SharpeR

Steven E. Pav *

August 12, 2013

Abstract

The SharpeR package provides basic functionality for testing significance of the Sharpe ratio of a series of returns, and of the Markowitz portfolio on a number of possibly correlated assets.[14] The goal of the package is to make it simple to estimate profitability (in terms of risk-adjusted returns) of strategies or asset streams.

1 The Sharpe ratio and Optimal Sharpe ratio

Sharpe defined the 'reward to variability ratio', now known as the 'Sharpe ratio', as the sample statistic

$$\hat{\zeta} = \frac{\hat{\mu}}{\hat{\sigma}},$$

where $\hat{\mu}$ is the sample mean, and $\hat{\sigma}$ is the sample standard deviation. [14] The Sharpe ratio was later redefined to include a 'risk-free' or 'disastrous rate of return': $\hat{\zeta} = (\hat{\mu} - r_0)/\hat{\sigma}$.

It is little appreciated in quantitative finance that the Sharpe ratio is identical to the sample statistic proposed by Gosset in 1908 to test for zero mean when the variance is unknown. [3] The 't-test' we know today, which includes an adjustment for sample size, was formulated later by Fisher. [2] Knowing that the Sharpe ratio is related to the t-statistic provides a 'natural arbitrage,' since the latter has been extensively studied. Many of the interesting properties of the t-statistic can be translated to properties about the Sharpe ratio.

Also little appreciated is that the multivariate analogue of the t-statistic, Hotelling's T^2, is related to the Markowitz portfolio. Consider the following portfolio optimization problem:

$$\max_{\hat{\boldsymbol{w}}:\hat{\boldsymbol{w}}^\top \hat{\boldsymbol{\Sigma}} \hat{\boldsymbol{w}} \leq R^2} \frac{\hat{\boldsymbol{w}}^\top \hat{\boldsymbol{\mu}} - r_0}{\sqrt{\hat{\boldsymbol{w}}^\top \hat{\boldsymbol{\Sigma}} \hat{\boldsymbol{w}}}}, \tag{1}$$

where $\hat{\boldsymbol{\mu}}$, $\hat{\boldsymbol{\Sigma}}$ are the sample mean vector and covariance matrix, r_0 is the risk-free rate, and R is a cap on portfolio 'risk' as estimated by $\hat{\boldsymbol{\Sigma}}$. (Note this differs from the traditional definition of the problem which imposes a 'self-financing constraint' which does not actually bound portfolio weights.) The solution to this problem is

$$\hat{\boldsymbol{w}}_* = \frac{R}{\sqrt{\hat{\boldsymbol{\mu}}^\top \hat{\boldsymbol{\Sigma}}^{-1} \hat{\boldsymbol{\mu}}}} \hat{\boldsymbol{\Sigma}}^{-1} \hat{\boldsymbol{\mu}}.$$

* shabbychef@gmail.com

The Sharpe ratio of this portfolio is

$$\hat{\zeta}_* = \frac{\hat{\boldsymbol{w}}_*{}^\top \hat{\boldsymbol{\mu}} - r_0}{\sqrt{\hat{\boldsymbol{w}}_*{}^\top \hat{\boldsymbol{\Sigma}} \hat{\boldsymbol{w}}_*}} = \sqrt{\hat{\boldsymbol{\mu}}^\top \hat{\boldsymbol{\Sigma}}^{-1} \hat{\boldsymbol{\mu}}} - \frac{r_0}{R} = \sqrt{T^2/n} - \frac{r_0}{R},$$

where T^2 is Hotelling's statistic, and n is the number of independent observations (*e.g.*, 'days') used to construct $\hat{\boldsymbol{\mu}}$. The term r_0/R is a deterministic 'drag' term that merely shifts the location of $\hat{\zeta}_*$, and so we can (mostly) ignore it when testing significance of $\hat{\zeta}_*$.

Under the (typically indefensible) assumptions that the returns are generated *i.i.d.* from a normal distribution (multivariate normal in the case of the portfolio problem), the distributions of $\hat{\zeta}$ and $\hat{\zeta}_*^2$ are known, and depend on the sample size and the population analogues, ζ and ζ_*^2. In particular, they are distributed as rescaled non-central t and F distributions. Under these assumptions on the generating processes, we can perform inference on the population analogues using the sample statistics.

The importance of each of these assumptions (*viz.* homoskedasticity, independence, normality, *etc.*) can and should be checked. [11, 13] The reader must be warned that this package is distributed without any warranty of any kind, and in no way should any analysis performed with this package be interpreted as implicit investment advice by the author(s).

The units of $\hat{\mu}$ are 'returns per time,' while those of $\hat{\sigma}$ are 'returns per square root time.' Consequently, the units of $\hat{\zeta}$ are 'per square root time.' Typically the Sharpe ratio is quoted in 'annualized' terms, *i.e.*, $\mathrm{yr}^{-1/2}$, but the units are omitted. I believe that units should be included as it avoids ambiguity, and simplifies conversions.

There is no clear standard whether arithmetic or geometric returns should be used in the computation of the Sharpe ratio. Since arithmetic returns are always greater than the equivalent geometric returns, one would suspect that arithmetic returns are *always* used when advertising products. However, I suspect that geometric returns are more frequently used in the analysis of strategies. Geometric returns have the attractive property of being 'additive', meaning that the geometric return of a period is the sum of those of subperiods, and thus the sign of the arithmetic mean of some geometric returns indicates whether the final value of a strategy is greater than the initial value. Oddly, the arithmetic mean of arithmetic returns does not share this property.

On the other hand, arithmetic returns are indeed additive *contemporaneously*: if $\boldsymbol{x}$ is the vector of arithmetic returns of several stocks, and $\hat{\boldsymbol{w}}$ is the dollar proportional allocation into those stocks at the start of the period, then $\boldsymbol{x}^\top \hat{\boldsymbol{w}}$ is the arithmetic return of the portfolio over that period. This holds even when the portfolio holds some stocks 'short.' Often this portfolio accounting is misapplied to geometric returns without even an appeal to Taylor's theorem.

2 Using the `sr` Class

An `sr` object encapsulates one or more Sharpe ratio statistics, along with the degrees of freedom, the rescaling to a t statistic, and the annualization and units information. One can simply stuff this information into an `sr` object, but it is more straightforward to allow `as.sr` to compute the Sharpe ratio for you.

```
library(SharpeR)
# suppose you computed the Sharpe of your
# strategy to be 1.3 / sqrt(yr), based on 1200
# daily observations.  store them as follows:
my.sr <- sr(sr = 1.3, df = 1200 - 1, ope = 252, epoch = "yr")
print(my.sr)

##        SR/sqrt(yr) Std. Error t value Pr(>t)
## Sharpe        1.30       0.46     2.8 0.0023 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

# multiple strategies can be tracked as well.
# one can attach names to them.
srstats <- c(0.5, 1.2, 0.6)
dim(srstats) <- c(3, 1)
rownames(srstats) <- c("strat. A", "strat. B", "benchmark")
my.sr <- sr(srstats, df = 1200 - 1, ope = 252, epoch = "yr")
print(my.sr)

##           SR/sqrt(yr) Std. Error t value Pr(>t)
## strat. A         0.50       0.46     1.1 0.1377
## strat. B         1.20       0.46     2.6 0.0045 **
## benchmark        0.60       0.46     1.3 0.0953 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Throughout, `ope` stands for 'Observations Per Epoch', and is the (average) number of returns observed per the annualization period, called the `epoch`. At the moment there is not much hand holding regarding these parameters: no checking is performed for sane values.

The `as.sr` method will compute the Sharpe ratio for you, from numeric, `data.frame`, `xts` or `lm` objects. In the latter case, it is assumed one is performing an attribution model, and the statistic of interest is the fit of the `(Intercept)` term divided by the residual standard deviation. Here are some examples:

