Descriptive Statistics and Stylized Facts

Herbert (Yuming) Liu & Eric Zivot

4/12/2021

R Set-up for Examples

```r
options(digits=3, width=70)
# Load packages
library(car)
library(corrplot)
library(dygraphs)
library(IntroCompFinR)
library(PerformanceAnalytics)
library(sn)
library(tseries)
library(zoo)
Sys.setenv(TZ="UTC")
```

Data for Examples

Daily prices on Microsoft and S&P500 Index from 1993-01-04 to 2014-12-31 taken from Yahoo!

```
data(msftDailyPrices, sp500DailyPrices)
smpl = "1993-01::2014-12"
msftDailyPrices = msftDailyPrices[smpl]
sp500DailyPrices = sp500DailyPrices[smpl]
msftMonthlyPrices = to.monthly(msftDailyPrices, OHLC=FALSE)
sp500MonthlyPrices = to.monthly(sp500DailyPrices, OHLC=FALSE)
msftSp500DailyPrices = merge(msftDailyPrices, sp500DailyPrices
msftSp500MonthlyPrices = merge(msftMonthlyPrices, sp500Monthly
```

- **xts** function `to.monthly()` extracts end-of-month value from daily data

Price data

```r
head(msftSp500DailyPrices, n=3)
```

```
##            MSFT SP500
## 1993-01-04 1.89   435
## 1993-01-05 1.92   434
## 1993-01-06 1.98   435
```

```r
head(msftSp500MonthlyPrices, n=3)
```

```
##          MSFT SP500
## Jan 1993 1.92   439
## Feb 1993 1.85   443
## Mar 1993 2.06   452
```

Compute Returns

Use **PerformanceAnalytics** function `Return.calculate()` to compute simple returns:

```
msftMonthlyRetS = Return.calculate(msftMonthlyPrices,
                                   method="simple")
msftDailyRetS = Return.calculate(msftDailyPrices,
                                 method="simple")
sp500MonthlyRetS = Return.calculate(sp500MonthlyPrices,
                                    method="simple")
sp500DailyRetS = Return.calculate(sp500DailyPrices,
                                  method="simple")
msftSp500MonthlyRetS = Return.calculate(msftSp500MonthlyPrices
                                        method="simple")
msftSp500DailyRetS = Return.calculate(msftSp500DailyPrices,
                                      method="simple")
```

Compute CC Returns

```
msftMonthlyRetC = log(1 + msftMonthlyRetS)
msftDailyRetC = log(1 + msftDailyRetS)
sp500MonthlyRetC = log(1 + sp500MonthlyRetS)
sp500DailyRetC = log(1 + sp500DailyRetS)
msftSp500MonthlyRetC = merge(msftMonthlyRetC,
                             sp500MonthlyRetC)
msftSp500DailyRetC = merge(msftDailyRetC,
                           sp500DailyRetC)
```

Return data

```
head(msftSp500DailyRetS, n=3)
```

```
##                 MSFT     SP500
## 1993-01-05  0.0159 -0.002389
## 1993-01-06  0.0312  0.000414
## 1993-01-07 -0.0202 -0.008722
```

```
head(msftSp500MonthlyRetS, n=3)
```

```
##             MSFT   SP500
## Feb 1993 -0.0365  0.0105
## Mar 1993  0.1135  0.0187
## Apr 1993 -0.0777 -0.0254
```

Covariance Stationarity

$$\{\ldots, X_1, \ldots, X_T, \ldots\} = \{X_t\}$$

is a covariance stationary stochastic process, and each X_t is identically distributed with unknown pdf $f(x)$.

Recall,

$$\begin{aligned} E[X_t] &= \mu \text{ indep of } t \\ \mathrm{var}(X_t) &= \sigma^2 \text{ indep of } t \\ \mathrm{cov}(X_t, X_{t-j}) &= \gamma_j \text{ indep of } t \\ \mathrm{cor}(X_t, X_{t-j}) &= \rho_j \text{ indep of } t \end{aligned}$$

Observed Sample and Descriptive Statistics

$$\{X_1 = x_1, \ldots, X_T = x_T\} = \{x_t\}_{t=1}^T$$

are observations generated by the stochastic process $\{X_t\}$.

Descriptive Statistics:

Data summaries (statistics) to describe certain features of the data $\{x_t\}_{t=1}^T$, to learn about the unknown pdf, $f(x)$, and to capture the observed dependencies in the data.

