Vignette for Risk and Performance Estimators Influence Functions Package

Anthony Christidis, R. Douglas Martin, Shengyu Zhang

June 30, 2019

Abstract

The InfluenceFunctions package (`RPEIF`) computes the influence functions for risk and performance estimators based on the returns of individual assets or groups of assets, the latter for example in the context of an investment portfolio.. This initial release of the package treats five risk estimators and seven performance estimators, including amont others, the standard deviation, value at risk, expected shortfall, Sharpe ratio, Sortino ratio estimators. The influence function of a risk or performance estimator provides a basis for computing an estimate of the finite-sample standard error when returns are serially uncorrelated, and also when the returns are serially correlated, as discussed in Chen and Martin [2018]. This `RPEIF` package is used in the package `RPESE` to compute the time series of influence function transformed returns needed to compute the standard errors of risk and performance estimators

1 Influence Functions Theoretical Background

The Influence Functions package (`RPEIF`) makes use of the definition and theory of influence functions in robust statistics, as introduced by Hampel (1974), further treated in Hampel et al. (1986). Recent use of influence functions for the analysis of parametric and nonparametric expected shortfall (ES) is discussed in Martin and Zhang (2018), and use of influence functions for computing standard errors of risk and perfomance estimators with serially correlated returns is discussed in Chen and Martin (2019). Here we provide the definition and basic properties of influence functions, with a view toward their use for understanding the influence of outiers on risk and performance estimators, and for computing standard errors of such estimators for both uncorrelated and serially correlated returns.

1.1 Risk and Performance Estimator Functional Representations

The large-sample value (as sample size n tends to infinity) of a risk or performance estimator may be represented as a functional $T = T(F)$ of the marginal distribution function F of a time series $r_1, r_2, \ldots, r_n$ of returns.[1] For example the functional for the mean (expected value) is

[1] The term functional refers to a function whose domain is an infinite dimensional space, e.g., the space of distribution functions.

$$\mu(F) = \int r dF(r) \tag{1}$$

and the functional for the standard deviation (returns volatility) is

$$\sigma(F) = \left[\int (r - \mu(F))^2 dF(r) \right]^{\frac{1}{2}}. \tag{2}$$

Given a functional representation $T(F)$ of an estimator, a finite-sample *non-parametric* estimator T_n is easily obtained by replacing the unknown distribution F by the empirical distribution F_n that has a jump of height $1/n$ at each of the observed returns values $r_1, r_2, \ldots, r_n$:

$$T_n = T(F_n) = T(r_1, r_2, \ldots, r_n). \tag{3}$$

.

For example, the finite-sample non-parametric estimators of the mean and standard deviation are the sample mean and sample standard deviation, respectively:

$$\hat{\mu}_n = \frac{1}{n} \sum_{t=1}^{n} r_t \qquad \hat{\sigma}_n = \left[\frac{1}{n} \sum_{t=1}^{n} (r_t - \hat{\mu}_n) \right]^{\frac{1}{2}}. \tag{4}$$

We note that one can also derive parametric estimators from parametric functional representation obtained by replacing F by F_θ, where θ is the parameter vector for a parametric distribution function. In this case one obtains the finite-sample estimator by replacing the unknown parameter by its estimator, typically the maximum-likelihood estimator (MLE). See for example, Martin and Zhang (2018) for a treatment of parametric and non-parametric ES estimators for normal and t-distributions. However, this first version of the `RPEIF` package only deals with non-parametric risk and performance estimators.

1.2 Estimator Influence Function Definition

Influence Functions are based on the use of the following mixture distribution perturbation of a fixed target distribution $F(x)$:

$$F_\gamma(x) = (1 - \gamma)F(x) + \gamma \delta_r(x),\ 0 \leq \gamma < 1/2 \tag{5}$$

where $\delta_r(x)$ is a point mass discrete distribution function with a jump of height one located at value r . The influence function of an estimator with functional form $T(F)$ is defined as:

$$IF(r; T, F) = \lim_{\gamma \to 0} \frac{T(F_\gamma) - T(F)}{\gamma} = \frac{d}{d\gamma} T(F_\gamma)|_{\gamma=0} \tag{6}$$

The influence function is a special directional derivative (i.e., a Gateaux derivative) of the functional $T(F)$ in the direction of a point mass distributions δ_r, evaluated at F.