```
set.seed(as.integer(charToRaw("set the seed")))
# Sharpe's 'model': just given a bunch of
# returns.
returns <- rnorm(253 * 8, mean = 3e-04, sd = 0.01)
asr <- as.sr(returns, ope = 253, epoch = "yr")
print(asr)

##   SR/sqrt(yr) Std. Error t value Pr(>t)
## x        0.56       0.35     1.6  0.056 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

# a data.frame with a single strategy
asr <- as.sr(data.frame(my.strategy = returns), ope = 253,
    epoch = "yr")
print(asr)

##             SR/sqrt(yr) Std. Error t value Pr(>t)
```

```
## my.strategy        0.56       0.35     1.6  0.056 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

When a `data.frame` with multiple columns is given, the Sharpe ratio of each is computed, and they are all stored:

```
# a data.frame with multiple strategies
asr <- as.sr(data.frame(strat1 = rnorm(253 * 8), strat2 = rnorm(253 *
    8, mean = 4e-04, sd = 0.01)), ope = 253, epoch = "yr")
print(asr)

##        SR/sqrt(yr) Std. Error t value Pr(>t)
## strat1      -0.043      0.354   -0.12  0.549
## strat2       0.573      0.354    1.62  0.053 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Here is an example using `xts` objects. In this case, if the `ope` is not given, it is inferred from the time marks of the input object.

```
require(quantmod)
# get price data, compute log returns on adjusted
# closes
get.ret <- function(sym, warnings = FALSE, ...) {
    # getSymbols.yahoo will barf sometimes; do a
    # trycatch
    trynum <- 0
    while (!exists("OHCLV") && (trynum < 7)) {
        trynum <- trynum + 1
        try(OHLCV <- getSymbols(sym, auto.assign = FALSE,
            warnings = warnings, ...), silent = TRUE)
    }
    adj.names <- paste(c(sym, "Adjusted"), collapse = ".",
        sep = "")
    if (adj.names %in% colnames(OHLCV)) {
        adj.close <- OHLCV[, adj.names]
    } else {
        # for DJIA from FRED, say.
        adj.close <- OHLCV[, sym]
    }
    rm(OHLCV)
    # rename it
    colnames(adj.close) <- c(sym)
    adj.close <- adj.close[!is.na(adj.close)]
    lrets <- diff(log(adj.close))
    # chop first
    lrets[-1, ]
}
get.rets <- function(syms, ...) {
    some.rets <- do.call("cbind", lapply(syms, get.ret,
        ...))
}
```

```
require(quantmod)
# quantmod::periodReturn does not deal properly
# with multiple columns, and the straightforward
# apply(mtms,2,periodReturn) barfs
my.periodReturn <- function(mtms, ...) {
    per.rets <- do.call(cbind, lapply(mtms, function(x) {
        retv <- periodReturn(x, ...)
        colnames(retv) <- colnames(x)
        return(retv)
    }))
}
# convert log return to mtm, ignoring NA
lr2mtm <- function(x, ...) {
    x[is.na(x)] = 0
    exp(cumsum(x))
}
```

```
some.rets <- get.rets(c("IBM", "AAPL", "XOM"), from = "2007-01-01",
    to = "2013-01-01")
print(as.sr(some.rets))

##      SR/sqrt(yr) Std. Error t value Pr(>t)
## IBM         0.54       0.41    1.31  0.095 .
## XOM         0.15       0.41    0.36  0.360
## AAPL        0.83       0.41    2.02  0.022 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The annualization of an `sr` object can be changed with the `reannualize` method. The name of the epoch and the observation rate can both be changed. Changing the annualization will not change statistical significance, it merely changes the units.

```
yearly <- as.sr(some.rets[, "XOM"])
monthly <- reannualize(yearly, new.ope = 21, new.epoch = "mo.")
print(yearly)

##     SR/sqrt(yr) Std. Error t value Pr(>t)
## XOM        0.15       0.41    0.36   0.36

# significance should be the same, but units
# changed.
print(monthly)

##     SR/sqrt(mo.) Std. Error t value Pr(>t)
## XOM        0.042      0.118    0.36   0.36
```

2.1 Attribution Models

When an object of class `lm` is given to `as.sr`, the fit `(Intercept)` term is divided by the residual volatility to compute something like the Sharpe ratio. In

terms of Null Hypothesis Significance Testing, nothing is gained by summarizing the `sr` object instead of the `lm` object. However, confidence intervals on the Sharpe ratio are quoted in the more natural units of reward to variability, and in annualized terms (or whatever the epoch is.)

As an example, here I perform a CAPM attribution to the monthly returns of `AAPL`. Note that the statistical significance here is certainly tainted by selection bias, a topic beyond the scope of this note.

```
# get the returns (see above for the function)
aapl.rets <- get.rets(c("AAPL", "SPY"), from = "2003-01-01",
    to = "2013-01-01")
# make them monthly:
mo.rets <- my.periodReturn(lr2mtm(aapl.rets), period = "monthly",
    type = "arithmetic")
rm(aapl.rets)  # cleanup
# look at both of them together:
both.sr <- as.sr(mo.rets)
print(both.sr)

##      SR/sqrt(yr) Std. Error t value  Pr(>t)
## SPY         0.51       0.32     1.6   0.055 .
## AAPL        1.36       0.33     4.3 1.7e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

# confindence intervals on the Sharpe:
print(confint(both.sr))

##      2.5 % 97.5 %
## SPY  -0.11    1.1
## AAPL  0.72    2.0

# perform a CAPM attribution, using SPY as 'the
# market'
linmod <- lm(AAPL ~ SPY, data = mo.rets)
# convert attribution model to Sharpe
CAPM.sr <- as.sr(linmod, ope = both.sr$ope, epoch = "yr")
# statistical significance does not change
# (though note the sr summary prints a 1-sided
# p-value)
print(summary(linmod))

##
## Call:
## lm(formula = AAPL ~ SPY, data = mo.rets)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.27110 -0.06408  0.00541  0.05054  0.30138
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.03374    0.00842    4.01  0.00011 ***
## SPY          1.31222    0.19489    6.73  6.4e-10 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.091 on 118 degrees of freedom
## Multiple R-squared:  0.278,Adjusted R-squared:  0.271
## F-statistic: 45.3 on 1 and 118 DF,  p-value: 6.36e-10

print(CAPM.sr)

##        SR/sqrt(yr) Std. Error t value  Pr(>t)
## linmod        1.28       0.33       4 5.4e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

# the confidence intervals tell the same story,
# but in different units:
print(confint(linmod, "(Intercept)"))

##             2.5 % 97.5 %
## (Intercept) 0.017   0.05

print(confint(CAPM.sr))

##        2.5 % 97.5 %
## linmod  0.63    1.9
```

