt{\bar{x}^\top S^{-1} \bar{x}}}
$$

and

$$
z_\star = \sqrt{\bar{x}^\top S^{-1} \bar{x}} - \frac{c_0}{R}
$$

The units of $z_\star$ are time$^{-1/2}$. Typically the Sharpe ratio is *annualized* by multiplying by $\sqrt{\text{ope}}$, where `ope` is the number of observations per year (or whatever the target annualization epoch.)

Note that if `ope` and `epoch` are not given, the converter from `xts` attempts to infer the observations per epoch, assuming yearly epoch.
Value

An object of class \texttt{sropt}.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

See Also

\texttt{sropt}, \texttt{sropt_distribution()} functions, \texttt{dsropt}, \texttt{psropt}, \texttt{qsropt}, \texttt{rsropt}

Other \texttt{sropt}: \texttt{confint.sropt}, \texttt{dsropt}, \texttt{is.sropt}, \texttt{pcosropt}, \texttt{powersropt}, \texttt{powersropt_test}, \texttt{reannualize}, \texttt{sropt_test.sropt}

Examples

\begin{verbatim}
nfac <- 5
nyr <- 10
ope <- 253
# simulations with no covariance structure.
# under the null:
set.seed(as.integer(charToRaw("be deterministic")))
Returns <- matrix(rnorm(ope*nyr*nfac,mean=0,sd=0.0125),ncol=nfac)
asro <- as.sropt(Returns,drag=0,ope=ope)
# under the alternative:
Returns <- matrix(rnorm(ope*nyr*nfac,mean=0.0005,sd=0.0125),ncol=nfac)
asro <- as.sropt(Returns,drag=0,ope=ope)
# generating correlated multivariate normal data in a more sane way
if (require(MASS)) {
  nstok <- 10
  nfac <- 3
  nyr <- 10
  ope <- 253
  X.like <- 0.01 * matrix(rnorm(500*nfac),ncol=nfac) %*% matrix(runif(nfac*nstok),ncol=nstok)
  Sigma <- cov(X.like) + diag(0.003,nstok)
  # under the null:
  Returns <- mvrnorm(ceiling(ope*nyr),mu=matrix(0,ncol=nstok),Sigma=Sigma)
asro <- as.sropt(Returns,ope=ope)
  # under the alternative
  Returns <- mvrnorm(ceiling(ope*nyr),mu=matrix(0.001,ncol=nstok),Sigma=Sigma)
asro <- as.sropt(Returns,ope=ope)
}
## Not run:
# using real data.
if (require(quantmod)) {
  get.ret <- function(sym,...) {
    OHLCV <- getSymbols(sym,auto.assign=FALSE,...)
lrets <- diff(log(OHLCV[,paste(c(sym,"Adjusted"),collapse="","sep=""))]
    # chomp first NA!
lrets[1,]
  }
\end{verbatim}
get.rets <- function(symss,...) {
  some.rets <- do.call("cbind",lapply(symss,get.rets,...))
}
some.rets <- get.rets(c("IBM","AAPL","A","C","SPY","XOM"))
asro <- as.sropt(some.rets)

## End(Not run)

---

confint.sr

Confidence Interval on (optimal) Signal-Noise Ratio

Description

Computes approximate confidence intervals on the (optimal) Signal-Noise ratio given the (optimal) Sharpe ratio. Works on objects of class `sr` and `sropt`.

Usage

## S3 method for class 'sr'
confint(object, parm, level = 0.95, level.lo = (1 - level)/2, level.hi = 1 - level.lo, type = c("exact", "t", "Z", "Mertens", "Bao"), ...)

## S3 method for class 'sropt'
confint(object, parm, level = 0.95, level.lo = (1 - level)/2, level.hi = 1 - level.lo, ...)

## S3 method for class 'del_sropt'
confint(object, parm, level = 0.95, level.lo = (1 - level)/2, level.hi = 1 - level.lo, ...)

Arguments

- **object**: an observed Sharpe ratio statistic, of class `sr` or `sropt`.
- **parm**: ignored here, but required for the general method.
- **level**: the confidence level required.
- **level.lo**: the lower confidence level required.
- **level.hi**: the upper confidence level required.
- **type**: which method to apply.
- **...**: further arguments to be passed to or from methods.
Details

Constructs confidence intervals on the Signal-Noise ratio given observed Sharpe ratio statistic. The available methods are:

- **exact**: The default, which is only exact when returns are normal, based on inverting the non-central t distribution.
- **t**: Uses the Johnson Welch approximation to the standard error, centered around the sample value.
- **Z**: Uses the Johnson Welch approximation to the standard error, performing a simple correction for the bias of the Sharpe ratio based on Miller and Gehr formula.
- **Mertens**: Uses the Mertens higher order approximation to the standard error, centered around the sample value.
- **Bao**: Uses the Bao higher order approximation to the standard error, performing a higher order correction for the bias of the Sharpe ratio.

Suppose \( x_i \) are \( n \) independent draws of a \( q \)-variate normal random variable with mean \( \mu \) and covariance matrix \( \Sigma \). Let \( \bar{x} \) be the (vector) sample mean, and \( S \) be the sample covariance matrix (using Bessel’s correction). Let

\[
z_* = \sqrt{\bar{x}^\top S^{-1} \bar{x}}
\]

Given observations of \( z_* \), compute confidence intervals on the population analogue, defined as

\[
\zeta_* = \sqrt{\mu^\top \Sigma^{-1} \mu}
\]

Value

A matrix (or vector) with columns giving lower and upper confidence limits for the parameter. These will be labelled as level.lo and level.hi in %, e.g. "2.5 %"

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References


See Also

- confint, se
- Other sr: as.sr, dsr, is.sr, plambdap, power.sr_test, predint, print.sr, reannualize, se, sr_equality_test, sr_test, sr_unpaired_test, sr_vcov, sr, summary.sr
- Other sropt: as.sropt, dsropt, is.sropt, pco_sropt, power.sropt_test, reannualize, sropt_test, sropt
Examples

```r
# using "sr" class:
ope <- 253
df <- ope * 6
xv <- rnorm(df, 1 / sqrt(ope))
mysr <- as.sr(xv, ope=ope)
confint(mysr, level=0.90)

# using "lm" class
yv <- xv + rnorm(length(xv))
amod <- lm(yv - xv)
mysr <- as.sr(amod, ope=ope)
confint(mysr, level.lo=0.05, level.hi=1.0)

# rolling your own.
ope <- 253
df <- ope * 6
zeta <- 1.0
rvs <- rsr(128, df, zeta, ope)
roll.own <- sr(s=rvs, df=df, c0=0, ope=ope)
aci <- confint(roll.own, level=0.95)
coverage <- 1 - mean((zeta < aci[,1]) | (aci[,2] < zeta))

# using "sropt" class
ope <- 253
df1 <- 4
df2 <- ope * 3
rvs <- as.matrix(rnorm(df1*df2), ncol=df1)
sro <- as.sropt(rvs, ope=ope)
aci <- confint(sro)

# on sropt, rolling your own.
rvs <- rsropt(128, df1, df2, zeta.s, ope)
roll.own <- sropt(z.s=rvs, df1, df2, drag=0, ope=ope)
aci <- confint(roll.own, level=0.95)
coverage <- 1 - mean((zeta.s < aci[,1]) | (aci[,2] < zeta.s))

# using "del_sropt" class
nfac <- 5
nyr <- 10
ope <- 253
set.seed(as.integer(charToRaw("be deterministic")))

Returns <- matrix(rnorm(ope*nyr*nfac, mean=0, sd=0.0125), ncol=nfac)

# hedge out the first one:
G <- matrix(diag(nfac)[1,], nrow=1)
asro <- as.del_sropt(Returns, G, drag=0, ope=ope)
aci <- confint(asro, level=0.95)

# under the alternative
Returns <- matrix(rnorm(ope*nyr*nfac, mean=0.001, sd=0.0125), ncol=nfac)
asro <- as.del_sropt(Returns, G, drag=0, ope=ope)
aci <- confint(asro, level=0.95)
```
Create an 'del_sropt' object.

Spawns an object of class del_sropt.

Usage

del_sropt(z.s, z.sub, df1, df2, df1.sub, drag = 0, ope = 1, epoch = "yr")

Arguments

- **z.s**: an optimum Sharpe ratio statistic, on some set of assets.
- **z.sub**: an optimum Sharpe ratio statistic, on a linear subspace of the assets. If larger than z.s an error is thrown.
- **df1**: the number of assets in the portfolio.
- **df2**: the number of observations.
- **df1.sub**: the rank of the linear subspace of the hedge constraint. by restricting attention to the subspace.
- **drag**: the 'drag' term, $c_0/R$. defaults to 0. It is assumed that drag has been annualized, i.e. has been multiplied by $\sqrt{\text{ope}}$. This is in contrast to the c0 term given to sr.
- **ope**: the number of observations per 'epoch'. For convenience of interpretation, The Sharpe ratio is typically quoted in 'annualized' units for some epoch, that is, 'per square root epoch', though returns are observed at a frequency of ope per epoch. The default value is 1, meaning the code will not attempt to guess what the observation frequency is, and no annualization adjustments will be made.
- **epoch**: the string representation of the 'epoch', defaulting to 'yr'.

Details

The del_sropt class contains information about the difference between two rescaled $T^2$-statistics, useful for spanning tests, and inference on hedged portfolios. The following are list attributes of the object:

- **sropt**: The (optimal) Sharpe ratio statistic of the 'full' set of assets.
- **sropt_sub**: The (optimal) Sharpe ratio statistic on some subset, or linear subspace, of the assets.
- **df1**: The number of assets.
- **df2**: The number of observations.
- **df1.sub**: The number of degrees of freedom in the hedge constraint.
- **drag**: The drag term, which is the 'risk free rate' divided by the maximum risk.
- **ope**: The 'observations per epoch'.
- **epoch**: The string name of the 'epoch'.

For the most part, this constructor should not be called directly, rather as_del_sropt should be called instead to compute the needed statistics.
**Value**

A list cast to class `del_sropt`, with attributes

- `sropt` the optimal Sharpe statistic.
- `sropt.sub` the optimal Sharpe statistic on the subspace.
- `df1` the number of assets.
- `df2` the number of observed vectors.
- `df1.sub` the input `df1` sub term.
- `drag` the input `drag` term.
- `ope` the input `ope` term.
- `T2` the Hotelling $T^2$ statistic.
- `T2.sub` the Hotelling $T^2$ statistic on the subspace.

**Note**

**WARNING:** This function is not well tested, may contain errors, may change in the next package update. Take caution.

2FIX: allow rownames?