It is straightforward, and more or less tedious, to derive formulas for the influence functions of risk and performance estimators. For example, the influence function of the sample mean is:

$$IF(r;\mu;F) = r - \mu \tag{7}$$

where $\mu = \mu(F)$ depends on the underlying returns marginal distributon F. The above influence function has the property that its expected value is zero, which is a reflection of the general property than an influence function has zero expected value:

$$E\left[IF(r;T,F)\right] = 0. \tag{8}$$

1.3 A Key Influence Function Property

A key influence function property is that for well behaved estimator functionals, the finite-sample estimator $T_n = T(F_n) = T(r_1, r_2, \ldots, r_n)$ can be expressed as the following linear combination of influence-function transformed returns:

$$T_n - T(F) = \frac{1}{n}\sum_{t=1}^{n} IF(r_t;T,F) + remainder \tag{9}$$

where the $IF(r_t;T,F)$ are influence-functon (IF) transformed returns, and the remainder goes to zero in a probablistic sense as $n \to \infty$. Thus the finite sample variance of T_n is approximately given by

$$Var\left(T_n\right) = Var\left[\frac{1}{n}\sum_{t=1}^{n} IF(r_t;T,F)\right] \tag{10}$$

and in the special case where the returns r_t are i.i.d., the IF-transformed returns are i.i.d., and the variance of T_n reduces to

$$Var\left(T_n\right) = \frac{1}{n}E[IF^2(r_1;T,F)]. \tag{11}$$

and the expectation on the right-hand side can be evaluated empirically as the sample mean of the $IF(r_t;T,F), t = 1, 2, \cdots, n$.

However, when the $r_t, t = 1, 2, \ldots, n$ are serially correlated, the IF-transformed returns time series $IF(r_t;T,F)$ will generally have serial correlation that needs to be accounted for in calculating the variance on the the right-hand-side of (10). Spectral analysis theory, extensively used in science and engineering, shows that the variance of the sum of the values of a serially correlated stationary time series is given by the spectral density of the time series at zero frequency. Thus the problem of estimating the variance (10), with possibly serially correlated returns, reduces to the problem of estimating the spectral density at zero frequency at zero frequncy of the the time series $IF(r_t;T,F)/n$. Chen and Martin [2018] show how to do this by a polynomial gneralized linear model (GLM) fitting method, with elastic net (EN) regularization, that works well when the returns are serially correlated, as well as when they are uncorrelated. Their methodology, which we refer to as the CM method, is implemented in the Estimator Standard Error (`ESE`) package, which in turn makes fundamental use of the `RPEIF` package we are discussing here.

2 RPEIF Package Estimators and their Influence Functions

Table 1 lists the symbolic or acronymn names and descriptions of the risk and performance estimators for which the `RPEIF` package computes influence functions. Each estimator has a functional representation, a sample based estimator as a function of a time series of asset returns, and an influence function formula. We illustrate this below for the case of the standard deviation and Sharpe ratio estimators.

Name	Estimator Description
μ	Mean
σ	Standard deviation
SSD	Semi-standard deviation
LPM	Lower partial moment of order 1 or 2 with threshold c
ES	Expected shortfall with tail probability α
VaR	Value-at-risk with tail probability α
SR	Sharpe ratio
SoR	Sortino ratio with threshold the mean or a constant c
ESratio	Mean excess return to ES ratio with tail probability α
VaRratio	Mean excess return to VaR ratio with tail probability α
Omega	Omega ratio with threshold c
RachR	Rachev ratio with lower upper tail probabilities α and β

Table 1: Estimator Names and Descriptions

Standard Deviation (SD)

The functional representation $\sigma(F)$ of the sample standard deviation estimator is given by (2), and the formula for the influence function of the sample standard deviation estimator is

$$IF(r;\sigma;F) = (2\sigma)^{-1}((r-\mu)^2 - \sigma^2) \tag{12}$$

where $\mu = \mu(F)$ and $\sigma = \sigma(F)$. The sample standard deviation estimator is given in (4).

Sharpe Ratio (SR)

The formula for the functional reprsentation of the Sharpe ratio is

$$SR(F) = \frac{\mu(F) - r_f}{\sigma(F)} = \frac{\mu_e(F)}{\sigma(F)} \tag{13}$$

where $\mu_e(F)$ is the mean excess return over the risk-free rate r_f, and the SR influence function formula is:

$$IF(r; SR; F) = -\frac{\mu_e}{2\sigma^3}(r-\mu)^2 + \frac{1}{\sigma}(r-\mu) + \frac{\mu_e}{2\sigma}\,. \tag{14}$$

The sample estimator of the Sharpe ratio is:

$$SR_n = \frac{\hat{\mu}_n - r_f}{\hat{\sigma}_n} \tag{15}$$

where $\hat{\mu}_n$ and $\hat{\sigma}_n$ are the sample mean and sample standard deviation.

The functional forms, and derivations of the influence function formulas for all the estimators in Table 1 may be found in Zhang et al. (2019).. For the convenience of interested readers, the influence functions of all the estimators in Table 1 are provided in Appendix A.