2.2 Testing Sharpe and Power

The function `sr_test` performs one- and two-sample tests for significance of Sharpe ratio. Paired tests for equality of Sharpe ratio can be performed via the `sr_equality_test`, which applies the tests of Leung *et al.* or of Wright *et al.* [9, 17]

```
# get the sector 'spiders'
secto.rets <- get.rets(c("XLY", "XLE", "XLP", "XLF",
    "XLV", "XLI", "XLB", "XLK", "XLU"), from = "2003-01-01",
    to = "2013-01-01")
# make them monthly:
mo.rets <- my.periodReturn(lr2mtm(secto.rets), period = "monthly",
    type = "arithmetic")
# one-sample test on utilities:
XLU.monthly <- mo.rets[, "XLU"]
print(sr_test(XLU.monthly), alternative = "two.sided")

##
##  One Sample sr test
##
## data:  XLU.monthly
## t = 2.4, df = 119, p-value = 0.01951
## alternative hypothesis: true signal-noise ratio is not equal to 0
## sample estimates:
##     [,1]
## XLU 0.22
```

```
## attr(,"names")
## [1] "Sharpe ratio of XLU.monthly"


# test for equality of Sharpe among the different
# spiders
print(sr_equality_test(secto.rets))

##
##  test for equality of Sharpe ratio, via chisq test
##
## data:  secto.rets
## T2 = 6.6, contrasts = 8, p-value = 0.577
## alternative hypothesis: true sum squared contrasts of SNR is not equal to 0

# perform a paired two-sample test via sr_test:
XLF.monthly <- mo.rets[, "XLF"]
print(sr_test(x = XLU.monthly, y = XLF.monthly, ope = 12,
    paired = TRUE))

##
##  Paired sr-test
##
## data:  XLU.monthly and XLF.monthly
## t = 1.8, df = 119, p-value = 0.07556
## alternative hypothesis: true difference in signal-noise ratios is not equal to 0
## sample estimates:
## difference in Sharpe ratios
##                        0.67
```

The *power* of the one-sample test for Sharpe ratio follows a form first published by Johnson and Welch:[4]

$$n = \frac{c}{\zeta^2},$$

where the constant c depends on the type I and type II rates and whether one is performing a one- or two-sided test. A handy mnemonic instantiation of this rule, similar to 'Lehr's rule,' [16, 8] is $e \approx n\zeta^2$, where ζ is the population signal-to-noise ratio, e is Euler's number, and n is the sample size, in the *same units* as ζ. That is, if one measures SNR in annualized units, then n is the number of *years*. The relative error in this approximation for determining the sample size is shown in Figure 1, as a function of ζ; the error is smaller than one percent in the tested range. Note that Euler's number appears here coincidentally, as it is nearly equal to $\left[\Phi^{-1}(0.95)\right]^2$.

```
ope <- 253
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 = 0.5, ope = ope)
    x$n/ope
})
plot(zetas, 100 * ((exp(1)/zetas^2) - ssizes)/ssizes,
```

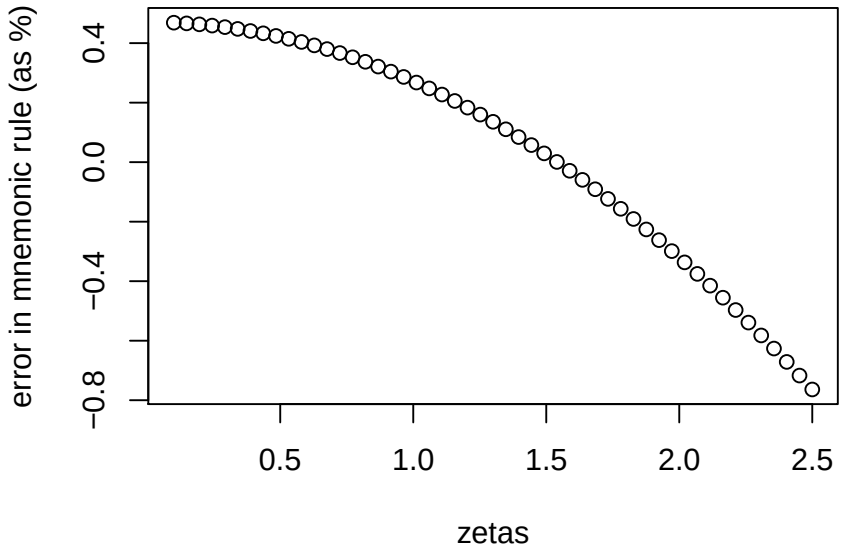

Figure 1: The percent error of the power mnemonic $e \approx n\zeta^2$ is plotted versus ζ.

```
    ylab = "error in mnemonic rule (as %)")
```

The power rules are sobering indeed. Suppose you were a hedge fund manager whose investors threatened to perform a one-sided t-test after one year. If your strategy's signal-to-noise ratio is less than $1.65\text{yr}^{-1/2}$ (a value which should be considered "very good"), your chances of 'passing' the t-test are less than fifty percent.

3 Using the `sropt` Class

The class `sropt` stores the 'optimal' Sharpe ratio, which is that of the optimal ('Markowitz') portfolio, as defined in Equation 1, as well as the relevant degrees of freedom, and the annualization parameters. Again, the constructor can be used directly, but the helper function is preferred:

```
# from a matrix object:
ope <- 253
n.stok <- 7
n.yr <- 8
# somewhat unrealistic: independent returns.
rand.rets <- matrix(rnorm(n.yr * ope * n.stok), ncol = n.stok)
asro <- as.sropt(rand.rets, ope = ope)
rm(rand.rets)
```

```
print(asro)

##        SR/sqrt(yr) T^2 value Pr(>T^2)
## Sharpe        0.61         3     0.89

# from an xts object
asro <- as.sropt(some.rets)
print(asro)

##        SR/sqrt(yr) T^2 value Pr(>T^2)
## Sharpe         0.9       4.8     0.19
```

One can compute confidence intervals for the population parameter $\zeta_* =_{\text{df}} \sqrt{\boldsymbol{\mu}^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}}$, called the 'optimal signal-noise ratio', based on inverting the non-central F-distribution. Estimates of ζ_* can also be computed, via Maximum Likelihood Estimation, or a 'shrinkage' estimate. [5, 15]

```
# confidence intervals:
print(confint(asro, level.lo = 0.05, level.hi = 1))

##      5 % 100 %
## [1,]   0   Inf

# estimation
print(inference(asro, type = "KRS"))

##      [,1]
## [1,] 0.57

print(inference(asro, type = "MLE"))

## [1] 0.64
```