**Author(s)**

Steven E. Pav <shabbychef@gmail.com>

**See Also**

- `reannualize`
- `as.del_sropt`

Other `del_sropt`: `as.del_sropt`, `is.del_sropt`

**Examples**

```r
# roll your own.
ope <- 253

set.seed(as.integer(charToRaw("be deterministic")))
n.stock <- 10
X <- matrix(rnorm(1000*n.stock),nrow=1000)
Sigma <- cov(X)
mu <- colMeans(X)
w <- solve(Sigma,mu)
z <- t(mu) %*% w
n.sub <- 6
w.sub <- solve(Sigma[1:n.sub,1:n.sub],mu[1:n.sub])
z.sub <- t(mu[1:n.sub]) %*% w.sub
df1.sub <- n.stock - n.sub

roll.own <- del_sropt(z.s=z,z.sub=z.sub,df1=10,df2=1000,
```

```
### Description

Density, distribution function, quantile function and random generation for the Sharpe ratio distribution with \( df \) degrees of freedom (and optional signal-noise-ratio \( zeta \)).

### Usage

\[
\begin{align*}
\text{dsr}(x, df, zeta, ope, ...) \\
\text{psr}(q, df, zeta, ope, ...) \\
\text{qsr}(p, df, zeta, ope, ...) \\
\text{rsr}(n, df, zeta, ope)
\end{align*}
\]

### Arguments

- **x, q** vector of quantiles.
- **df** the number of observations the statistic is based on. This is one more than the number of degrees of freedom in the corresponding t-statistic, although the effect will be small when \( df \) is large.
- **zeta** the 'signal-to-noise' parameter, \( \zeta \) defined as the population mean divided by the population standard deviation, 'annualized'.
- **ope** the number of observations per 'epoch'. For convenience of interpretation, The Sharpe ratio is typically quoted in 'annualized' units for some epoch, that is, 'per square root epoch', though returns are observed at a frequency of \( ope \) per epoch. The default value is 1, meaning the code will not attempt to guess what the observation frequency is, and no annualization adjustments will be made.
- **...** arguments passed on to the respective t-distribution functions, namely `lower.tail` with default `TRUE`, `log` with default `FALSE`, and `log.p` with default `FALSE`.
- **p** vector of probabilities.
- **n** number of observations.
Details

Suppose \( x_i \) are \( n \) independent draws of a normal random variable with mean \( \mu \) and variance \( \sigma^2 \). Let \( \bar{x} \) be the sample mean, and \( s \) be the sample standard deviation (using Bessel’s correction). Let \( c_0 \) be the 'risk free rate'. Then

\[
z = \frac{\bar{x} - c_0}{s}
\]

is the (sample) Sharpe ratio.

The units of \( z \) is time\(^{-1/2} \). Typically the Sharpe ratio is *annualized* by multiplying by \( \sqrt{d} \), where \( d \) is the number of observations per epoch (typically a year).

Letting \( z = \sqrt{\frac{\bar{x} - c_0}{s}} \), where the sample estimates are based on \( n \) observations, then \( z \) takes a (non-central) Sharpe ratio distribution parametrized by \( n \) 'degrees of freedom', non-centrality parameter \( \zeta = \frac{\mu - c_0}{\sigma} \), and annualization parameter \( d \).

The parameters are encoded as follows:

- \( n \) is denoted by df.
- \( \zeta \) is denoted by zeta.
- \( d \) is denoted by ope. ('Observations Per Year')

If the returns violate the assumptions of normality, independence, etc (as they always should in the real world), the sample Sharpe Ratio will not follow this distribution. It does provide, however, a reasonable approximation in many cases.

Value

dsr gives the density, psr gives the distribution function, qsr gives the quantile function, and rsr generates random deviates.

Invalid arguments will result in return value NaN with a warning.

Note

This is a thin wrapper on the t distribution. The functions dt, pt, qt can accept ncp from limited range \((|\delta| \leq 37.62)\). Some corrections may have to be made here for large zeta.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References


See Also

reannualize
t-distribution functions, dt, pt, qt, rt

Other sr: as.sr, confint.sr, is.sr, plambdap, power.sr_test, predint, print.sr, reannualize, se, sr_equality_test, sr_test, sr_unpaired_test, sr_vcov, sr, summary.sr
Examples

```r
# Examples
rvs <- rsr(128, 253*6, 0, 253)
dvs <- dsr(rvs, 253*6, 0, 253)
pvs.H0 <- psr(rvs, 253*6, 0, 253)
pvs.HA <- psr(rvs, 253*6, 1, 253)

## Not run:
plot(ecdf(pvs.H0))
plot(ecdf(pvs.HA))

## End(Not run)
```

---

**dsropt**

*The (non-central) maximal Sharpe ratio distribution.*

---

**Description**

Density, distribution function, quantile function and random generation for the maximal Sharpe ratio distribution with `df1` and `df2` degrees of freedom (and optional maximal signal-noise-ratio `zeta.s`).

**Usage**

```r
dsropt(x, df1, df2, zeta.s, ope, drag = 0, log = FALSE)
psropt(q, df1, df2, zeta.s, ope, drag = 0, ...)
qsropt(p, df1, df2, zeta.s, ope, drag = 0, ...)
rsropt(n, df1, df2, zeta.s, ope, drag = 0, ...)
```

**Arguments**

- `x`, `q` vector of quantiles.
- `df1` the number of assets in the portfolio.
- `df2` the number of observations.
- `zeta.s` the non-centrality parameter, defined as $\zeta = \sqrt{\mu^\top \Sigma^{-1} \mu}$, for population parameters. defaults to 0, *i.e.* a central maximal Sharpe ratio distribution.
- `ope` the number of observations per ‘epoch’. For convenience of interpretation, the Sharpe ratio is typically quoted in ‘annualized’ units for some epoch, that is, ‘per square root epoch’, though returns are observed at a frequency of `ope` per epoch. The default value is 1, meaning the code will not attempt to guess what the observation frequency is, and no annualization adjustments will be made.
- `drag` the ‘drag’ term, $c_0/R$. defaults to 0. It is assumed that `drag` has been annualized, *i.e.* is given in the same units as `x` and `q`.
- `log` logical; if TRUE, densities `f` are given as `log(f)`. 

---
Suppose \( x_i \) are \( n \) independent draws of a \( q \)-variate normal random variable with mean \( \mu \) and covariance matrix \( \Sigma \). Let \( \bar{x} \) be the (vector) sample mean, and \( S \) be the sample covariance matrix (using Bessel’s correction). Let

\[
Z(w) = \frac{w^\top \bar{x} - c_0}{\sqrt{w^\top Sw}}
\]

be the (sample) Sharpe ratio of the portfolio \( w \), subject to risk free rate \( c_0 \).

Let \( w^* \) be the solution to the portfolio optimization problem:

\[
\max_{w:0<w^\top Sw\leq R^2} Z(w),
\]

with maximum value \( z^* = Z(w^*) \). Then

\[
w^* = R \frac{S^{-1} \bar{x}}{\sqrt{\bar{x}^\top S^{-1} \bar{x}}}
\]

and

\[
z^* = \sqrt{\bar{x}^\top S^{-1} \bar{x}} - \frac{c_0}{R}
\]

The variable \( z^* \) follows an Optimal Sharpe ratio distribution. For convenience, we may assume that the sample statistic has been annualized in the same manner as the Sharpe ratio, that is by multiplying by \( d \), the number of observations per epoch.

The Optimal Sharpe Ratio distribution is parametrized by the number of assets, \( q \), the number of independent observations, \( n \), the noncentrality parameter, \( \zeta^* = \sqrt{\mu^\top \Sigma^{-1} \mu} \), the 'drag' term, \( c_0 / R \), and the annualization factor, \( d \). The drag term makes this a location family of distributions, and by default we assume it is zero.

The parameters are encoded as follows:

- \( q \) is denoted by df1.
- \( n \) is denoted by df2.
- \( \zeta^* \) is denoted by zeta.s.
- \( d \) is denoted by ope.
- \( c_0 / R \) is denoted by drag.

**Value**

dsropt gives the density, psropt gives the distribution function, qsropt gives the quantile function, and rsropt generates random deviates.

Invalid arguments will result in return value NaN with a warning.
Note

This is a thin wrapper on the Hotelling T-squared distribution, which is a wrapper on the F distribution.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References


See Also

reannualize

F-distribution functions, df, pf, qf, rf, Sharpe ratio distribution, dsr, psr, qsr, rsr.

Other sropt: as.sropt, confint.sr, is.sropt, pco_sropt, power.sropt_test, reannualize, sropt_test, sropt

Examples