Nuisance Parameters

We note that the influence functon formulas above, as well as in Appendix A, contain one or more *nuisance* parameters that need to be specified in order to compute influence function values for various values of a return r. For example, the IF of the sample standard deviation depends on the two nuisance parameters $\mu = \mu(F)$ and $\sigma = \sigma(F)$, and the Sharpe ratio IF depends on the three nuisance parameters $\mu = \mu(F)$, $\sigma = \sigma(F)$, and $SR = SR(F)$.

3 Using the RPEIF Package to Evaluate Influence Functions and Compute Influence-Function Transformed Returns

Load and install and load the `RPEIF` package with the code:

```
library(devtools)
install_github("AnthonyChristidis/RPEIF")
```

The `RPEIF` package computes the influence functions time series of the returns for the risk and performance measures in `RPESE`, which in turn is used in `PerformanceAnalytics` if the user wishes to return the standard errors for the risk and performance measures. The `RPEIF` package is therefore required by the `RPESE` package.

To demonstrate the usage of the `RPEIF` package, we will make use of the `edhec` data set from the `PerformanceAnalytics` package. This data set contains hedge fund returns from January, 1997 to August, 2008. You can load that data set, and confirm that it is an `xts` time series object with the following R code.

```
library(RPEIF)
data(edhec, package = 'PerformanceAnalytics')
class(edhec)


## [1] "xts" "zoo"
```

Use of the code line

```
colnames(edhec, package = 'PerformanceAnalytics')
```

will reveal to you that the hedge fund style names are too long to display well in plots. So use the following code to replace those long names with shorter names as follows, and view the first six values of the CA fund.

```
colnames(edhec) = c("CA", "CTAG", "DIS", "EM","EMN", "ED", "FIA",
                    "GM", "LS", "MA", "RV", "SS", "FoF")
colnames(edhec)

##  [1] "CA"   "CTAG" "DIS"  "EM"   "EMN"  "ED"   "FIA"  "GM"   "LS"
## [10] "MA"   "RV"   "SS"   "FoF"

head(edhec[,"CA"])

## [1] 0.0119 0.0123 0.0078 0.0086 0.0156 0.0212
```

To see what functions are contained in the `RPEIF` package, use the code line:

```
library(RPEIF)
ls("package:RPEIF")
```

The list of functions you see as a result of the above code will include the functions shown in the second column of 2. We use the term `IF.xxxx` as a generic for any one of the functions in the second column. The third column lists the names of nuisance parameters that need to be specified one way or another to compute the values of an influence function at various values of the influence function argument. The fourth column contains normal distribution "typical" values for the nuisance parameters.

You can get help on any of the functions IF.xxxx with a commond like the following:

```
# Help file for the IF function
help(IF.SD)
```

Estimator	IF Function	Nuisance Parameters	Normal Dist. Nuisance Pars
μ	`IF.mean`	μ	.005
σ	`IF.SD`	μ, σ	.005, .07
SSD	`IF.SSD`	μ, SSD, $SMEAN$	.005, .0495, -0.0279
LPM1	`IF.LPM`	$LPM1_c$	.0255
LPM2	`IF.LPM`	$LPM2_c$	.00218
ES	`IF.ES`	q_α, ES_α	-.0847, .273
VaR	`IF.VaR`	q_α, $f(q_\alpha)$	-.0847, 2.507
SR	`IF.SR`	μ_e, σ, SR,	.01, .05, .20
SoR_c	`IF.SoR`	μ, $LPM2_c$, SoR_c	.01, .00898, .3337
SoR_μ	`IF.SoR`	μ, SSD, $SMEAN$, SoR_μ	.01, .0354, -.0199, .2929
ESratio	`IF.ESratio`	μ, q_α, ES_α, $ESratio$	.01, -.0541, .0777, .129
VaRratio	`IF.VaRratio`	μ, q_α, $f(q_\alpha)$, $VaRratio$	.01, -.0541, 3.99, .185
RachR	`IF.RachR`	q_α, ES_α, $q_{1-\beta}$, EG_β, $RachR$	-.0541, .0777, .0741, 0.0977, 1.257
Omega	`IF.Omega`	$LPM1_c$, $UPM1_c$, Ω	.0153, .0253, 1.652

Table 2: IF R Code Functions and Corresponding Nuisance Parameters

The R `IF.xxxx` functions are used for two distinct purposes. The first is to evaluate an estimator influence function at a set of argument values, and plot them to display the shape of the influence function. This allows the user to expolore the different shapes of the influence functions of different estimators. The second and primary purpose of the functions is to compute influence-function transformed time series of returns, as a first step in the overall method of computing standard errors for risk and perfromance estimators.