A nice rule of thumb is that, to a first order approximation, the MLE of ζ_* is zero exactly when $\hat{\zeta}_*^2 \leq p/n$, where p is the number of assets. [5, 15] Inspection of this inequality confirms that $\hat{\zeta}_*$ and n can be expressed 'in the same units', meaning that if $\hat{\zeta}_*$ is in $\text{yr}^{-1/2}$, then n should be the number of *years*. For example, if the Markowitz portfolio on 8 assets over 7 years has a Sharpe ratio of $1\text{yr}^{-1/2}$, the MLE will be zero. This can be confirmed empirically as below.

```
ope <- 253
zeta.s <- 0.8
n.check <- 1000
df1 <- 10
df2 <- 6 * ope
rvs <- rsropt(n.check, df1, df2, zeta.s, ope, drag = 0)
roll.own <- sropt(z.s = rvs, df1, df2, drag = 0, ope = ope,
    epoch = "yr")
MLEs <- inference(roll.own, type = "MLE")
zerMLE <- MLEs <= 0
crit.value <- 0.5 * (max(rvs[zerMLE])^2 + min(rvs[!zerMLE])^2)
aspect.ratio <- df1/(df2/ope)
cat(sprintf("empirical cutoff for zero MLE is %2.2f yr^{-1}\n",
    crit.value))
```

```
## empirical cutoff for zero MLE is 1.67 yr^{-1}

cat(sprintf("the aspect ratio is %2.2f yr^{-1}\n",
    aspect.ratio))

## the aspect ratio is 1.67 yr^{-1}
```

3.1 The Haircut

Care must be taken interpreting the confidence intervals and the estimated optimal SNR. This is because ζ_* is the *maximal* population SNR achieved by any portfolio; it is at least equal to, and potentially much larger than, the SNR achieved by the portfolio based on sample statistics, $\hat{\boldsymbol{w}}_*$. There is a gap or 'haircut' due to mis-estimation of the optimal portfolio. One would suspect that this gap is worse when the true effect size (*i.e.*, ζ_*) is smaller, when there are fewer observations (n), and when there are more assets (p).

I define the haircut as the quantity

$$h =_{\text{df}} 1 - \frac{\hat{\boldsymbol{w}}_*{}^\top \boldsymbol{\mu}}{\zeta_* \sqrt{\hat{\boldsymbol{w}}_*{}^\top \boldsymbol{\Sigma} \hat{\boldsymbol{w}}_*}} = 1 - \left(\frac{\hat{\boldsymbol{w}}_*{}^\top \boldsymbol{\mu}}{\boldsymbol{\nu}_*{}^\top \boldsymbol{\mu}} \right) \left(\frac{\sqrt{\boldsymbol{\nu}_*{}^\top \boldsymbol{\Sigma} \boldsymbol{\nu}_*}}{\sqrt{\hat{\boldsymbol{w}}_*{}^\top \boldsymbol{\Sigma} \hat{\boldsymbol{w}}_*}} \right), \tag{2}$$

where $\boldsymbol{\nu}_*$ is the population optimal portfolio, positively proportional to $\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}$. Modeling the haircut is not straightforward because it is a random quantity which is not observed. That is, it mixes the unknown population parameters $\boldsymbol{\Sigma}$ and $\boldsymbol{\mu}$ with the sample quantity $\hat{\boldsymbol{w}}_*$, which is random.

When n/p is large, the following is a reasonable approximation to the distribution of h:

$$\sqrt{p-1} \tan\left(\arcsin\left(1-h\right)\right) \approx t\left(\sqrt{n}\zeta_*, p-1\right), \tag{3}$$

where $t\left(x, y\right)$ is a non-central t-distribution with non-centrality parameter x and y degrees of freedom. This approximation can be found by ignoring all variability in the sample estimate of the covariance matrix, that is by assuming that the sample optimal portfolio was computed with the *population* covariance: $\hat{\boldsymbol{w}}_* \propto \boldsymbol{\Sigma}^{-1}\hat{\boldsymbol{\mu}}$. Because mis-estimation of the covariance matrix should contribute some error, I expect that this approximation is a 'stochastic lower bound' on the true haircut. Numerical simulations, however, suggest it is a fairly tight bound for large n/p. (I would be willing to guess that the true distribution involves a non-central F-distribution, but the proof is beyond me at the moment.)

Here I look at the haircut via Monte Carlo simulations:

```
require(MASS)

# simple markowitz.
simple.marko <- function(rets) {
    mu.hat <- as.vector(apply(rets, MARGIN = 2, mean,
        na.rm = TRUE))
    Sig.hat <- cov(rets)
    w.opt <- solve(Sig.hat, mu.hat)
    retval <- list(mu = mu.hat, sig = Sig.hat, w = w.opt)
```

```
    return(retval)
}
# make multivariate pop. & sample w/ given
# zeta.star
gen.pop <- function(n, p, zeta.s = 0) {
    true.mu <- matrix(rnorm(p), ncol = p)
    # generate an SPD population covariance. a hack.
    xser <- matrix(rnorm(p * (p + 100)), ncol = p)
    true.Sig <- t(xser) %*% xser
    pre.sr <- sqrt(true.mu %*% solve(true.Sig, t(true.mu)))
    # scale down the sample mean to match the zeta.s
    true.mu <- (zeta.s/pre.sr[1]) * true.mu
    X <- mvrnorm(n = n, mu = true.mu, Sigma = true.Sig)
    retval = list(X = X, mu = true.mu, sig = true.Sig,
        SNR = zeta.s)
    return(retval)
}
# a single simulation
sample.haircut <- function(n, p, ...) {
    popX <- gen.pop(n, p, ...)
    smeas <- simple.marko(popX$X)
    # I have got to figure out how to deal with
    # vectors...
    ssnr <- (t(smeas$w) %*% t(popX$mu))/sqrt(t(smeas$w) %*%
        popX$sig %*% smeas$w)
    hcut <- 1 - (ssnr/popX$SNR)
    # for plugin estimator, estimate zeta.star
    asro <- sropt(z.s = sqrt(t(smeas$w) %*% smeas$mu),
        df1 = p, df2 = n)
    zeta.hat.s <- inference(asro, type = "KRS")  # or 'MLE', 'unbiased'
    return(c(hcut, zeta.hat.s))
}

# set everything up
set.seed(as.integer(charToRaw("496509a9-dd90-4347-aee2-1de6d3635724")))
ope <- 253
LONG.FORM <- FALSE
n.sim <- if (LONG.FORM) 2048 else 512
n.stok <- if (LONG.FORM) 8 else 6
n.yr <- 4
n.obs <- ceiling(ope * n.yr)
zeta.s <- 1.2/sqrt(ope)  # optimal SNR, in daily units

# run some experiments
system.time(experiments <- replicate(n.sim, sample.haircut(n.obs,
    n.stok, zeta.s)))
```

```
##    user  system elapsed
##   1.080   0.036   1.080
```

```
hcuts <- experiments[1, ]
print(summary(hcuts))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.01    0.15    0.24    0.29    0.39    1.42
```