```r
# generate some variates
ngen <- 128
ope <- 253
df1 <- 8
df2 <- ope * 10
drag <- 0
# sample
rvs <- rsropt(ngen, df1, df2, drag, ope)
hist(rvs)
# these should be uniform:
isp <- psropt(rvs, df1, df2, drag, ope)
plot(ecdf(isp))
```

---

**inference**

Inference on noncentrality parameter of F-like statistic

**Description**

Estimates the non-centrality parameter associated with an observed statistic following an optimal Sharpe Ratio distribution.
Usage

inference(z.s, type = c("KRS", "MLE", "unbiased"))

# S3 method for class 'sropt'
inference(z.s, type = c("KRS", "MLE", "unbiased"))

# S3 method for class 'del_sropt'
inference(z.s, type = c("KRS", "MLE", "unbiased"))

Arguments

z.s an object of type sropt, or del_sropt
type the estimator type. one of c("KRS", "MLE", "unbiased")

Details

Let $F$ be an observed statistic distributed as a non-central $F$ with $\nu_1$, $\nu_2$ degrees of freedom and non-centrality parameter $\delta^2$. Three methods are presented to estimate the non-centrality parameter from the statistic:

- an unbiased estimator, which, unfortunately, may be negative.
- the Maximum Likelihood Estimator, which may be zero, but not negative.
- the estimator of Kubokawa, Roberts, and Shaleh (KRS), which is a shrinkage estimator.

The sropt distribution is equivalent to an $F$ distribution up to a square root and some rescalings.
The non-centrality parameter of the sropt distribution is the square root of that of the Hotelling, i.e. has units 'per square root time'. As such, the 'unbiased' type can be problematic!

Value

an estimate of the non-centrality parameter, which is the maximal population Sharpe ratio.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References


See Also

F-distribution functions, df.
Other sropt Hotelling: sric
Examples

# generate some sropts
nfac <- 3
nyr <- 5
ope <- 253
# simulations with no covariance structure.
# under the null:
set.seed(as.integer(charToRaw("deterministic")))
Returns <- matrix(rnorm(nyr*nfac,mean=0, sd=0.0125), ncol=nfac)
asro <- as.sropt(Returns, drag=0, ope=ope)
est1 <- inference(asro, type='unbiased')
est2 <- inference(asro, type='KRS')
est3 <- inference(asro, type='MLE')

# under the alternative:
Returns <- matrix(rnorm(ope*nyr*nfac,mean=0.0005, sd=0.0125), ncol=nfac)
asro <- as.sropt(Returns, drag=0, ope=ope)
est1 <- inference(asro, type='unbiased')
est2 <- inference(asro, type='KRS')
est3 <- inference(asro, type='MLE')

# sample many under the alternative, look at the estimator.
df1 <- 3
df2 <- 512
ope <- 253
zeta.s <- 1.25
rvs <- rsropt(128, df1, df2, zeta.s, ope)
roll.own <- sropt(z.s=rvs, df1, df2, drag=0, ope=ope)
est1 <- inference(roll.own, type='unbiased')
est2 <- inference(roll.own, type='KRS')
est3 <- inference(roll.own, type='MLE')

# for del_sropt:
nfac <- 5
nyr <- 10
ope <- 253
set.seed(as.integer(charToRaw("fix seed")))
Returns <- matrix(rnorm(nyr*nfac,mean=0.0005, sd=0.0125), ncol=nfac)
# hedge out the first one:
G <- matrix(diag(nfac)[1,], nrow=1)
asro <- as.del_sropt(Returns, G, drag=0, ope=ope)
est1 <- inference(asro, type='unbiased')
est2 <- inference(asro, type='KRS')
est3 <- inference(asro, type='MLE')

---

is.del_sropt  Is this in the "del_sropt" class?
is.sr

Description
Checks if an object is in the class 'del_sropt'

Usage
is.del_sropt(x)

Arguments
x an object of some kind.

Details
To satisfy the minimum requirements of an S3 class.

Value
a boolean.

Author(s)
Steven E. Pav <shabbychef@gmail.com>

See Also
del_sropt
Other del_sropt: as.del_sropt, del_sropt

Examples
roll.own <- del_sropt(z.s=2,z.sub=1,df1=10,df2=1000,df1.sub=3,ope=1,epoch="yr")

is.sropt(roll.own)

---

is.sr 

Is this in the "sr" class?

---

Description
Checks if an object is in the class 'sr'

Usage
is.sr(x)

Arguments
x an object of some kind.
Details
To satisfy the minimum requirements of an S3 class.

Value
a boolean.

Author(s)
Steven E. Pav <shabbychef@gmail.com>

See Also
sr
Other sr: as.sr, confint.sr, dsr, plambdap, power.sr_test, predint.print.sr, reannualize, se, sr_equality_test, sr_test, sr_unpaired_test, sr_vcov, sr_summary.sr

Examples
```r
rvs <- as.sr(rnorm(253*8), ope=253)
is.sr(rvs)
```

is.sropt Is this in the "sropt" class?

Description
Checks if an object is in the class 'sropt'

Usage
```r
is.sropt(x)
```

Arguments

  x an object of some kind.

Details
To satisfy the minimum requirements of an S3 class.

Value
a boolean.

Author(s)
Steven E. Pav <shabbychef@gmail.com>
isn_vcov

See Also
sropt
Other sropt: as.sropt, confint.sr, dsropt, pco_sropt, power.sropt_test, reannualize, sropt_test, sropt

Examples

nfac <- 5
covr <- 10
ope <- 253
# simulations with no covariance structure.
# under the null:
set.seed(as.integer(charToRaw("be deterministic")))
Returns <- matrix(rnorm(ope*nyr*nfac,mean=0,sd=0.0125),ncol=nfac)
asro <- as.sropt(Returns,dr=0,ope=ope)
isn.sropt(asro)

---

ism_vcov

Compute variance covariance of Inverse 'Unified' Second Moment

Description

Computes the variance covariance matrix of the inverse unified second moment matrix.

Usage

ism_vcov(X,vcov.func=vcov,fit.intercept=TRUE)

Arguments

X an $n \times p$ matrix of observed returns.
vcov.func a function which takes an object of class lm, and computes a variance-covariance matrix. If equal to the string "normal", we assume multivariate normal returns.
fit.intercept a boolean controlling whether we add a column of ones to the data, or fit the raw uncentered second moment.

Details

Given $p$-vector $x$ with mean $\mu$ and covariance, $\Sigma$, let $y$ be $x$ with a one prepended. Then let $\Theta = E( y y^T )$, the uncentered second moment matrix. The inverse of $\Theta$ contains the (negative) Markowitz portfolio and the precision matrix.

Given $n$ contemporaneous observations of $p$-vectors, stacked as rows in the $n \times p$ matrix $X$, this function estimates the mean and the asymptotic variance-covariance matrix of $\Theta^{-1}$.

One may use the default method for computing covariance, via the vcov function, or via a 'fancy' estimator, like sandwich::vcovHAC, sandwich::vcovHC, etc.
Value

a list containing the following components:

- **mu**
  - a vector of the negative Markowitz portfolio, then the vech'd precision matrix of the sample data

- **Qhat**
  - the estimated variance covariance matrix.

- **n**
  - the number of rows in `X`.

- **p**
  - the number of assets.

Note

By flipping the sign of `X`, the inverse of Θ contains the positive Markowitz portfolio and the precision matrix on `X`. Performing this transform before passing the data to this function should be considered idiomatic.

This function will be deprecated in future releases of this package. Users should migrate at that time to a similar function in the MarkowitzR package.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References


See Also

*sm_vcov*, *sr_vcov*

Examples

```r
X <- matrix(rnorm(1000*3),ncol=3)
# putting in -X is idiomatic:
ism <- ism_vcov(-X)
isigmas.n <- ism_vcov(~X,vcov.func="normal")
isigmas.n <- ism_vcov(~X,fit.intercept=FALSE)
# compute the marginal Wald test statistics:
ism.mu <- ism$mu[1:ism$p]
isism.Sg <- ism$Qhat[1:ism$p,1:ism$p]
wald.stats <- ism.mu / sqrt(diag(ism.Sg))

# make it fat tailed:
X <- matrix(rt(1000*3,df=5),ncol=3)
ism <- ism_vcov(X)

# Not run:
if (require(sandwich)) {
  ism <- ism_vcov(X,vcov.func=vcovHC)
}
```
The 'confidence distribution' for maximal Sharpe ratio.

Distribution function and quantile function for the 'confidence distribution' of the maximal Sharpe ratio. This is just an inversion to perform inference on $\zeta^*$ given observed statistic $z^*$.

Usage

```r
pco_sropt(q,df1,df2,z,s,ope,lower.tail=TRUE,log.p=FALSE)
qco_sropt(p,df1,df2,z,s,ope,lower.tail=TRUE,log.p=FALSE,lb=0,ub=Inf)
```

Arguments

- `q` vector of quantiles.
- `df1` the number of assets in the portfolio.
- `df2` the number of observations.
- `z,s` an observed Sharpe ratio statistic, annualized.
- `ope` the number of observations per 'epoch'. For convenience of interpretation, The Sharpe ratio is typically quoted in 'annualized' units for some epoch, that is, 'per square root epoch', though returns are observed at a frequency of ope per epoch. The default value is 1, meaning the code will not attempt to guess what the observation frequency is, and no annualization adjustments will be made.
- `lower.tail` logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.
- `log.p` logical; if TRUE, probabilities $p$ are given as log($p$).
- `p` vector of probabilities.
- `lb` the lower bound for the output of `qco_sropt`.
- `ub` the upper bound for the output of `qco_sropt`. 
Details

Suppose \( z_* \) follows a Maximal Sharpe ratio distribution (see SharpeR-package) for known degrees of freedom, and unknown non-centrality parameter \( \zeta_* \). The 'confidence distribution' views \( \zeta_* \) as a random quantity once \( z_* \) is observed. As such, the CDF of the confidence distribution is the same as that of the Maximal Sharpe ratio (up to a flip of lower.tail); while the quantile function is used to compute confidence intervals on \( \zeta_* \) given \( z_* \).

Value

\texttt{pco\_sropt} gives the distribution function, and \texttt{qco\_sropt} gives the quantile function.

Invalid arguments will result in return value NaN with a warning.

Note

When lower.tail is true, \texttt{pco\_sropt} is monotonic increasing with respect to \( q \), and decreasing in \texttt{sropt}; these are reversed when lower.tail is false. Similarly, \texttt{qco\_sropt} is increasing in \texttt{sign(as.double(lower.tail) - 0.5)} * \texttt{p} and \texttt{- sign(as.double(lower.tail) - 0.5)} * \texttt{sropt}.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

See Also

\texttt{reannualize, dsropt, psropt, qsropt, rsropt}

Other \texttt{sropt}: \texttt{as\_sropt, confint\_sr, dsropt, is\_sropt, power\_sropt\_test, reannualize, sropt\_test, sropt}

Examples

```r
zeta.s <- 2.0
ope <- 253
ntest <- 50
df1 <- 4
df2 <- 6 * ope
rvs <- rsropt(ntest, df1=df1, df2=df2, zeta.s=zeta.s)
qvs <- seq(0.10, length.out=51)
pps <- pco_sropt(qvs, df1, df2, rvs[1], ope)
## Not run:
if (require(txtplot))
  txtplot(qvs, pps)
## End(Not run)
pps <- pco_sropt(qvs, df1, df2, rvs[1], ope, lower.tail=FALSE)
## Not run:
if (require(txtplot))
  txtplot(qvs, pps)
```
The lambda-prime distribution.

Description

Distribution function and quantile function for LeCoutre's lambda-prime distribution with df degrees of freedom and the observed t-statistic, tstat.

Usage

plambdap(q, df, tstat, lower.tail = TRUE, log.p = FALSE)
qlambdap(p, df, tstat, lower.tail = TRUE, log.p = FALSE)
rlambdap(n, df, tstat)

Arguments

q
vector of quantiles.
df
the degrees of freedom of the t-statistic.
tstat
the observed (non-central) t-statistic.
lower.tail
logical; if TRUE (default), probabilities are \( P[X \leq x] \), otherwise, \( P[X > x] \).
log.p
logical; if TRUE, probabilities p are given as \( \log(p) \).
p
vector of probabilities.
n
number of observations. If 'length(n) > 1', the length is taken to be the number required.
Details

Let $t$ be distributed as a non-central $t$ with $\nu$ degrees of freedom and non-centrality parameter $\delta$. We can view this as

$$t = \frac{Z + \delta}{\sqrt{V/\nu}}$$

where $Z$ is a standard normal, $\delta$ is the non-centrality parameter, $V$ is a chi-square RV with $\nu$ degrees of freedom, independent of $Z$. We can rewrite this as

$$\delta = t\sqrt{V/\nu} + Z.$$

Thus a 'lambda-prime' random variable with parameters $t$ and $\nu$ is one expressable as a sum

$$t\sqrt{V/\nu} + Z$$

for Chi-square $V$ with $\nu$ d.f., independent from standard normal $Z$

Value

dlambdap gives the density, plambdap gives the distribution function, qlambdap gives the quantile function, and rlambda generates random deviates.

Invalid arguments will result in return value NaN with a warning.

Note

plambdap should be an increasing function of the argument $q$, and decreasing in $tstat$. qlambdap should be increasing in $p$

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References


See Also

t-distribution functions, dt, pt, qt, rt

Other sr: as, sr, confint, sr, dsr, is, sr, power, sr.test, predint, print, sr, reannualize, se, sr.equality, sr.test, sr.unpaired.test, sr.vcov, sr, summary
Examples