The arguments of the R functions in Table (2) are all the same, except for the nuisance parameters which are estimator specific. For example, the arguments of `IF.SD` and `IF.SR` are as follows.

```
args(IF.SD)

## function (returns = NULL, evalShape = FALSE, retVals = NULL,
##     nuisPars = NULL, k = 4, IFplot = FALSE, IFprint = TRUE, compile = TRUE,
##     prewhiten = FALSE, ar.prewhiten.order = 1, cleanOutliers = FALSE,
##     cleanMethod = c("locScaleRob", "Boudt")[1], eff = 0.99, alpha.robust = 0.05,
##     ...)
## NULL

args(IF.SR)

## function (returns = NULL, evalShape = FALSE, retVals = NULL,
```

```
##     nuisPars = NULL, k = 4, IFplot = FALSE, IFprint = TRUE, rf = 0,
##     compile = TRUE, prewhiten = FALSE, ar.prewhiten.order = 1,
##     cleanOutliers = FALSE, cleanMethod = c("locScaleRob", "Boudt")[1],
##     eff = 0.99, alpha.robust = 0.05, ...)
## NULL
```

You see that the arguments of the two functions are the same except for the nuisance parameters arguments `parsSD.IF` and `parsSR.IF`.

Note that you get no results using an `IF.xxxx` function with no arguements, for example try `IF.mean()` and see what you get. You need to either set `evalShape = T`, or supply a returns xts object for the argument `returns =` , in order for an `IF.xxxx` function to compute results.

3.1 Evaluating and Plotting Influence Functions for Shape Comparisons

In order to compute the values and plot the shapes of influence functions using the R `IF.xxxx` functions in Table 2, the values of the nuisance parameters in the third column of 2 need to be specified, and there are two basic methods of doing so. The first method is to use "typical" values for those parameters in the case of a specific returns distribution. Such valures are provided for the case of a normal returns distribution in the fourth column of the Table 2. Details concerning how these values are obtained are provided in Szhang et al. (2019). The second method is to provide a time series of returns of an asset of interest as one of the functon's arguments, and in that case the R function will estimate the values of the nuisance parameters from the time series of returns.

Evaluating and Plotting Influence Functions Using Default Nuisance Parameter Values

Using the argument `evalShape = T` and `IFplot = T` as follows, you can easily get influence function plots. For example, you get plots of the influence functions of the standard deviation and Sharpe ratio estimators using the following code.

```
par(mfrow = c(1,2),pty = "s")
outSD = IF.SD(evalShape = T, IFplot = T, IFprint = T)
outSR = IF.SR(evalShape = T, IFplot = T, IFprint = T)
par(mfrow = c(1,1))
```

The plots are shown in Figure 1.

Figure 1: Influence Functions of Standard Deviation and Sharpe Ratio ("typical" nuisance parameters values)

Evaluating and Plotting Influence Functions Using Nuisance Parameter Values Estimated from a Returns Time Series

You can also compute influence function shapes using nuisance parameter values that are estimated from a retirms time series of interest, the latter of which is specified by using the argument `returns =` of the `IF.xxxx` functions. For example, you could use the CA hedge fund time series for this purpose by using the following code, the results of which are shown in Figure 2. Note that in using the `IF.SD` and `IF.SR` functions, the `evalShape = T` is retained.

```
library(xts)  # Must have installed xts package
retCA = edhec$CA
par(mfrow = c(1,2),pty = "s")
outSD = IF.SD(returns = retCA, evalShape = T, IFplot = T, IFprint = T)
outSR = IF.SR(returns = retCA, evalShape = T, IFplot = T, IFprint = T)
par(mfrow = c(1,1))
```

Figure 2: Influence Functions of Standard Deviation and Sharpe Ratio (estimated nuisance parameters values)

3.2 Evaluating and Plotting Influence-Function Transformed Returns Time Series

The general method of estimating standard errors of risk and performance estimators, described in Chen and Martin (2019), consists of computing a good estimate of the variance of time series sample average of influence-function transformed returns shown on the right-hand-side of (10). In case you are interested in seeing what influence-function transformed returns look like for actual returns data, can easily compute such returns with the `IF.xxxx` function. For example,you can do so for SD and SR estimators for the CA hedge fund returns with the following code, whose results for the CA returns, and for the SD and SR estimator influence function transformed returns are shown in Figure 3.

```
retCA = edhec$CA
outSD = IF.SD(returns = retCA, IFplot = T, IFprint = T)
outSR = IF.SR(returns = retCA, IFplot = T, IFprint = T)
par(mfrow = c(3,1))
plot(retCA,lwd = .8, ylab = "Returns", main = "CA Hedge Fund Returns")
plot(outSD,lwd = .8, main = "IF.SD Transformed Returns")
plot(outSR,lwd = .8, main = "IF.SR Transformed Returns")
```