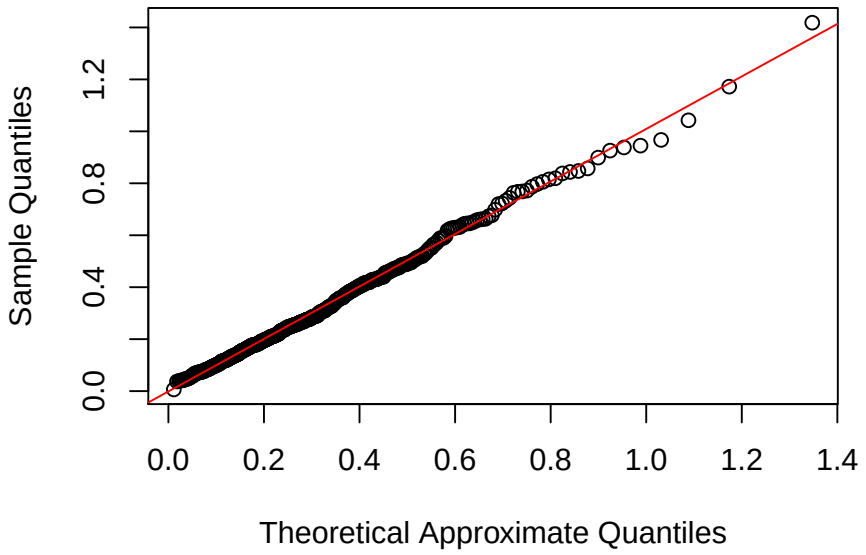

Figure 2: Q-Q plot of 512 simulated haircut values versus the approximation given by Equation 3 is shown.

```
# haircut approximation in the equation above
qhcut <- function(p, df1, df2, zeta.s, lower.tail = TRUE) {
    1 - sin(atan((1/sqrt(df1 - 1)) * qt(p, df = df1 -
        1, ncp = sqrt(df2) * zeta.s, lower.tail = !lower.tail)))
}
# if you wanted to look at how bad the plug-in
# estimator is, then uncomment the following (you
# are warned): zeta.hat.s <- experiments[2,];
# qqplot(qhcut(ppoints(length(hcuts)),n.stok,n.obs,zeta.hat.s),hcuts,
# xlab = 'Theoretical Approximate Quantiles',
# ylab = 'Sample Quantiles');
# qqline(hcuts,datax=FALSE,distribution =
# function(p) { qhcut(p,n.stok,n.obs,zeta.hat.s)
# }, col=2)

# qqplot;
qqplot(qhcut(ppoints(length(hcuts)), n.stok, n.obs,
    zeta.s), hcuts, xlab = "Theoretical Approximate Quantiles",
    ylab = "Sample Quantiles")
qqline(hcuts, datax = FALSE, distribution = function(p) {
    qhcut(p, n.stok, n.obs, zeta.s)
}, col = 2)
```

I check the quality of the approximation given in Equation 3 by a Q-Q plot

in Figure 2. For the case where $n = 1012$ (4 years of daily observations), $p = 6$ and $\zeta_* = 1.2\text{yr}^{-1/2}$, the t-approximation is very good indeed.

The median value of the haircut is on the order of 24%, meaning that the median population SNR of the sample portfolios is around $0.91\text{yr}^{-1/2}$. The maximum value of the haircut over the 512 simulations, however is 1.42, which is larger than one; this happens if and only if the sample portfolio has negative expected return: $\hat{\boldsymbol{w}}_*{}^\top \boldsymbol{\mu} < 0$. In this case the Markowitz portfolio is actually *destroying value* because of modeling error: the mean return of the selected portfolio is negative, even though positive mean is achievable.

The approximation in Equation 3 involves the unknown population parameters $\boldsymbol{\mu}$ and Σ, but does not make use of the observed quantities $\hat{\boldsymbol{\mu}}$ and $\hat{\Sigma}$. It seems mostly of theoretical interest, perhaps for producing prediction intervals on h when planning a trading strategy (*i.e.*, balancing n and p). A more practical problem is that of estimating confidence intervals on $\hat{\boldsymbol{w}}^\top \boldsymbol{\mu} / \sqrt{\hat{\boldsymbol{w}}^\top \Sigma^{-1} \hat{\boldsymbol{w}}}$ having observed $\hat{\boldsymbol{\mu}}$ and $\hat{\Sigma}$. In this case one *cannot* simply plug-in some estimate of ζ_* computed from $\hat{\zeta}_*$ (via MLE, KRS, *etc.*) into Equation 3. The reason is that the error in the approximation of ζ_* is not independent of the modeling error that causes the haircut.

3.2 Approximating Overfit

A high-level sketch of quant work is as follows: construct a trading system with some free parameters, θ, backtest the strategy for $\theta_1, \theta_2, \ldots, \theta_m$, then pick the θ_i that maximizes the Sharpe ratio in backtesting. 'Overfit bias' (variously known as 'datamining bias,' 'garbatrage,' 'backtest arb,' *etc.*) is the upward bias in one's estimate of the true signal-to-noise of the strategy parametrized by θ_{i^*} due to one using the same data to select the strategy and estimate it's performance. [1][Chapter 6]

As an example, consider a basic Moving Average Crossover strategy. Here θ is a vector of 2 window lengths. One longs the market exactly when one moving average exceeds the other, and shorts otherwise. One performs a brute force search of all allowable window sizes. Before deciding to deploy money into the MAC strategy parametrized by θ_{i^*}, one has to estimate its profitability.

There is a way to roughly estimate overfit bias by viewing the problem as a portfolio problem, and performing inference on the optimal Sharpe ratio. To do this, suppose that $\boldsymbol{x}_i$ is the n-vector of backtested returns associated with θ_i. (I will assume that all the backtests are over the same period of time.) Then approximately embed the backtested returns vectors in the subset of a p-dimensional subspace. That is, by a process like PCA, make the approximation:

$$\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\} \approx \mathcal{K} \subset \mathcal{L} =_{\text{df}} \{\mathsf{Y}\hat{\boldsymbol{w}} \mid \hat{\boldsymbol{w}} \in \mathbb{R}^p\}$$

Abusing notation, let $\hat{\zeta}(\theta)$ be the sample Sharpe ratio associated with parameters θ, and also let $\hat{\zeta}(\boldsymbol{x})$ be the Sharpe ratio associated with the vector of returns $\boldsymbol{x}$. Then make the approximation

$$\hat{\zeta}(\theta_*) =_{\text{df}} \max_{\theta_1,\ldots,\theta_m} \hat{\zeta}(\theta_i) \approx \max_{\boldsymbol{x}\in\mathcal{K}} \hat{\zeta}(\boldsymbol{x}) \leq \max_{\boldsymbol{x}\in\mathcal{L}} \hat{\zeta}(\boldsymbol{x}) = \hat{\zeta}_*.$$

This is a conservative approximation: the true maximum over $\mathcal{L}$ is presumably much larger than $\hat{\zeta}(\theta_*)$. One can then use $\hat{\zeta}(\theta_*)$ as $\hat{\zeta}_*$ over a set of p assets,

perform inference on ζ_*, which, by a series of approximations as above, is an approximate upper bound on $\zeta(\theta_*)$.

This approximate attack on overfitting will work better when one has a good estimate of p, when m is relatively large and p relatively small, and when the linear approximation to the set of backtested returns is good. Moreover, the definition of $\mathcal{L}$ explicitly allows shorting, whereas the backtested returns vectors $\boldsymbol{x}_i$ may lack the symmetry about zero to make this a good approximation. By way of illustration, consider the case where the trading system is set up such that different θ produce minor variants on a clearly losing strategy: in this case we might have $\hat{\zeta}(\theta_*) < 0$, which cannot hold for $\hat{\zeta}_*$.