```r
rvs <- rnorm(128)
pvs <- plambdap(rvs, 253*6, 0.5)
plot(ecdf(pvs))
pvs <- plambdap(rvs, 253*6, 1)
plot(ecdf(pvs))
pvs <- plambdap(rvs, 253*6, -0.5)
plot(ecdf(pvs))
# test vectorization:
qv <- qlambdap(0.1,128,2)
qv <- qlambdap(c(0.1),128,2)
qv <- qlambdap(c(0.2),128,2)
qv <- qlambdap(c(0.2),253,2)
qv <- qlambdap(c(0.1,0.2),128,2)
qv <- qlambdap(c(0.1,0.2),c(128,253),2)
qv <- qlambdap(c(0.1,0.2),c(128,253),c(2,4))
qv <- qlambdap(c(0.1,0.2),c(128,253),c(2,4,8,16))
# random generation
rv <- rlambdap(1000,252,2)
```

---

**power.sropt_test**  
*Power calculations for optimal Sharpe ratio tests*

**Description**

Compute power of test, or determine parameters to obtain target power.

**Usage**

```r
power.sropt_test(df1=NULL, df2=NULL, zeta.s=NULL, 
                 sig.level=0.05, power=NULL, ope=1)
```

**Arguments**

- **df1**
  - the number of assets in the portfolio.
- **df2**
  - the number of observations.
- **zeta.s**
  - the 'signal-to-noise' parameter, defined as ...
- **sig.level**
  - Significance level (Type I error probability).
- **power**
  - Power of test (1 minus Type II error probability).
- **ope**
  - the number of observations per 'epoch'. For convenience of interpretation, The Sharpe ratio is typically quoted in 'annualized' units for some epoch, that is, 'per square root epoch', though returns are observed at a frequency of ope per epoch. The default value is 1, meaning the code will not attempt to guess what the observation frequency is, and no annualization adjustments will be made.
Details

Suppose you perform a single-sample test for significance of the optimal Sharpe ratio based on the corresponding single-sample T^2-test. Given any four of: the effect size (the population optimal SNR, ζ), the number of assets, the number of observations, and the type I and type II rates, this function computes the fifth.

Exactly one of the parameters df1, df2, zeta, sig.level must be passed as NULL, and that parameter is determined from the others. Notice that sig.level has non-NULL default, so NULL must be explicitly passed if you want to compute it.

Value

Object of class power.htest, a list of the arguments (including the computed one) augmented with method, note and n.epoch elements, the latter is the number of epochs under the given annualization (ope), NA if none given.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

See Also

reannualize
power.t.test, sropt_test

Other sropt: as.sropt, confint.sr, dsropt, is.sropt, pco_sropt, reannualize, sropt_test, sropt

Examples

anex <- power.sropt_test(8,4*253,1,0.05,NULL,ope=253)
Arguments

- **n**: Number of observations
- **zeta**: The 'signal-to-noise' parameter, defined as the population mean divided by the population standard deviation, 'annualized'.
- **sig.level**: Significance level (Type I error probability).
- **power**: Power of test (1 minus Type II error probability).
- **alternative**: One- or two-sided test.
- **ope**: The number of observations per 'epoch'. For convenience of interpretation, the Sharpe ratio is typically quoted in 'annualized' units for some epoch, that is, 'per square root epoch', though returns are observed at a frequency of ope per epoch. The default value is 1, meaning the code will not attempt to guess what the observation frequency is, and no annualization adjustments will be made.

Details

Suppose you perform a single-sample test for significance of the Sharpe ratio based on the corresponding single-sample t-test. Given any three of: the effect size (the population SNR, $\zeta$), the number of observations, and the type I and type II rates, this function computes the fourth. This is a thin wrapper on `power.t.test`.

Exactly one of the parameters `n`, `zeta`, `power`, and `sig.level` must be passed as NULL, and that parameter is determined from the others. Notice that `sig.level` has non-NULL default, so NULL must be explicitly passed if you want to compute it.

Value

Object of class `power.htest`, a list of the arguments (including the computed one) augmented with `method`, `note` and `n` epoch elements, the latter is the number of epochs under the given annualization (ope), NA if none given.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References


predint

See Also

`reannualize`, `power.t.test`, `sr_test`

Other sr: `as.sr`, `confint.sr`, `dser`, `is.sr`, `plambdap`, `predint`, `print.sr`, `reannualize`, `se`, `sr.equality.test`, `sr_test`, `sr_unpaired.test`, `sr_vcov`, `sr`, `summary.sr`

Examples

```r
anex <- power.sr_test(253,1,0.05,NULL,ope=253)
anex <- power.sr_test(n=253,zeta=NULL,sig.level=0.05,power=0.5,ope=253)
anex <- power.sr_test(n=NULL,zeta=0.5,sig.level=0.05,power=0.5,ope=253)

# Lehr's Rule
zetas <- seq(0.1,2.5,length.out=51)
ssizes <- sapply(zetas,function(zed) {
  x <- power.sr_test(n=NULL,zeta=zed,sig.level=0.05,power=0.8,
    alternative="two.sided",ope=253)
  x$n / 253})

# should be around 8.
print(summary(ssizes * zetas * zetas))

# e = n z^2 mnemonic approximate rule for 0.05 type I, 50% power
ssizes <- sapply(zetas,function(zed) {
  x <- power.sr_test(n=NULL,zeta=zed,sig.level=0.05,power=0.5,ope=253)
  x$n / 253 })

print(summary(ssizes * zetas * zetas - exp(1)))
```

---

**predint**

**prediction interval for Sharpe ratio**

Description

Computes the prediction interval for Sharpe ratio.

Usage

```r
predint(x,oosdf,oosrescal=1/sqrt(oosdf+1),ope=NULL,level=0.95, level.lo=(1-level)/2,level.hi=1-level.lo)
```

Arguments

- `x`: a (non-empty) numeric vector of data values, or an object of class `sr`.
- `oosdf`: the future (or 'out of sample', thus 'oos') degrees of freedom. In the vanilla Sharpe case, this is the number of future observations minus one.
- `oosrescal`: the rescaling parameter for the future Sharpe ratio. The default value holds for the case of unattributed models ('vanilla Shape'), but can be set to some other value to deal with the magnitude of attribution factors in the future period.
predint

ope: the number of observations per 'epoch'. For convenience of interpretation, the Sharpe ratio is typically quoted in 'annualized' units for some epoch, that is, 'per square root epoch', though returns are observed at a frequency of ope per epoch. The default value is to take the same ope from the input x object, if it is unambiguous.

level: the confidence level required.

level.lo: the lower confidence level required.

level.hi: the upper confidence level level required.

Details

Given \( n_0 \) observations \( x_i \) from a normal random variable, with mean \( \mu \) and standard deviation \( \sigma \), computes an interval \([y_1, y_2]\) such that with a fixed probability, the sample Sharpe ratio over \( n \) future observations will fall in the given interval. The coverage is over repeated draws of both the past and future data, thus this computation takes into account error in both the estimate of Sharpe and the as yet unrealized returns.

Value

A matrix (or vector) with columns giving lower and upper confidence limits for the parameter. These will be labelled as level.lo and level.hi in %, e.g. "2.5 %"

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References


See Also

confint.sr.

Other sr: as.sr, confint.sr, dsr, is.sr, plambdap, power.sr_test, print.sr, reannualize, se, sr_equality_test, sr_test, sr_unpaired_test, sr_vcov, sr_summary.sr

Examples

```r
# should reject null
etc <- predint(rnorm(1000, mean=0.5, sd=0.1), oosdf=127, ope=1)
etc <- predint(matrix(rnorm(1000*5, mean=0.05), ncol=5), oosdf=63, ope=1)

# check coverage
mu <- 0.0005
sg <- 0.013
```
n1 <- 512
n2 <- 256
p <- 100
x1 <- matrix(rnorm(n1*p,mean=mu,sd=sig),ncol=p)
x2 <- matrix(rnorm(n2*p,mean=mu,sd=sig),ncol=p)
sr1 <- as.sr(x1)
sr2 <- as.sr(x2)
## Not run:
# takes too long to run ...
etc1 <- predint(sr1, oosdf=n2-1, level=0.95)
is.ok <- (etc1[,1] <= sr2$sr) & (sr2$sr <= etc1[,2])
covr <- mean(is.ok)
## End(Not run)

print.sr

Print values.

Description

Displays an object, returning it *invisibly*, (via `invisible(x)`.)

Usage