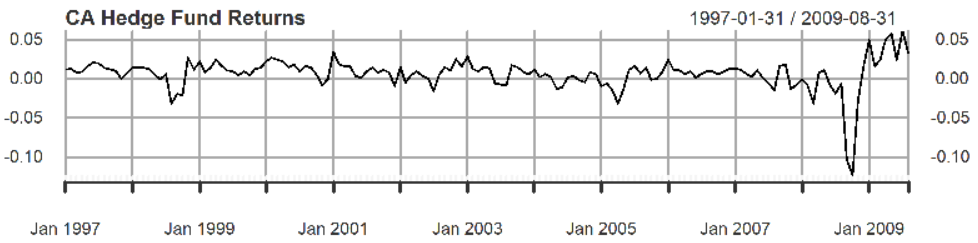

Figure 3: Standard Deviation and Sharpe Ratio Influence-Function Transformed Returns for CA Hedge Fund

4 Outlier Cleaning

A very reliable outlier cleaning method that shrinks outliers can be obtained based on a robust location M-estimator and an associated robust scale estimator $\hat{s}$. A location M-estimator is computed as a solution of the equation

$$\sum_{t=1}^{n} \psi\left(\frac{r_t - \hat{\mu}_M}{\hat{s}}\right) = 0 \tag{16}$$

where $\psi = \psi(x)$ is a suitable odd "psi" function, and $\hat{s}$ is a robust scale estimate of the residuals $\epsilon_t = r_t - \hat{\mu}_M$. For an introduction to location M-estimators and their computation, see Sections 2.3 and 2.7 of Maronna et al. (2019).

Here we use the special optimal psi function

$$\psi_{opt}(x) = SGN(x)\left(|x| + \frac{a}{\phi(x)}\right)^{+}, \text{with } a = 0.002449 \tag{17}$$

where $\phi(x)$ is the standard normal density function, and a plot of this psi function is displayed in Figure 4. The function $\psi_{opt}(x)$ has a "smooth rejection" character in that the function is continuous, and all returns with a scaled residual $(r_t - \hat{\mu}_M)/\hat{s}$ larger in magnitude than 3.568 are "rejected", in the sense that such returns have no influence on the estimate $\hat{\mu}_M$ since such terms are set to zero in (16). For further details on the function $\psi_{opt}(x)$, see Section 5.8 of Maronna et al. (2019).

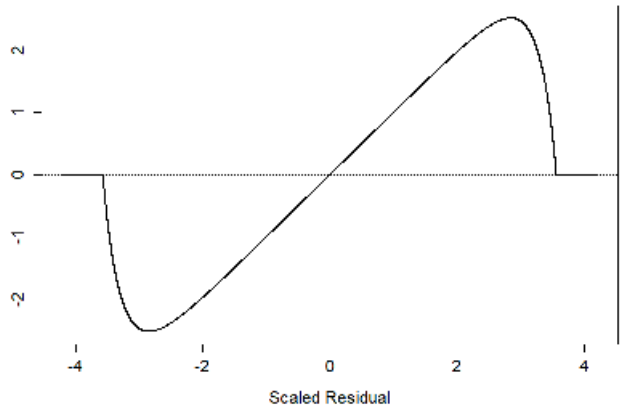

Figure 4: Optimal Bias Robust Psi Function with $\psi_{opt}(x) = 0$ for $|x| > 3.568$ and 99% Normal Distribution Efficiency.

Based on a location estimate $\hat{\mu}_M$ and associated scale estimate $\hat{s}$, it is natural to define returns outliers as those returns r_t that fall outside the interval $[\hat{\mu}_M - 3.568 \cdot \hat{s}, \hat{\mu}_M + 3.568 \cdot \hat{s}]$. Such outliers are then "cleaned" by shrinking them to the nearest boundary of that interval.

For the fixed income arbitrage (FIA) hedge fund returns, it turns out that $\hat{\mu}_M = .00640$ and $\hat{s} = .00547$ is $[-.0131, .0259]$. Correspondingly, FIA returns with values less than -.0131, or greater than .0259, are detected as outliers and shrunk accordingly. The following code results in Figure 5, which shows the sample mean IF transformed FIA returns (which are equal to FIA returns minus the very small mean of the FIA returns) in the top plot, and the outlier cleaned IF tranformaed FIA returns in the bottom plot.

```
retFIA = edhec$FIA
iftrFIA = IF.mean(returns = retFIA,IFprint = T)
iftrFIAclean = IF.mean(returns = retFIA, cleanOutliers = T, eff = 0.99, IFprint = T)
par(mfrow = c(2,1))
plot(iftrFIA,main = "FIA Returns",lwd = .8)
plot(iftrFIAclean,main = "Outlier Cleaned FIA Returns",lwd = .8)
par(mfrow = c(1,1))
```