One can estimate p via Monte Carlo simulations, by actually performing PCA, or via the 'SWAG' method. Surprisingly, often one's intuitive estimate of the true 'degrees of freedom' in a trading system is reasonably good.

```
require(TTR)
# brute force search two window MAC
brute.force <- function(lrets, rrets = exp(lrets) -
    1, win1, win2 = win1) {
    mtms <- c(1, exp(cumsum(lrets)))  # prepend a 1.
    # do all the SMAs;
    SMA1 <- sapply(win1, function(n) {
        SMA(mtms, n = n)
    })
    symmetric <- missing(win2)
    if (!symmetric)
        SMA2 <- sapply(win2, function(n) {
            SMA(mtms, n = n)
        })

    mwin <- max(c(win1, win2))
    zeds <- matrix(NaN, nrow = length(win1), ncol = length(win2))
    upb <- if (symmetric)
        length(win1) - 1 else length(win1)
    # 2FIX: vectorize this!
    for (iidx in 1:upb) {
        SM1 <- SMA1[, iidx]
        lob <- if (symmetric)
            iidx + 1 else 1
        for (jidx in lob:length(win2)) {
            SM2 <- if (symmetric)
                SMA1[, jidx] else SMA2[, jidx]
            trades <- sign(SM1 - SM2)
            dum.bt <- trades[mwin:(length(trades) -
                1)] * rrets[mwin:length(rrets)]  # braindead backtest.
            mysr <- as.sr(dum.bt)
            zeds[iidx, jidx] <- mysr$sr
            # abuse symmetry of arithmetic returns
            if (symmetric)
                zeds[jidx, iidx] <- -zeds[iidx, jidx]
        }
    }
    retv <- max(zeds, na.rm = TRUE)
    return(retv)
```

```
}
# simulate one.
sim.one <- function(nbt, win1, ...) {
    lrets <- rnorm(nbt + max(win1), sd = 0.01)
    retv <- brute.force(lrets, win1 = win1, ...)
    return(retv)
}
# set everything up
set.seed(as.integer(charToRaw("e23769f4-94f8-4c36-bca1-28c48c49b4fb")))
ope <- 253
n.yr <- 4
n.obs <- ceiling(ope * n.yr)
n.sim <- if (LONG.FORM) 2048 else 1024
win1 <- if (LONG.FORM) c(2, 4, 8, 16, 32, 64, 128,
    256) else c(4, 16, 64, 256)

# run them
system.time(max.zeds <- replicate(n.sim, sim.one(n.obs,
    win1)))

##    user  system elapsed
##     3.8     0.0     3.8

# qqplot;
qqplot(qsropt(ppoints(length(max.zeds)), df1 = 2, df2 = n.obs),
    max.zeds, xlab = "Theoretical Approximate Quantiles",
    ylab = "Sample Quantiles")
qqline(max.zeds, datax = FALSE, distribution = function(p) {
    qsropt(p, df1 = 2, df2 = n.obs)
}, col = 2)
```

Here I illustrate the quality of the approximation for the two-window simple MAC strategy. I generate log returns which are homoskedastic, driftless, and have zero autocorrelation. In this case, *every* MAC strategy has zero expected return (ignoring trading costs). In spite of this deficiency in the market, I find the best combination of window sizes by looking at 4 years of daily data. By selecting the combination of windows with the highest Sharpe ratio, then using that maximal value as an estimate of the selected model's true signal-noise-ratio, I have subjected myself to overfit bias. I repeat this experiment 1024 times, then Q-Q plot the maximal Sharpe ratio values over those experiments versus an optimal Sharpe ratio distribution assuming $p = 2$ in Figure 3. The fit is reasonable[1] except in the case where the maximal in-sample Sharpe ratio is very low (recall that it can be negative for this brute-force search, whereas the optimal Sharpe ratio distribution does not produce negative values). This case is unlikely to lead to a trading catastrophe, however.

It behooves me to replicate the above experiment 'under the alternative,' *e.g.*, when the market has autocorrelated returns, to see if the approximation holds up when $\zeta_* > 0$. I leave this for future iterations. Instead, I apply the $p = 2$ approximation to the brute-force MAC overfit on `SPY`.

[1] And much better when the overfitting is more aggressive, which takes more processing time to simulate.

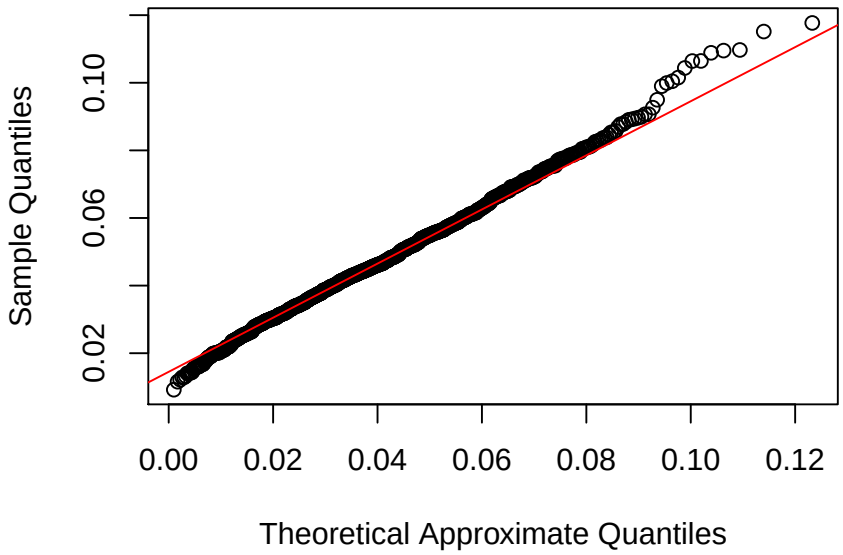

Figure 3: Q-Q plot of 1024 achieved optimal Sharpe ratio values from brute force search over both windows of a Moving Average Crossover under the null of driftless log returns with zero autocorrelation versus the approximation by a 2-parameter optimal Sharpe ratio distribution is shown.

```
# is MAC on SPY significant?
SPY.lret <- get.rets(c("SPY"), from = "2003-01-01",
    to = "2013-01-01")
# oops! there might be NAs in there!
mysr <- as.sr(SPY.lret, na.rm = TRUE)  # just to get the ope
print(mysr)

##     SR/sqrt(yr) Std. Error t value Pr(>t)
## SPY        0.31       0.32    0.99   0.16

# get rid of NAs
SPY.lret[is.na(SPY.lret)] <- 0
# try a whole lot of windows:
win1 <- seq(4, 204, by = 10)
zeds <- brute.force(SPY.lret, win1 = win1)
SPY.MAC.asro <- sropt(z.s = zeds, df1 = 2, df2 = length(SPY.lret) -
    max(win1), ope = mysr$ope)
print(SPY.MAC.asro)

##        SR/sqrt(yr) T^2 value Pr(>T^2)
## Sharpe        0.46         2     0.37

print(inference(SPY.MAC.asro, type = "KRS"))

## [1] 0.33
```

The Sharpe ratio for `SPY` over this period is $0.31\text{yr}^{-1/2}$. The optimal Sharpe ratio for the tested MAC strategies is $0.46\text{yr}^{-1/2}$. The KRS estimate (using the $p = 2$ approximation) for the upper bound on signal-to-noise of the optimal MAC strategy is only $0.33\text{yr}^{-1/2}$. [5] This leaves little room for excitement about MAC strategies on `SPY`.

4 Distribution Functions

There are `dpqr` functions for the Sharpe ratio distribution, as well as the 'optimal Sharpe ratio' distribution. These are merely rescaled non-central t and F distributions, provided for convenience, and for testing purposes. See the help for `dsr` and `dsropt` for more details.