```r
## S3 method for class 'sr'
print(x, ...)
## S3 method for class 'sropt'
print(x, ...)
## S3 method for class 'del_sropt'
print(x, ...)
```

Arguments

- `x` an object of class `sr` or `sropt`.
- `...` further arguments to be passed to or from methods.

Value

the object, wrapped in `invisible`.

Author(s)

Steven E. Pav <shabbychef@gmail.com>
reannualize

References


See Also

Other sr: as.sr, confint.sr, dsr, is.sr, plambdap, power.sr_test, predint, reannualize, se.sr, sr_equality_test, sr_test, sr_unpaired_test, sr_vcov, sr, summary.sr

Examples

# compute a 'daily' Sharpe
mysr <- as.sr(rnorm(253*8), ope=1, epoch="day")
print(mysr)
# roll your own.
ope <- 253
zeta <- 1.0
n <- 6 * ope
rvs <- rsr(1,n,zeta,ope=ope)
roll.own <- sr(sr=rvs, df=n-1, ope=ope, rescal=sqrt(1/n))
print(roll.own)
# put a bunch in. naming becomes a problem.
rvs <- rsr(5,n,zeta, ope=ope)
roll.own <- sr(sr=rvs, df=n-1, ope=ope, rescal=sqrt(1/n))
print(roll.own)
# for sropt objects:
nfac <- 5
nyr <- 10
ope <- 253
# simulations with no covariance structure.
# under the null:
set.seed(as.integer(charToRaw("be deterministic")))
Returns <- matrix(rnorm(nopem*nyr*nfac, mean=0, sd=0.0125), ncol=nfac)
asro <- as.sropt(Returns, drag=0, ope=ope)
print(asro)

reannualize

Change the annualization of a Sharpe ratio.

Description

Changes the annualization factor of a Sharpe ratio statistic, or the rate at which observations are made.

Usage

reannualize(object, new.ope = NULL, new.epoch = NULL)

## S3 method for class 'sr'
reannualize(object, new.ope = NULL, new.epoch = NULL)

## S3 method for class 'sropt'
reannualize(object, new.ope = NULL, new.epoch = NULL)

Arguments

object an object of class sr or sropt.
new.ope the new observations per epoch. If none given, it is not updated.
new.epoch a string representation of the epoch. If none given, it is not updated.

Value

the input object with the annualization and/or epoch updated.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

See Also

sr
sropt

Other sr: as.sr, confint.sr, dsr, is.sr, plambdap, power.sr_test, predint, print.sr, se,
sr.equality_test, sr_test, sr_unpaired_test, sr_vcov, sr, summary.sr

Other sropt: as.sropt, confint.sr, dsropt, is.sropt, pco_sropt, power.sropt_test, sropt_test,
sropt

Examples

# compute a 'daily' Sharpe
mysr <- as.sr(rnorm(253*8), ope=1, epoch="day")
# turn into annual
mysr2 <- reannualize(mysr, new.ope=253, new.epoch="yr")

# for sropt
ope <- 253
zeta.s <- 1.0
df1 <- 10
df2 <- 6 * ope
rvs <- rsropt(1,df1,df2,zeta.s,ope,drag=0)
roll.own <- sropt(z.s=rvs,df1,df2,drag=0, ope=ope, epoch="yr")
# make 'monthly'
roll.monthly <- reannualize(roll.own, new.ope=21, new.epoch="mo.")
# make 'daily'
roll.daily <- reannualize(roll.own, new.ope=1, new.epoch="day")
Standard error computation

Description

Estimates the standard error of the Sharpe ratio statistic.

Usage

se(z, type)

## S3 method for class 'sr'
se(z, type = c("t", "Lo", "Mertens", "Bao"))

Arguments

- `z`: an observed Sharpe ratio statistic, of class `sr`.
- `type`: estimator type. one of "t", "Lo", "Mertens", "Bao"
- `...`: further arguments to be passed to or from methods.

Details

For an observed Sharpe ratio, estimate the standard error. The following methods are recognized:

- **t**: The default, based on Johnson & Welch, with a correction for small sample size. Also known as 'Lo'.
- **Mertens**: An approximation to the standard error taking into skewness and kurtosis of the returns distribution.
- **Bao**: An even higher accuracy approximation using higher order moments.

There should be very little difference between these except for very small sample sizes.

Value

an estimate of standard error.

Note

The units of the standard error are consistent with those of the input `sr` object.

Author(s)

Steven E. Pav <shabbychef@gmail.com>
References


See Also

sr-distribution functions, dsr, sr_variance.

Other sr: asr, confint.sr, dsr, is.sr, plambdap, power.sr_test, predint, print.sr, reannualize, sr_equality_test, sr_test, sr_unpaired_test, sr_vcov, sr.summary.sr

Examples

```r
asr <- as.sr(rnorm(128,0,2))
anse <- se(asr,type="t")
anse <- se(asr,type="Lo")
```

**SharpeR-NEWS**

*News for package 'SharpeR':*

**Description**

News for package 'SharpeR'

**Changes in SharpeR Version 1.2.0 (2016-10-07)**

- move github figures to location CRAN understands
- be smarter about S3 classes: do not redefine summary and print.
- add bias and variance from Bao (2009).
- support estimation of higher order moments in as.sr, and expands methods for se and confidence interval computations.
- incorporate higher order methods into one sample sr tests.
Changes in **SharpeR** Version 1.1.0 (2016-03-14)
- fix \texttt{sr\_vcov} on array input.
- add SRIC method.
- add SRIC to \texttt{print.sropt}.
- change \texttt{predint} output to matrix.

Changes in **SharpeR** Version 1.0.0 (2015-06-18)
- sane version numbers.
- unpaired \texttt{k} sample test of Sharpe.
- rely on same for unpaired 2 sample test.
- prediction intervals for Sharpe based on upsilon.
- more tests.

Changes in **SharpeR** Version 0.1501 (2014-12-06)
- fix inference of mark frequency from e.g. \texttt{xts} objects.
- add \texttt{rlambdap}.

Changes in **SharpeR** Version 0.1401 (2014-01-05)
- fix second moment asymptotic covariance.
- add confidence distribution functions for \texttt{sr}, \texttt{sr\_opt}.

Changes in **SharpeR** Version 0.1310 (2013-10-30)
- inverse second moment asymptotic covariance.

Changes in **SharpeR** Version 0.1309 (2013-09-20)
- spanning/hedging tests.
- \texttt{sr} equality test via callback variance covariance computation.
- split vignette in two.

Changes in **SharpeR** Version 0.1307 (2013-05-30)
- proper d.f. in \texttt{sr} objects with different nan fill.
- restore vignette.

**SharpeR Initial Version** 0.1306 (2013-05-21)
- put on CRAN
Compute variance covariance of 'Unified' Second Moment

Description

Computes the variance covariance matrix of sample mean and second moment.

Usage

\[
\text{sm_vcov}(X, \text{vcov.func}=\text{vcov}, \text{fit.intercept}=\text{TRUE})
\]

Arguments

- **X**: an \(n \times p\) matrix of observed returns.
- **vcov.func**: a function which takes an object of class \text{lm}, and computes a variance-covariance matrix. If equal to the string "normal", we assume multivariate normal returns.
- **fit.intercept**: a boolean controlling whether we add a column of ones to the data, or fit the raw uncentered second moment.

Details

Given \(p\)-vector \(x\), the 'unified' sample is the \(p(p + 3)/2\) vector of \(x\) stacked on top of vech\((xx^\top)\).

Given \(n\) contemporaneous observations of \(p\)-vectors, stacked as rows in the \(n \times p\) matrix \(X\), this function computes the mean and the variance-covariance matrix of the 'unified' sample.

One may use the default method for computing covariance, via the \text{vcov} function, or via a 'fancy' estimator, like \text{sandwich:vcovHAC}, \text{sandwich:vcovHC}, etc.

Value

- a list containing the following components:
  - **mu**: a \(q = p(p + 3)/2\) vector of the mean, then the vech'd second moment of the sample data.
  - **Ohat**: the \(q \times q\) estimated variance covariance matrix. Only the informative part is returned: one may assume a row and column of zeros in the upper left.
  - **n**: the number of rows in \(X\).
  - **p**: the number of assets.

Note

This function will be deprecated in future releases of this package. Users should migrate at that time to a similar function in the MarkowitzR package.

Author(s)

Steven E. Pav <shabbychef@gmail.com>
sr

Create an 'sr' object.

Description

Spawns an object of class sr.

Usage

sr(sr, df, c0 = 0, ope = 1, rescal = sqrt(1/(df + 1)), epoch = "yr",
   cumulants = NULL)
Arguments

sr  a Sharpe ratio statistic.

df  the degrees of freedom of the equivalent t-statistic.

c0  the 'risk-free' or 'disastrous' rate of return. This is assumed to be given in the
    same units as x, *not* in 'annualized' terms.

ope  the number of observations per 'epoch'. For convenience of interpretation, the
    Sharpe ratio is typically quoted in 'annualized' units for some epoch, that is,
    'per square root epoch', though returns are observed at a frequency of ope per
    epoch. The default value is 1, meaning the code will not attempt to guess what
    the observation frequency is, and no annualization adjustments will be made.

rescal  the rescaling parameter.

epoch  the string representation of the 'epoch', defaulting to 'yr'.

cumulants  an optional array of the higher order cumulants of the returns distribution. The
    first element shall be the skew; the second the excess kurtosis. Up to the sixth
    cumulant can be given. Higher order approximations for the moments of the
    Sharpe ratio can be computed based on these cumulants.

Details

The sr class contains information about a rescaled t-statistic. The following are list attributes of the
object:

sr  The Sharpe ratio statistic.

df  The d.f. of the equivalent t-statistic.

c0  The drag 'risk free rate' used.

ope  The 'observations per epoch'.

rescal  The rescaling parameter.

epoch  The string name of the 'epoch'.

The stored Sharpe statistic, sr is equal to the t-statistic times rescal ∗ sqrt(ope).
For the most part, this constructor should *not* be called directly, rather as.sr should be called
instead to compute the Sharpe ratio.

Value

a list cast to class sr.

Note

2FIX: allow rownames?

Author(s)

Steven E. Pav <shabbychef@gmail.com>
References

See Also
reannualize
as.sr

Other sr: as.sr, confint.sr, dsr, is.sr, plambdap, power.sr_test, predint, print.sr, reannualize, se, sr_equality_test, sr_test, sr_unpaired_test, sr_vcov, summary.sr

Examples
# roll your own.
ope <- 253
zeta <- 1.0
n <- 3 * ope
rvs <- rsr(1,n,zeta,ope=ope)
roll.own <- sr(s=rvs,df=n-1,ope=ope,rescal=sqrt(1/n))
# put a bunch in. naming becomes a problem.
rvs <- rsr(5,n,zeta,ope=ope)
roll.own <- sr(s=rvs,df=n-1,ope=ope,rescal=sqrt(1/n))

sric Sharpe Ratio Information Coefficient

Description
Computes the Sharpe Ratio Information Coefficient of Paulsen and Soehl, an asymptotically unbiased estimate of the out-of-sample Sharpe of the in-sample Markowitz portfolio.

Usage
sric(z.s)

Arguments
z.s an object of type sropt

Details
Let \( X \) be an observed \( T \times k \) matrix whose rows are i.i.d. normal. Let \( \mu \) and \( \Sigma \) be the sample mean and sample covariance. The Markowitz portfolio is
\[
\mathbf{w} = \Sigma^{-1} \mathbf{\mu},
\]
which has an in-sample Sharpe of
\[
\zeta = \sqrt{\mu^\top \Sigma^{-1} \mu}.
\]
The Sharpe Ratio Information Criterion is defined as
\[
SRIC = \zeta - \frac{k - 1}{T\zeta}.
\]

The expected value (over draws of \(X\) and of future returns) of the \(SRIC\) is equal to the expected value of the out-of-sample Sharpe of the (in-sample) portfolio \(w\) (again, over the same draws.)

Value

The Sharpe Ratio Information Coefficient.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References


See Also

Other sropt Hotelling: inference

Examples