Figure 5: FIA Returns and Outlier Cleaned FIA Returns

Just as a check to compare the FIA returns and outlier cleaned FIA returns, we show in Figure 6 the former plot with a solid line, and the latter as overlaid dots

```
plot(iftrFIA, lwd = 0.8, main =
     "FIA Returns (line) and Outlier Cleaned FIA Returns (dots)")
points(iftrFIAclean,pch = 16)
```

Figure 6: Overlaid FIA Returns (line) and Outlier Cleaned Returns (dots)

5 Prewhitening

Spectral density function estimation is a frequently used method in the field of signal processing, and in other engineering and science applications. Prewhitening is a technique often used to improve the performance of spectral density estimators. Since the core of the method described in Chen and Martin [2018] is estimation of a spectral density at frequency zero of an influence-function transformed returns time series IF_t, it is natural to be able to use prewhitening of that time series. The `seCorIFPW` variant of the basic seCorIF method in the `EstimatorStandardError` package implement such prewhitening..

A prewhitened version IF_t^{pw} of the IF_t time series is computed as

$$IF_t^{pw} = IF_t - \hat{\rho} IF_{t-1} \tag{18}$$

where $\hat{\rho}$ is a lag-one serial correlation coefficient estimat of the IF_t. In general the IF_t^{pw} series is not a serially uncorrelated (white noise) series, but it has considerably less serial correlation than IF_t, and a periodogram estimator based on IF_t^{pw} will suffer from relatively little bias compared with one based on IF_t.

The following code computes, and plots in Figure 7, the IF.SR transformed FIA returns, and the prewhitened IF.SR transformed FIA returns.

```
iftrFIA = IF.SR(returns = retFIA,IFprint = T)
PWiftrFIA = IF.SR(returns = retFIA,prewhiten = T, IFprint = T)
par(mfrow = c(2,1))
plot(iftrFIA,main = "IF.SR Transformed FIA Returns",lwd = .8)
plot(PWiftrFIA,main = "Prewhitened IF.SR Transformed FIA Returns",lwd = .8)
par(mfrow = c(1,1))
```

Figure 7: IF.SR Transformed FIA Returns (top) and Prewhitened Version of the Same (bottom)

The following code computes and displays the lag-1 through lag-5 autocorrelations of the time series in Figure 7:

```
corr.iftrFIA = acf(iftrFIA, lag.max = 5, plot = F)
corr.PWiftrFIA = acf(PWiftrFIA, lag.max = 5, plot = F)
out = data.frame(round(rbind(corr.iftrFIA$acf,corr.PWiftrFIA$acf),2))
names(out) = c("Lag0","Lag1","Lag2","Lag3","Lag4","Lag5")
row.names(out) = c("autocorr: IF.SR FIA ret","autocorr: PW IF.SR FIA ret")
out[-1]

##                            Lag1  Lag2  Lag3 Lag4  Lag5
```

```
## autocorr: IF.SR FIA ret    0.50  0.17  0.05 0.02 -0.14
## autocorr: PW IF.SR FIA ret 0.05 -0.09 -0.04 0.09 -0.13
```

Outlier Cleaning Prior to Prewhitening

Since outliers can have adverse influence not only on risk and performance estimators, but also on the estimator $\hat{\rho}$ used for prewhitening, it is always a good idea to apply the outlier cleaning mehod of Section 4 before prewhitening. We illustrate doing this with the following code and plots, and autocorrelation calculations.

```
iftrFIAcl = IF.mean(returns = retFIA,cleanOutliers = T, IFprint = T)
PWiftrFIAcl = IF.mean(returns = retFIA,cleanOutliers = T,prewhiten = T, IFprint = T)
par(mfrow = c(2,1))
plot(iftrFIAcl,type = "b", main = "FIA Outlier Cleaned Returns",lwd = .8)
plot(PWiftrFIAcl,type = "b", main = "Prewhitened FIA Outlier Cleaned Returns",lwd = .8)
par(mfrow = c(1,1))
```

Figure 8: IF.SR Transformed Outlier Cleaned FIA Returns (top) and Prewhitened Version of the Same (bottom)