5 Hypothesis Tests

The function `sr_equality_test`, implements the tests of Leung *et al.* and of Wright *et al.* for testing the equality of Sharpe ratio. [9, 17] I have found that these tests (or my implementation thereof!) apparently reject the null 'for the wrong reason'.

Here I confirm that the tests give approximately uniform p-values under the null in three situations: when stock returns are independent Gaussians, when they are independent and t-distributed, and when they are Gaussian and correlated. Visual confirmation is via Figure 4 through Figure 6, showing Q-Q plots against uniformity of the putative p-values in Monte Carlo simulations with data drawn from the null.

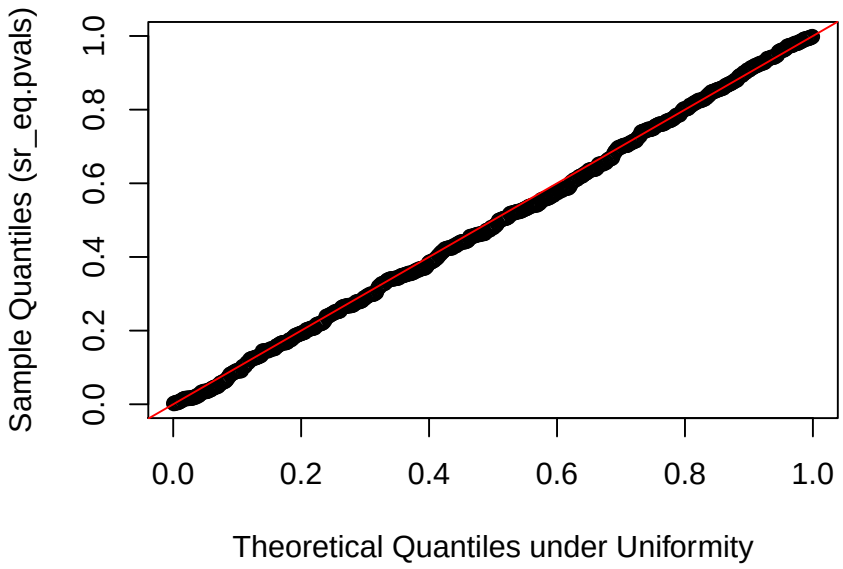

Figure 4: P-values from `sr_equality_test` on 512 multiple realizations of independent, normally distributed returns series are Q-Q plotted against uniformity.

```
# function for Q-Q plot against uniformity
qqunif <- function(x, xlab = "Theoretical Quantiles under Uniformity",
    ylab = NULL, ...) {
    if (is.null(ylab))
        ylab = paste("Sample Quantiles (", deparse(substitute(x)),
            ")", sep = "")
    qqplot(qunif(ppoints(length(x))), x, xlab = xlab,
        ylab = ylab, ...)
    abline(0, 1, col = "red")
}

# under normality.
n.ro <- if (LONG.FORM) 253 * 4 else 253 * 2
n.co <- if (LONG.FORM) 20 else 4
n.sim <- if (LONG.FORM) 1024 else 512
set.seed(as.integer(charToRaw("e3709e11-37e0-449b-bcdf-9271fb1666e5")))
afoo <- replicate(n.sim, sr_equality_test(matrix(rnorm(n.ro *
    n.co), ncol = n.co), type = "F"))
sr_eq.pvals <- unlist(afoo["p.value", ])
qqunif(sr_eq.pvals)
```

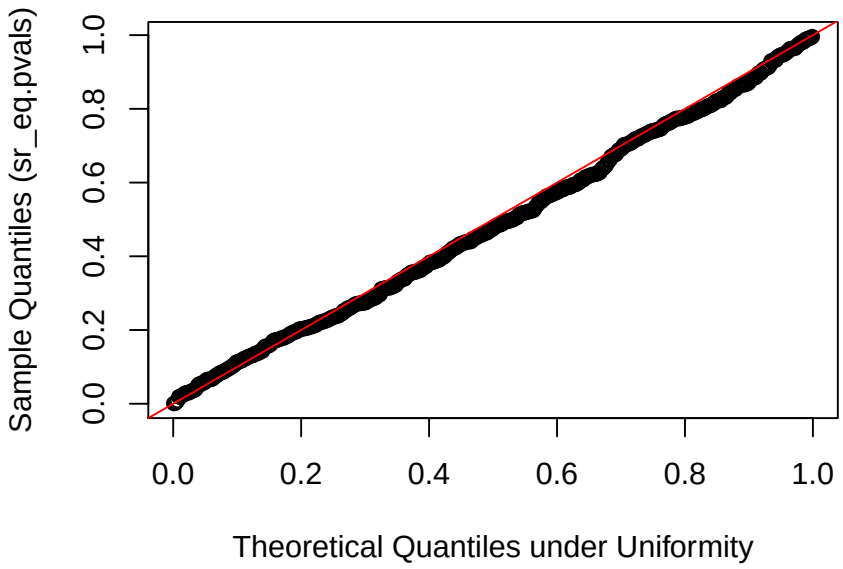

Figure 5: P-values from `sr_equality_test` on 512 multiple realizations of independent, t-distributed returns series are Q-Q plotted against uniformity.

```
# try heteroskedasticity?
set.seed(as.integer(charToRaw("81c97c5e-7b21-4672-8140-bd01d98d1d2e")))
afoo <- replicate(n.sim, sr_equality_test(matrix(rt(n.ro *
    n.co, df = 4), ncol = n.co), type = "F"))
sr_eq.pvals <- unlist(afoo["p.value", ])
qqunif(sr_eq.pvals)
```

```
# try correlated returns
n.fact <- max(2, n.co - 5)
gen.ret <- function(n1, n2, f = max(2, n2 - 2), fuzz = 0.1) {
    A <- matrix(rnorm(n1 * f), nrow = n1)
    B <- matrix(rnorm(f * n2), nrow = f)
    C <- sqrt(1 - fuzz^2) * A %*% B + fuzz * matrix(rnorm(n1 *
        n2), nrow = n1)
}
set.seed(as.integer(charToRaw("e4d61c2c-efb3-4cba-9a6e-5f5276ce2ded")))
afoo <- replicate(n.sim, sr_equality_test(gen.ret(n.ro,
    n.co, n.fact), type = "F"))
sr_eq.pvals <- unlist(afoo["p.value", ])
qqunif(sr_eq.pvals)
```

As a followup, I consider the case where the returns are those of several randomly generated market-timing strategies which are always fully invested