```r
# generate some sropts
nfac <- 3
cy <- 5
ope <- 253
# simulations with no covariance structure.
# under the null:
set.seed(as.integer(charToRaw("fix seed")))
Returns <- matrix(rnorm(ope*nyr*nfac,mean=0,sd=0.0125),ncol=nfac)
asro <- as.sropt(Returns,drag=0,ope=ope)
srv <- sric(asro)
```

```r
sropt
Create an 'sropt' object.
```

Description

Spawns an object of class sropt.

Usage

```
sropt(z.s, df1, df2, drag = 0, ope = 1, epoch = "yr", T2 = NULL)
```
Arguments

z.s an optimum Sharpe ratio statistic.
df1 the number of assets in the portfolio.
df2 the number of observations.
drag the 'drag' term, \(c_0/R\). defaults to 0. It is assumed that drag has been annualized, \textit{i.e.} has been multiplied by \(\sqrt{\text{ope}}\). This is in contrast to the \(c_0\) term given to sr.

ope the number of observations per 'epoch'. For convenience of interpretation, The Sharpe ratio is typically quoted in 'annualized' units for some epoch, that is, 'per square root epoch', though returns are observed at a frequency of ope per epoch. The default value is 1, meaning the code will not attempt to guess what the observation frequency is, and no annualization adjustments will be made.

epoch the string representation of the 'epoch', defaulting to 'yr'.

T2 the Hotelling \(T^2\) statistic. If not given, it is computed from the given information.

Details

The \texttt{sropt} class contains information about a rescaled \(T^2\)-statistic. The following are list attributes of the object:

\begin{itemize}
  \item \texttt{sropt} The (optimal) Sharpe ratio statistic.
  \item \texttt{df1} The number of assets.
  \item \texttt{df2} The number of observations.
  \item \texttt{drag} The drag term, which is the 'risk free rate' divided by the maximum risk.
  \item \texttt{ope} The 'observations per epoch'.
  \item \texttt{epoch} The string name of the 'epoch'.
\end{itemize}

For the most part, this constructor should \textit{not} be called directly, rather \texttt{as.sropt} should be called instead to compute the needed statistics.

Value

a list cast to class \texttt{sropt}, with the following attributes:

\begin{itemize}
  \item \texttt{sropt} the optimal Sharpe statistic.
  \item \texttt{df1} the number of assets.
  \item \texttt{df2} the number of observed vectors.
  \item \texttt{drag} the input drag term.
  \item \texttt{ope} the input ope term.
  \item \texttt{epoch} the input epoch term.
  \item \texttt{T2} the Hotelling \(T^2\) statistic.
\end{itemize}

Note

2FIX: allow rownames?
sropt_test

test for optimal Sharpe ratio

Description
Performs one sample tests of Sharpe ratio of the Markowitz portfolio.

Usage
sropt_test(X, alternative=c("greater","two.sided","less"),
            zeta.s=0, ope=1, conf.level=0.95)

Arguments
X a (non-empty) numeric matrix of data values, each row independent, each column representing an asset, or an object of class sropt.

alternative a character string specifying the alternative hypothesis. must be one of "two.sided", "greater" (default) or "less". You can specify just the initial letter.

zeta.s a number indicating the null hypothesis value.
ope the number of observations per 'epoch'. For convenience of interpretation, The Sharpe ratio is typically quoted in 'annualized' units for some epoch, that is, 'per square root epoch', though returns are observed at a frequency of ope per epoch. The default value is 1, meaning the code will not attempt to guess what the observation frequency is, and no annualization adjustments will be made.

conf.level confidence level of the interval. (not used yet)

Details

Suppose \( x_i \) are \( n \) independent draws of a \( q \)-variate normal random variable with mean \( \mu \) and covariance matrix \( \Sigma \). This code tests the hypothesis

\[
H_0 : \mu^\top \Sigma^{-1} \mu = \delta_0^2
\]

The default alternative hypothesis is the one-sided

\[
H_1 : \mu^\top \Sigma^{-1} \mu > \delta_0^2
\]

but this can be set otherwise.

Note there is no 'drag' term here since this represents a linear offset of the population parameter.

Value

A list with class "htest" containing the following components:

- statistic the value of the \( T^2 \)-statistic.
- parameter a list of the degrees of freedom for the statistic.
- p.value the p-value for the test.
- conf.int a confidence interval appropriate to the specified alternative hypothesis. NYI.
- estimate the estimated optimal Sharpe, annualized
- null.value the specified hypothesized value of the optimal Sharpe.
- alternative a character string describing the alternative hypothesis.
- method a character string indicating what type of test was performed.
- data.name a character string giving the name(s) of the data.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

See Also

- reannualize
- sr_test, t.test.

Other sropt: as.sropt, confint.sr, dsropt, is.sropt, pco_sropt, power.sropt_test, reannualize, sropt
Examples

```r
# test for uniformity
pvs <- replicate(128, { x <- sropt_test(matrix(rnorm(1000*4), ncol=4), alternative="two.sided")
                      x$p.value })
plot(ecdf(pvs))
abline(0,1,col='red')

# input a sropt objects:
nfac <- 5
nyr <- 10
ope <- 253
# simulations with no covariance structure.
# under the null:
set.seed(as.integer(charToRaw("be deterministic")))
Returns <- matrix(rnorm(ope*nyr*nfac, mean=0, sd=0.0125), ncol=nfac)
asro <- as.sropt(Returns, drag=0, ope=ope)
stest <- sropt_test(asro, alternative="two.sided")
```

---

**sr_bias**  

**sr_bias**.

---

**Description**

Computes the asymptotic bias of the sample Sharpe ratio based on moments.

**Usage**

```
sr_bias(snr, n, cumulants, type = c("simple", "second_order"))
```

**Arguments**

- **snr**  
  the population Signal Noise ratio. Often one will use the population estimate instead.

- **n**  
  the sample size that the Sharpe ratio is observed on.

- **cumulants**  
  a vector of the third through fourth, or the third through seventh population cumulants of the random variable. More terms are needed for the higher accuracy approximation.

- **type**  
  determines the order of accuracy of the bias approximation. Takes values of
  - **simple** We compute the simple approximation using only the skewness and excess kurtosis.
  - **second_order** We compute the more accurate approximation, given by Bao, which is accurate to $o(n^{-2})$.
Details

The sample Sharpe ratio has bias of the form

\[ B = \left( \frac{3}{4n} + \frac{3\gamma_2}{8n} \right) \zeta - \frac{1}{2n} \gamma_1 + o \left( n^{-3/2} \right), \]

where \( \zeta \) is the population Signal Noise ratio, \( n \) is the sample size, \( \gamma_1 \) is the population skewness, and \( \gamma_2 \) is the population excess kurtosis. This form of the bias appears as Equation (5) in Bao, which claims an accuracy of only \( o \left( n^{-1} \right) \). The author believes this approximation is slightly more accurate.

A more accurate form is given by Bao (Equation (3)) as

\[ B = \frac{3\zeta}{4n} + \frac{49\zeta}{32n^2} \gamma_1 \left( \frac{1}{2n} + \frac{3}{8n^2} \right) + \gamma_2 \zeta \left( \frac{3}{8n} - \frac{15}{32n^2} \right) + \frac{3\gamma_3}{8n^2} + \frac{5\gamma_4\zeta}{16n^2} + \frac{5\gamma_2^2\zeta}{4n^2} + \frac{105\gamma_2^2\zeta}{128n^2} - \frac{15\gamma_1\gamma_2}{16n^2} + o \left( n^{-2} \right), \]

where \( \gamma_3 \) through \( \gamma_5 \) are the fifth through seventh cumulants of the error term.

Value

the approximate bias of the Sharpe ratio. The bias is the expected value of the sample Sharpe minus the Signal Noise ratio.

Note

much of the code is adapted from Gauss code provided by Yong Bao.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References


See Also

sr_variance

Examples

# bias under normality:
sr_bias(1, 100, rep(0,2), type='simple')
sr_bias(1, 100, rep(0,5), type='second_order')

# plugging in sample estimates
x <- rnorm(1000)
n <- length(x)
mu <- mean(x)
sr_equality_test

## Description

Performs a hypothesis test of equality of Sharpe ratios of \( p \) assets given paired observations.

## Usage