The following code computes and displays the lag-1 through lag-5 autocorrelations of the time series in Figure 8:

```
corr.iftrFIAcl = acf(iftrFIAcl, lag.max = 5, plot = F)
corr.PWiftrFIAcl = acf(PWiftrFIAcl, lag.max = 5, plot = F)
out = data.frame(round(rbind(corr.iftrFIAcl$acf,corr.PWiftrFIAcl$acf),2))
names(out) = c("Lag0","Lag1","Lag2","Lag3","Lag4","Lag5")
row.names(out) =  c("autocorr: IF.SR Cleaned FIA ret","autocorr: PW IF.SR Cleaned FIA ret")
out[,-1]


##                                     Lag1  Lag2 Lag3 Lag4  Lag5
## autocorr: IF.SR Cleaned FIA ret     0.39  0.14 0.18 0.06 -0.18
## autocorr: PW IF.SR Cleaned FIA ret -0.01 -0.07 0.15 0.09 -0.17
```

Appendix A. Estimator Functional Forms and Influence Function Formulas

Here we provide the functional form of all the estimators in Table2, and the formulas for their influence functions. The interested reader can find the derivations of these formulas in Zhang et al. (2019).

Mean

The functional representation $\mu(F)$ of the sample mean estimator is given by (1), and the formula for the influence function of the sample mean is:

$$IF(r;\mu;F) = r - \mu \tag{19}$$

where $\mu = \mu(F)$.

Standard Deviation

The functional form of standard deviation is

$$\sigma(F) = \left(\int (x - \mu(F))^2 dF(x) \right)^{1/2}. \tag{20}$$

and the formula for the influence function of the standard deviation is:

$$IF(r;\sigma;F) \quad = \quad \frac{1}{2\sigma} \left((r-\mu)^2 - \sigma^2 \right) \tag{21}$$

Semi-Standard Deviation (SSD)

The functional respresentation $SSD(F)$ of the sample semi-standard deviaton is

$$SSD(F) = \left(\int_{-\infty}^{\mu(F)} (x - \mu(F))^2 dF(x) \right)^{1/2} \tag{22}$$

and the formula for the influence function of the sample semi-standard deviation is:

$$IF(r; SSD; F) = \frac{(r-\mu)^2 \cdot \mathrm{I}(r \leq \mu) - 2 \cdot SMEAN \cdot (r - \mu) - SSD^2}{2 \cdot SSD} \tag{23}$$

where $\mu = \mu(F)$, $SSD = SSD(F)$, and $SMEAN$ is the "semi-mean" $SMEAN(F) = \int_{-\infty}^{\mu}(x-\mu)dF(x)$.

Lower Partial Moment (LPM)

There are two versions of LPM available, the lower partial moment of order one $LPM1_c$, and the lower partial moment of order two $LPM2_c$, whose functional representation are special cases of the order n lower partial moment

$$LPMn_c(F) = \int_{-\infty}^{c} (c - x)^n dF(x) \tag{24}$$

where c is a user specified constant "threshold". This threshold is oftern referred to as the *minimum acceptable return*, (MAR).

The influence function formulas of these two lower partial moments are:

$$IF(r; LPM1_c; F) = (c - r)\mathrm{I}(r \leq c) - LPM1_c \tag{25}$$

where $LPM1_c = LPM1_c(F)$, and

$$IF(r; LPM2_c; F) = (c - r)^2 I(r \leq c) - LPM2_c \tag{26}$$

where $LPM2_c = LPM_c(F)$,

Value-at-Risk (VaR)

The functional form for value-at-risk is

$$VaR_\alpha(F) = -q_\alpha(F). \tag{27}$$

The formula for the influence function of VaR is:

$$IF(r; VaR_\alpha; F) = \frac{1}{f(q_\alpha)} \left(\mathrm{I}(r \leq q_\alpha) - \alpha \right) \tag{28}$$

Expected Shortfall (ES)

The functional form of expected shortfall is

$$ES_\alpha(F) = -\frac{1}{\alpha}\int_{-\infty}^{q_\alpha(F)} x \cdot dF(x). \tag{29}$$

The formula for the influence function of the ES is:

$$IF(r; ES; F) = -\frac{\mathrm{I}(r \leq q_\alpha)}{\alpha} \cdot (r - q_\alpha) - q_\alpha - ES_\alpha \tag{30}$$

Sharpe Ratio (SR)

The functional reprsentation of the Sharpe ratio is

$$SR(F) = \frac{\mu(F) - r_f}{\sigma(F)} = \frac{\mu_e(F)}{\sigma(F)}. \tag{31}$$

and the influence function for the Sharpe-Ratio is

$$IF(r; SR; F) \quad = \quad -\frac{\mu_e}{2\sigma^3}(r-\mu)^2 + \frac{1}{\sigma}(r-\mu) + \frac{\mu_e}{2\sigma}$$

where $\mu_e = \mu_e(F) = \mu(F) - r_f$, and $\sigma = \sigma(F)$.