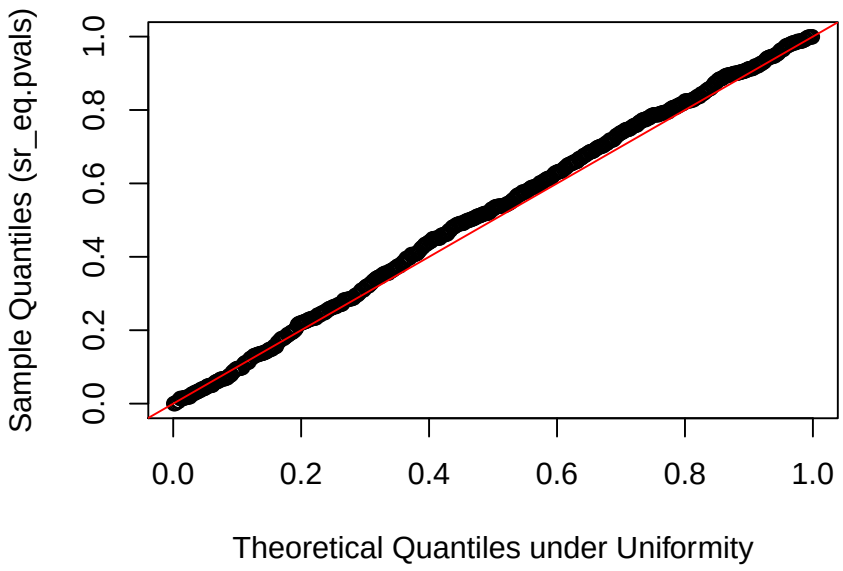

Figure 6: P-values from `sr_equality_test` on 512 multiple realizations of correlated, normally-distributed returns series are Q-Q plotted against uniformity.

long or short in the market, with equal probability. I look at 128 simulations of 50 random market timing strategies rebalancing weekly on a 10 year history of `SPY`. The 'p-values' from `sr_equality_test` applied to the returns are Q-Q plotted against uniformity in Figure 7. The type I rate of this test is far larger than the nominal rate: it is rejecting for 'uninteresting reasons'. I suspect this test breaks down because of small sample size or small 'aspect ratio' (ratio of weeks to strategies in this example). In summary, one should exercise caution when interpreting the results of the Sharpe equality test: it would be worthwhile to compare the results against a 'Monte Carlo null' as illustrated here.

```
n.co <- if (LONG.FORM) 100 else 50
n.sim <- if (LONG.FORM) 1024 else 128
SPY.lret <- get.rets(c("SPY"), from = "2003-01-01",
    to = "2013-01-01")
SPY.wk.rret <- my.periodReturn(lr2mtm(SPY.lret), period = "weekly",
    type = "arithmetic")
gen.tim <- function(n2) {
    mkt.timing.signal <- sign(rnorm(n2 * length(SPY.wk.rret)))
    mkt.ret <- matrix(rep(SPY.wk.rret, n2) * mkt.timing.signal,
        ncol = n2)
}
set.seed(as.integer(charToRaw("447cfe85-b612-4b14-bd01-404e6e99aca4")))
system.time(afoo <- replicate(n.sim, sr_equality_test(gen.tim(n.co),
    type = "F")))

##    user  system elapsed
##   1.216   0.012   1.198

sr_eq.pvals <- unlist(afoo["p.value", ])
qqunif(sr_eq.pvals)
```

References

[1] Aronson, D.˜R. *Evidence-Based Technical Analysis: Applying the Scientific Method and Statistical Inference to Trading Signals*. Wiley, Hoboken, NJ, Nov. 2006.

[2] Fisher, R.˜A. Applications of "Student's" distribution. *Metron 5* (1925), 90–104.

[3] Gosset, W.˜S. The probable error of a mean. *Biometrika 6*, 1 (March 1908), 1–25. Originally published under the pseudonym "Student".

[4] Johnson, N.˜L., and Welch, B.˜L. Applications of the non-central t-distribution. *Biometrika 31*, 3-4 (Mar. 1940), 362–389.

[5] Kubokawa, T., Robert, C.˜P., and Saleh, A. K. M.˜E. Estimation of noncentrality parameters. *Canadian Journal of Statistics 21*, 1 (1993), 45–57.

[6] Lecoutre, B. Another look at confidence intervals for the noncentral T distribution. *Journal of Modern Applied Statistical Methods 6*, 1 (2007), 107–116.

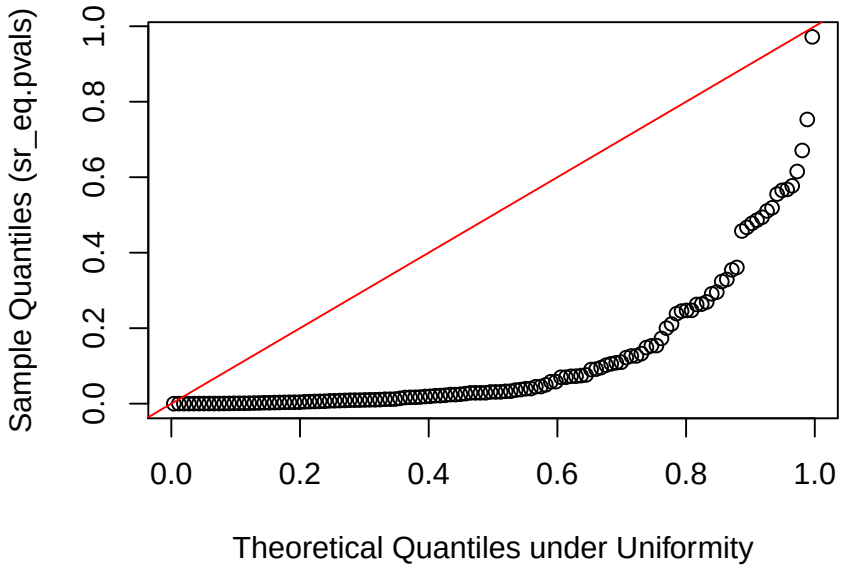

Figure 7: P-values from `sr_equality_test` on 128 multiple realizations of 50 market timing strategies' returns series are Q-Q plotted against uniformity. Nominal coverage is not maintained: the test is far too liberal in this case.

[7] Ledoit, O., and Wolf, M. Robust performance hypothesis testing with the Sharpe ratio. *Journal of Empirical Finance 15*, 5 (Dec 2008), 850–859.

[8] Lehr, R. Sixteen S-squared over D-squared: a relation for crude sample size estimates. *Statistics in Medicine 11*, 8 (1992), 1099–102.

[9] Leung, P.-L., and Wong, W.-K. On testing the equality of multiple Sharpe ratios, with application on the evaluation of iShares. *Journal of Risk 10*, 3 (2008), 15–30.

[10] Lo, A.˜W. The Statistics of Sharpe Ratios. *Financial Analysts Journal 58*, 4 (July/August 2002).

[11] Lumley, T., Diehr, P., Emerson, S., and Chen, L. The importance of the normality assumption in large public health data sets. *Annu Rev Public Health 23* (2002), 151–69+.

[12] Miller, R.˜E., and Gehr, A.˜K. Sample size bias and Sharpe's performance measure: A note. *Journal of Financial and Quantitative Analysis 13*, 05 (1978), 943–946.

[13] Opdyke, J.˜D. Comparing sharpe ratios: So where are the p-values? *Journal of Asset Management 8*, 5 (2007).

[14] Sharpe, W.˜F. Mutual fund performance. *Journal of Business 39* (1965), 119.

[15] Spruill, M.˜C. Computation of the maximum likelihood estimate of a noncentrality parameter. *Journal of Multivariate Analysis 18*, 2 (1986), 216–224.

[16] van Belle, G. *Statistical Rules of Thumb (Wiley Series in Probability and Statistics)*, 1˜ed. Wiley-Interscience, Mar. 2002.

[17] Wright, J.˜A., Yam, S. C.˜P., and Yung, S.˜P. A note on a test for the equality of multiple Sharpe ratios and its application on the evaluation of iShares. Tech. rep., 2012. to appear.