```r
sr_equality_test(X, type=c("chisq","F","t"),
                  alternative=c("two.sided","less","greater"),
                  contrasts=NULL,
                  vcov.func=vcov)
```

## Arguments

- **X**
  - an \( n \times p \) matrix of paired observations.
- **type**
  - which approximation to use. "chisq" is preferred when the returns are non-normal, but the approximation is asymptotic. the "t" test is only supported when \( k = 1 \).
- **alternative**
  - a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter. This is only relevant for the "t" test. "greater" corresponds to \( H_a : E s > 0 \).
- **contrasts**
  - an \( k \times p \) matrix of the contrasts
- **vcov.func**
  - a function which takes a model of class lm (one of the form \( x \sim 1 \)), and produces a variance-covariance matrix. The default is \( \text{vcov} \), which produces a 'vanilla' estimate of covariance. Other sensible options are \( \text{vcovHAC} \) from the sandwich package.

## Details

Given \( n \ i.i.d. \) observations of the excess returns of \( p \) strategies, we test

\[
H_0 : \frac{\mu_i}{\sigma_i} = \frac{\mu_j}{\sigma_j}, 1 \leq i < j \leq p
\]

using the method of Wright, et. al.

More generally, a matrix of constrasts, \( E \) can be given, and we can test

\[
H_0 : E s = 0,
\]
where $s$ is the vector of Sharpe ratios of the $p$ strategies.

When $E$ consists of a single row (a single contrast), as is the case when the default contrasts are used and only two strategies are compared, then an approximate t-test can be performed against the alternative hypothesis $H_a : Es > 0$

Both chi-squared and F- approximations are supported; the former is described by Wright. et. al., the latter by Leung and Wong.

Value

Object of class htest, a list of the test statistic, the size of $X$, and the method noted.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References


See Also

sr_test

Other sr: as.sr, confint.sr, dsr, is.sr, plambdap, power.sr_test, predint, print.sr, reannualize, se, sr_test, sr_unpaired_test, sr_vcov, sr, summary.sr

Examples

# under the null
rv <- sr_equality_test(matrix(rnorm(500*5),ncol=5))
# under the alternative (but with identity covariance)
op <- 253
nyr <- 10
nco <- 5
rets <- 0.01 * sapply(seq(0,1.7/sqrt(ope),length.out=nco),
  function(mu) { rnorm(ope*nyr,mean=mu,sd=1) })
rv <- sr_equality_test(rets)

## Not run:
# using real data
if (require(quantmod)) {
  get.ret <- function(sym,...) {
    OHLCV <- getSymbols(sym,auto.assign=FALSE,...)
    lrets <- diff(log(OHLCV[,paste(c(sym,"Adjusted"),collapse=".",sep=""))]
    lrets[-1,]
  }
  get.rts <- function(syms,...) { some.rts <- do.call("cbind",lapply(syms,get.ret,...))
    some.rts <- get.rts(c("IBM","AAPL","NFLX","SPY"))
  }
  pvs <- sr_equality_test(some.rts)
}
# test for uniformity
pvs <- replicate(1024, { x <- sr_equality_test(matrix(rnorm(400*4)),400,5,type="chisq")
    x$p.value })
plot(ecdf(pvs))
abline(0,1,col='red')

## End(Not run)
## Not run:
if (require(sandwich)) {
  set.seed(as.integer(charToRaw("0b2fd4e9-3bdf-4e3e-9c75-25c6d18c331f")))
  n.manifest <- 10
  n.latent <- 4
  n.day <- 1024
  snr <- 0.95
  latent.rts <- matrix(rnorm(n.day*n.latent),ncol=n.latent) %*%
  matrix(runif(n.latent*n.manifest),ncol=n.manifest)
  noise.rts <- matrix(rnorm(n.day*n.manifest),ncol=n.manifest)
  some.rts <- snr * latent.rts + sqrt(1-snr^2) * noise.rts
  # naive vcov
  pvs0 <- sr_equality_test(some.rts)
  # HAC vcov
  pvs1 <- sr_equality_test(some.rts,vcov.fnc=vcovHAC)
  # more elaborately:
  pvs <- sr_equality_test(some.rts,vcov.fnc=function(amod) {
    vcovHAC(amod,prewhite=TRUE) })
}

## End(Not run)

---

**sr_test**

test for Sharpe ratio

**Description**

Performs one and two sample tests of Sharpe ratio on vectors of data.
Usage

sr_test(x, y = NULL, alternative = c("two.sided", "less", "greater"),
    zeta = 0, ope = 1, paired = FALSE, conf.level = 0.95,
    type = c("exact", "t", "Z", "Mertens", "Bao"), ...)

Arguments

x a (non-empty) numeric vector of data values, or an object of class sr, containing
    a scalar sample Sharpe estimate.
y an optional (non-empty) numeric vector of data values, or an object of class
    sr, containing a scalar sample Sharpe estimate. Only an unpaired test can be
    performed when at least one of x and y are of class sr
alternative a character string specifying the alternative hypothesis, must be one of "two.sided"
    (default), "greater" or "less". You can specify just the initial letter.
zeta a number indicating the null hypothesis offset value, the S value.
ope the number of observations per 'epoch'. For convenience of interpretation, The
    Sharpe ratio is typically quoted in 'annualized' units for some epoch, that is,
    'per square root epoch', though returns are observed at a frequency of ope per
    epoch. The default value is 1, meaning the code will not attempt to guess what
    the observation frequency is, and no annualization adjustments will be made.
paired a logical indicating whether you want a paired test.
conf.level confidence level of the interval.
type which method to apply.
... further arguments to be passed to or from methods.

Details

Given n observations $x_i$ from a normal random variable, with mean $\mu$ and standard deviation $\sigma$, tests

$$H_0 : \frac{\mu}{\sigma} = S$$

against two or one sided alternatives.

Can also perform two sample tests of Sharpe ratio. For paired observations $x_i$ and $y_i$, tests

$$H_0 : \frac{\mu_x}{\sigma_x} = \frac{\mu_y}{\sigma_y}$$

against two or one sided alternative, via sr_equality_test.

For unpaired (and independent) observations, tests

$$H_0 : \frac{\mu_x}{\sigma_x} - \frac{\mu_y}{\sigma_y} = S$$

against two or one-sided alternatives via the upsilon distribution.

The one sample test admits a number of different methods:

**exact** The default, which is only exact when returns are normal, based on inverting the non-central
    t distribution.


**t** Uses the Johnson Welch approximation to the standard error, centered around the sample value.

**Z** Uses the Johnson Welch approximation to the standard error, performing a simple correction for the bias of the Sharpe ratio based on Miller and Gehr formula.

**Mertens** Uses the Mertens higher order approximation to the standard error, centered around the sample value.

**Bao** Uses the Bao higher order approximation to the standard error, performing a higher order correction for the bias of the Sharpe ratio.

See `confint.sr` for more information on these types

**Value**

A list with class "htest" containing the following components:

- `statistic` the value of the t- or Z-statistic.
- `parameter` the degrees of freedom for the statistic.
- `p.value` the p-value for the test.
- `conf.int` a confidence interval appropriate to the specified alternative hypothesis. NYI for some cases.
- `estimate` the estimated Sharpe or difference in Sharpes depending on whether it was a one-sample test or a two-sample test. Annualized
- `null.value` the specified hypothesized value of the Sharpe or difference of Sharpes depending on whether it was a one-sample test or a two-sample test.
- `alternative` a character string describing the alternative hypothesis.
- `method` a character string indicating what type of test was performed.
- `data.name` a character string giving the name(s) of the data.

**Author(s)**

Steven E. Pav <shabbychef@gmail.com>

**References**


**See Also**

- `reannualize`
- `sr_equality_test, sr_unpaired_test, t.test`

Other sr: `as.sr, confint.sr, dsr, is.sr, plambdap, power.sr_test, predint, print.sr, reannualize, se, sr_equality_test, sr_unpaired_test, sr_vcov, sr, summary.sr`
Examples

# should reject null
x <- sr_test(rnorm(1000, mean=0.5, sd=0.1), zeta=2, ope=1, alternative="greater")

# should not reject null
x <- sr_test(rnorm(1000, mean=0.5, sd=0.1), zeta=2, ope=1, alternative="less")

# test for uniformity
pvs <- replicate(128, x <- sr_test(rnorm(1000), ope=253, alternative="two.sided")
x$p.value )

plot(ecdf(pvs))
abline(0,1,col='red')

# testing an object of class sr
asr <- as.sr(rnorm(1000,1/sqrt(253)), ope=253)
checkit <- sr_test(asr, zeta=0)


sr_unpaired_test  test for equation on unpaired Sharpe ratios

Description

Performs hypothesis tests on a single equation on k independent samples of Sharpe ratio.

Usage

sr_unpaired_test(srs, contrasts = NULL, null.value = 0,
  alternative = c("two.sided", "less", "greater"), ope = NULL,
  conf.level = 0.95)

Arguments

srs  a (non-empty) list of objects of class sr, each containing a scalar sample Sharpe estimate. Or a single object of class sr with multiple Sharpe estimates. If the sr objects have different annualizations (ope parameters), a warning is thrown, since it is presumed that the contrasts all have the same units, but the test proceeds.

contrasts an array of the constrasts, the a_j values. Defaults to c(1,-1,1,...).

null.value the constant null value, the b. Defaults to 0.

alternative a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.

ope the number of observations per 'epoch'. For convenience of interpretation, The Sharpe ratio is typically quoted in 'annualized' units for some epoch, that is, 'per square root epoch', though returns are observed at a frequency of ope per epoch. The default value is to take the same ope from the input srs object, if it is unambiguous. Otherwise, it defaults to 1, with a warning thrown.

conf.level confidence level of the interval.

... further arguments to be passed to or from methods.
Details

For $1 \leq j \leq k$, suppose you have $n_j$ observations of a normal random variable with mean $\mu_j$ and standard deviation $\sigma_j$, with all observations independent. Given constants $a_j$ and value $b$, this code tests the null hypothesis

$$H_0 : \sum_j a_j \frac{\mu_j}{\sigma_j} = b$$

against two or one sided alternatives.

Value

A list with class "htest" containing the following components:

- statistic: NULL here.
- parameter: a list of upsilon parameters.
- p.value: the p-value for the test.
- conf.int: a confidence interval appropriate to the specified alternative hypothesis.
- estimate: the estimated equation value, just the weighted sum of the sample Sharpe ratios.
- null.value: the specified hypothesized value of the sum of Sharpes.
- alternative: a character string describing the alternative hypothesis.
- method: a character string indicating what type of test was performed.
- data.name: a character string giving the name(s) of the data.

Note

This code is based on the ‘upsilon’ code from sadists, which may be inaccurate for a large number of series. Take caution when applying this test. File a bug report if you are negatively impacted.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References


See Also

sr_equality_test, sr_test, t.test.

Other sr: as.sr, confint.sr, dsr, is.sr, plambdap, power.sr_test, predint, print.sr, reannualize, se, sr_equality_test, sr_test, sr_vcov, sr, summary.sr
Examples

# basic usage
set.seed(as.integer(charToRaw("set the seed")))
# default contrast is 1,-1,1,-1,1,-1
etc <- sr_unpaired_test(as.sr(matrix(rnorm(1000*6,mean=0.02,sd=0.1),ncol=6)))
print(etc)

etc <- sr_unpaired_test(as.sr(matrix(rnorm(1000*4,mean=0.0005,sd=0.01),ncol=4)),
  alternative='greater')
print(etc)

etc <- sr_unpaired_test(as.sr(matrix(rnorm(1000*4,mean=0.0005,sd=0.01),ncol=4)),
  contrasts=c(1,1,1),null.value=-0.1,alternative='greater')
print(etc)

inp <- list(as.sr(rnorm(500)),as.sr(runif(200)-0.5),
  as.sr(rnorm(30)),as.sr(rnorm(100)))
etc <- sr_unpaired_test(inp)

inp <- list(as.sr(rnorm(500)),as.sr(rnorm(100,mean=0.2,sd=1)))
etc <- sr_unpaired_test(inp,contrasts=c(1,1),null.value=0.2)
etc$conf.int

sr_variance

Description

Computes the variance of the sample Sharpe ratio.

Usage

sr_variance(snr, n, cumulants)

Arguments

snr  
the population Signal Noise ratio. Often one will use the population estimate instead.

n  
the sample size that the Sharpe ratio is observed on.

cumulants  
a vector of the third through fourth, or the third through seventh population cumulants of the random variable. More terms are needed for the higher accuracy approximation.
Details

The sample Sharpe ratio has variance of the form

\[
V = \frac{1}{n} \left( 1 + \frac{\zeta^2}{2} \right) + \frac{1}{n^2} \left( \frac{19\zeta^2}{8} + 2 \right) - \gamma_1 \zeta \left( \frac{1}{n} + \frac{5}{2n^2} \right) + \gamma_2 \zeta^2 \left( \frac{1}{4n} + \frac{3}{8n^2} \right) + \frac{5\gamma_3 \zeta}{4n^2} + \gamma_1^2 \left( \frac{7}{4n^2} - \frac{3\zeta^2}{2n^2} \right) + \frac{39\gamma_2^2 \zeta^2}{32n^2} - \frac{15\gamma_1 \gamma_2 \zeta^4}{4n^2} + o\left(\frac{1}{n^2}\right),
\]

where \( \zeta \) is the population Signal Noise ratio, \( n \) is the sample size, \( \gamma_1 \) is the population skewness, and \( \gamma_2 \) is the population excess kurtosis, and \( \gamma_3 \) through \( \gamma_5 \) are the fifth through seventh cumulants of the error term. This form of the variance appears as Equation (4) in Bao.

Value

the variance of the sample statistic.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References


See Also

sr_bias.

Examples