Sortino Ratio (SoR)

There are two versions of the Sortino ratio. The first version of SoR has the functional representation

$$SoR_c(F) = \frac{\mu(F) - r_f}{\sqrt{LPM2_c(F)}} = \frac{\mu_e(F)}{\sqrt{LPM2_c(F)}} \tag{32}$$

where the subscript c reflects the use of the constant threshold in the denominator lower partial moment of order two.

The second version of SoR has the functional form

$$SoR_\mu(F) = \frac{\mu(F) - r_f}{SSD(F)} = \frac{\mu_e(F)}{SSD(F)} \tag{33}$$

where the subscript μ indicates the use of the mean in the denominator semi-standard deviaton.

The influence function for $SoR_c(F)$ is

$$IF(r; SoR_c; F) = \frac{-SoR_c \cdot I(r \leq c)}{2 \cdot LPM2_c}(r-c)^2 + \frac{1}{\sqrt{LPM2_c}}(r-\mu) + \frac{SoR_c}{2} \tag{34}$$

where $\mu = \mu(F)$,$LPM2_c = LPM2_c(F)$, $SoR_c = SoR_c(F)$.

The influence function for $SoR_\mu(F)$ is

$$IF(r; SoR_\mu; F) = -\frac{SoR_\mu \cdot I(r \leq \mu)}{2 \cdot SSD^2}(r-\mu)^2 + \left(\frac{SoR_\mu \cdot SMEAN}{SSD^2} + \frac{1}{SSD}\right)(r-\mu) + \frac{SoR_\mu}{2} \tag{35}$$

where $\mu = \mu(F)$,$SSD = SSD(F)$, $SoR_\mu = SoR_\mu(F)$.

Expected Shortfall Ratio (ESratio)

The functional form is

$$ESratio(F) = \frac{\mu(F) - r_f}{ES_\alpha(F)} = \frac{\mu_e(F)}{ES_\alpha(F)} \tag{36}$$

The formula for the influence function of the ESratio is:

$$IF(r; ESratio; F) = \frac{r-\mu}{ES_\alpha} - \frac{ESratio}{ES_\alpha}\left(-q_\alpha - ES_\alpha - \frac{I(r \leq q_\alpha)}{\alpha} \cdot (r - q_\alpha)\right) \tag{37}$$

Value-at-Risk Ratio (VaRratio)

The functional form of the VaR ratio is:

$$VaRratio(F) = \frac{\mu(F) - r_f}{VaR(F)} = \frac{\mu_e(F)}{-q_\alpha(F)} \tag{38}$$

The formula for the influence function of the VaRratio is:

$$IF(r; VaRratio; F) = -\frac{r-\mu}{q_\alpha} + \frac{VaRratio}{q_\alpha} \times \frac{\mathrm{I}(r \leq q_\alpha) - \alpha}{f(q_\alpha)}. \tag{39}$$

Rachev Ratio (RachR)

The functional form is:

$$RachR(F) = \frac{EG_\beta}{ES_\alpha} = \frac{\frac{1}{\beta} \cdot \int_{q_{1-\beta}}^{+\infty} x dF(x)}{-\frac{1}{\alpha} \cdot \int_{-\infty}^{q_\alpha} x dF(x)}$$

where $ES_\alpha = ES_\alpha(F)$ is the expected shortfall at level α and $EG_\beta = EG_\beta(F)$ is the expected tail gain at upper β-quantile defined by the following equation

$$EG_\beta = \frac{1}{\beta} \cdot \int_{q_{1-\beta}}^{+\infty} x dF(x)$$

The formula for the influence function of the Rachev ratio is:

$$IF(r; RaR; F) = \frac{1}{ES_\alpha}\left(\frac{I(r \geq q_{1-\beta})}{\beta}(r - q_{1-\beta}) + q_{1-\beta} - EG_\beta\right) - \frac{RachR}{ES_\alpha} \cdot \left(\frac{-I(r \leq q_\alpha)}{\alpha}(r - q_\alpha) - q_\alpha - ES_\alpha\right) \tag{40}$$

Omega Ratio (Omega)

The functional is:

$$\Omega(F) \quad = \frac{UPM1_c(F)}{LPM1_c(F)} \quad = \frac{\int_c^{+\infty}(x - c)\, f(x) dx}{\int_{-\infty}^{c}(c - x)\, f(x) dx}$$

The formula for the influence function of the Omega ratio is:

$$IF(r; \Omega; F) = \frac{1}{LPM1_c} \cdot \Big((r - c) \cdot I(r \geq c) - UPM1_c\Big) - \frac{\Omega}{LPM1_c} \cdot \Big((c - r) \cdot \mathrm{I}(r \leq c) - LPM1_c\Big) \tag{41}$$

References

X. Chen and R. D. Martin. Standard errors of risk and performance measure estimators for serially correlated returns. 2018. URL https://ssrn.com/abstract=3085672.