```r
# variance under normality:
sr_variance(1, 100, rep(0,5))
```

---

**sr_vcov**

*Compute variance covariance of Sharpe Ratios.*

**Description**

Computes the variance covariance matrix of sample Sharpe ratios.

**Usage**

```r
sr_vcov(X, vcov.func=vcov, ope=1)
```
Arguments

- **x**: an $n \times p$ matrix of observed returns. If not a matrix, but a numeric of length $n$, then it is coerced into a $n \times 1$ matrix.
- **vcov.func**: a function which takes an object of class *lm*, and computes a variance-covariance matrix.
- **ope**: the number of observations per 'epoch'. For convenience of interpretation, the Sharpe ratio is typically quoted in 'annualized' units for some epoch, that is, 'per square root epoch', though returns are observed at a frequency of $\text{ope}$ per epoch. The default value is 1, meaning the code will not attempt to guess what the observation frequency is, and no annualization adjustments will be made.

Details

Given $n$ contemporaneous observations of $p$ returns streams, this function estimates the asymptotic variance covariance matrix of the vector of sample Sharpes, $[\zeta_1, \zeta_2, \ldots, \zeta_p]$.

One may use the default method for computing covariance, via the `vcov` function, or via a 'fancy' estimator, like `sandwich::vcovHAC`, `sandwich::vcovHC`, etc.

This code first estimates the covariance of the $2p$ vector of the vector $x$ stacked on its Hadamard square, $x^2$. This is then translated back to a variance covariance on the vector of sample Sharpe ratios via the Delta method.

Value

A list containing the following components:

- **SR**: a vector of (annualized) Sharpe ratios.
- **Ohat**: a $p \times p$ variance covariance matrix.
- **p**: the number of assets.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References


See Also

- `reannualize`
- `sr_distributions` functions, `dsr`

Other sr: `as_sr`, `confint_sr`, `dsr`, `is_sr`, `plambdap`, `power_sr_test`, `predint`, `print_sr`, `reannualize`, `se`, `sr_equality_test`, `sr_test`, `sr_unpaired_test`, `sr_summary_sr`
Examples

```r
X <- matrix(rnorm(1000*3),ncol=3)
colnames(X) <- c("ABC","XYZ","WORM")
Sigmas <- sr_vcov(X)
# make it fat tailed:
X <- matrix(rt(1000*3,df=5),ncol=3)
Sigmas <- sr_vcov(X)
## Not run:
if (require(sandwich)) {
  Sigmas <- sr_vcov(X,vcov.func=vcovHC)
}

## End(Not run)
# add some autocorrelation to X
Xf <- filter(X,c(0.2),"recursive")
colnames(Xf) <- colnames(X)
Sigmas <- sr_vcov(Xf)
## Not run:
if (require(sandwich)) {
  Sigmas <- sr_vcov(Xf,vcov.func=vcovHAC)
}

## End(Not run)
# should run for a vector as well
X <- rnorm(1000)
SS <- sr_vcov(X)
```

summary.sr

### Summary

**Summary a Sharpe, or (delta) optimal Sharpe object.**

#### Description

Computes a ‘summary’ of an object, adding in some statistics.

#### Usage

```r
## S3 method for class 'sr'
summary(object, ...)

## S3 method for class 'sropt'
summary(object, ...)
```

#### Arguments

- **object**: an object of class `sr`, `sropt` or `del_sropt`.
- **...**: additional arguments affecting the summary produced, though ignored here.
summary.sr

Details

Enhances an object of class sr, sropt or del_sropt to also include t- or T-statistics, p-values, and so on.

Value

When an sr object is input, the object cast to class summary.sr with some additional fields:

- **tval** the equivalent t-statistic.
- **pval** the p-value under the null.
- **serr** the standard error of the Sharpe ratio.

When an sropt object is input, the object cast to class summary.sropt with some additional fields:

- **pval** the p-value under the null.
- **SRIC** the SRIC value, see sric.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References


See Also

- print.sr.
- Other sr: as.sr, confint.sr, dsr, is.sr, plambdap, power.sr_test, predint, print.sr, reannualize, se, sr_equality_test, sr_test, sr_unpaired_test, sr_vcov, sr

Examples

# Sharpe's 'model': just given a bunch of returns.
set.seed(1234)
asr <- as.sr(rnorm(253*3), ope=253)
summary(asr)
Index

*Topic distribution
  dsr, 18
dsropt, 20
pco_sropt, 29
plambdap, 31
*Topic htest
  confint.sr, 13
  inference, 22
  power.sr_test, 34
  power.sropt_test, 33
  se, 41
  sr_equality_test, 54
  sr_test, 56
  sr_unpaired_test, 59
  sropt_test, 50
*Topic package
  SharperR-package, 2
*Topic univar
  as.del_sropt, 6
  as.sr, 8
  as.sropt, 10
  del_sropt, 16
  ism_vcov, 27
  sm_vcov, 44
  sr, 45
  sr_vcov, 62
  sropt, 48

  as.del_sropt, 6, 16, 17, 25
  as.sr, 8, 14, 19, 26, 32, 36, 37, 39, 40, 42, 46,
  47, 55, 58, 60, 63, 65
  as.sropt, 10, 14, 22, 27, 30, 34, 40, 49–51

  confint, 14
  confint.del_sropt (confint.sr), 13
  confint.sr, 10, 12, 13, 19, 22, 26, 27, 30, 32,
  34, 36, 37, 39, 40, 42, 47, 50, 51, 55, 58, 60, 63, 65
  confint.sropt (confint.sr), 13
  del_sropt, 7, 16, 25
df, 22, 23
dsropt, 10, 14, 18, 22, 26, 32, 36, 37, 39, 40, 42,
  47, 55, 58, 60, 63, 65
dsropt_test, 12, 14, 20, 27, 30, 34, 40, 50, 51

  dt, 19, 32
  inference, 22, 48
  is.del_sropt, 7, 17, 24
  is.sr, 10, 14, 19, 25, 32, 36, 37, 39, 40, 42,
  47, 55, 58, 60, 63, 65
  is.sropt, 12, 14, 22, 26, 30, 34, 40, 50, 51
  ism_vcov, 27, 45
  pco_sropt, 12, 14, 22, 27, 29, 34, 40, 50, 51
  pf, 22
  plambdap, 10, 14, 19, 26, 31, 36, 37, 39, 40,
  42, 47, 55, 58, 60, 63, 65
  power.sr_test, 10, 14, 19, 26, 32, 34, 37, 39,
  40, 42, 47, 55, 58, 60, 63, 65
  power.sropt_test, 12, 14, 22, 27, 30, 33, 40,
  50, 51
  power.t.test, 34–36
  predint, 10, 14, 19, 26, 32, 36, 39, 40, 42,
  47, 55, 58, 60, 63, 65
  print.del_sropt (print.sr), 38
  print.sr, 10, 14, 19, 26, 32, 36, 37, 38, 40,
  42, 47, 55, 58, 60, 63, 65
  print.sropt (print.sr), 38
  psr, 10, 22
  psr (dsr), 18
  psropt, 12, 30
  psropt (dsropt), 20
  pt, 19, 32
  qco_sropt (pco_sropt), 29
  qf, 22
  qlambdap (plambdap), 31
  qsr, 10, 22
  qsr (dsr), 18
INDEX

qsropt, 12, 30
qsropt (dsropt), 20
qt, 19, 32
reannualize, 10, 12, 14, 17, 19, 22, 26, 27, 30, 32, 34, 36, 37, 39, 42, 47, 50, 51, 55, 58, 60, 63, 65
rft, 22
rlambdap (plambdap), 31
rsr, 10, 22
rsr (dsr), 18
rsropt, 12, 30
rsropt (dsropt), 20
rt, 19, 32
se, 10, 14, 19, 26, 32, 36, 37, 39, 40, 41, 47, 55, 58, 60, 63, 65
sharperMnews, 42
sharperM-package, 2
sm_vcov, 28, 44
sr, 6, 7, 10–12, 14, 16, 19, 26, 32, 36, 37, 39, 40, 42, 45, 49, 55, 58, 60, 63, 65
sr_bias, 52, 62
sr_equality_test, 10, 14, 19, 26, 32, 36, 37, 39, 40, 42, 47, 54, 57, 58, 60, 63, 65
sr_test, 10, 14, 19, 26, 32, 36, 37, 39, 40, 42, 47, 51, 55, 56, 60, 63, 65
sr_unpaired_test, 10, 14, 19, 26, 32, 36, 37, 39, 40, 42, 47, 55, 58, 59, 63, 65
sr_variance, 42, 53, 61
sr_vcov, 10, 14, 19, 26, 28, 32, 36, 37, 39, 40, 42, 45, 47, 55, 58, 60, 62, 65
sric, 23, 47, 65
sropt, 7, 12, 14, 22, 27, 30, 34, 40, 48, 51
sropt_test, 12, 14, 22, 27, 30, 34, 40, 50, 50
summary.sr, 10, 14, 19, 26, 32, 36, 37, 39, 40, 42, 47, 55, 58, 60, 63, 64
summary.sropt (summary.sr), 64
t.test, 51, 58, 60
vcov, 27, 44, 54, 63