- Can be graphical or numerical

Time Plots

Line plot of time series data with time/dates on horizontal axis

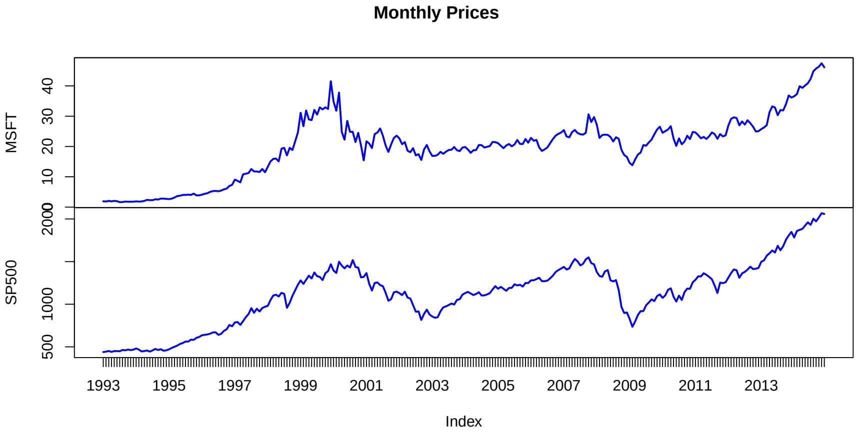
- Visualization of data - uncover trends, assess stationarity and time dependence
- Spot unusual behavior
- Plotting multiple time series can reveal commonality across series

R Functions

Function	Package	Description
`plot.ts`	**stats**	basic line plot
`plot.zoo`	**zoo**	plot `xts/zoo` time series
`plot.xts`	**xts**	plot `xts` time series
`autoplot`	**forecast**	`ggplot` for time series

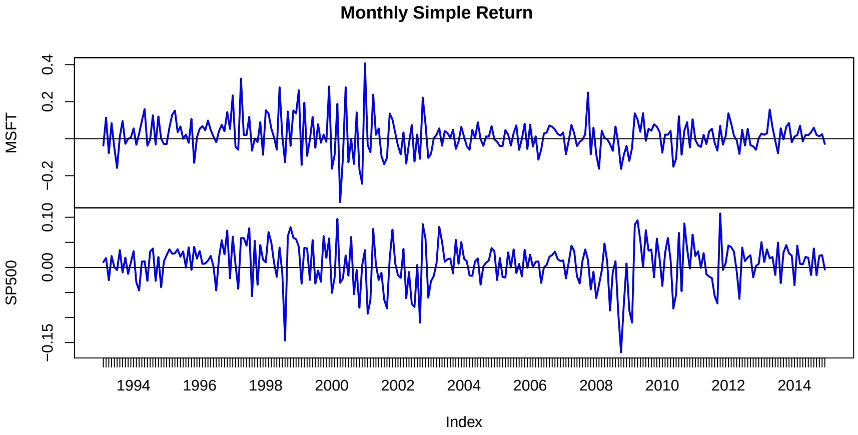
Line Plot of Monthly Prices

```
plot.zoo(msftSp500MonthlyPrices, main="Monthly Prices",
         lwd=2, col="blue")
```

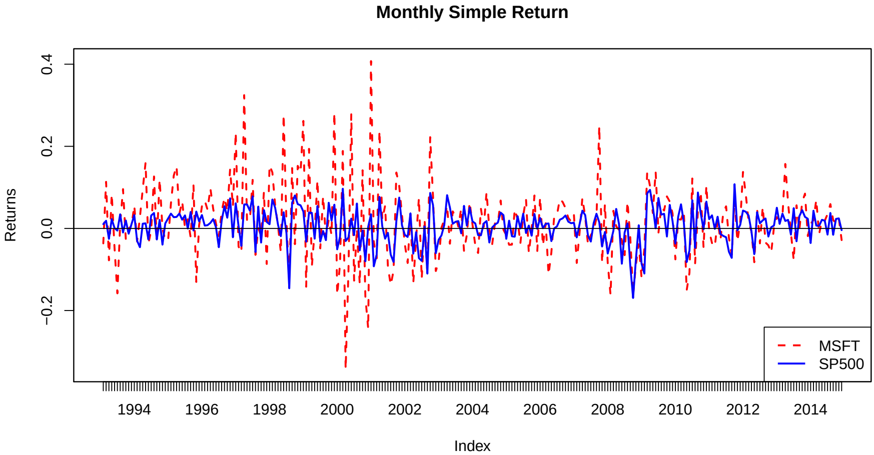


Line Plot of Monthly Returns

```
plot.zoo(msftSp500MonthlyRetS, main="Monthly Simple Return",
         panel=my.panel,lwd=2, col="blue")
```

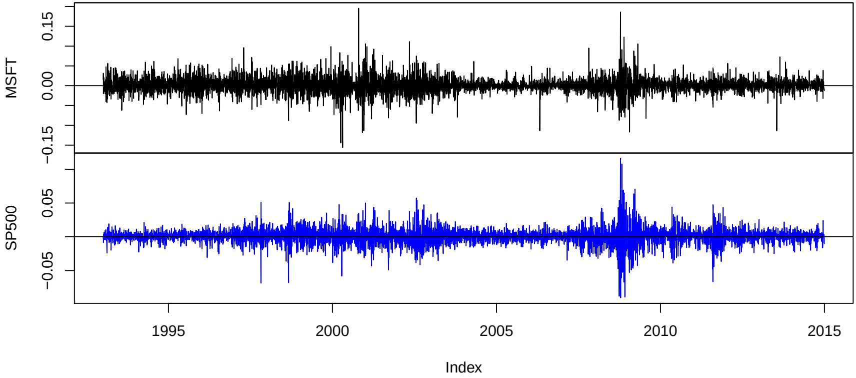


Line Plot of Monthly Returns - Same Plot

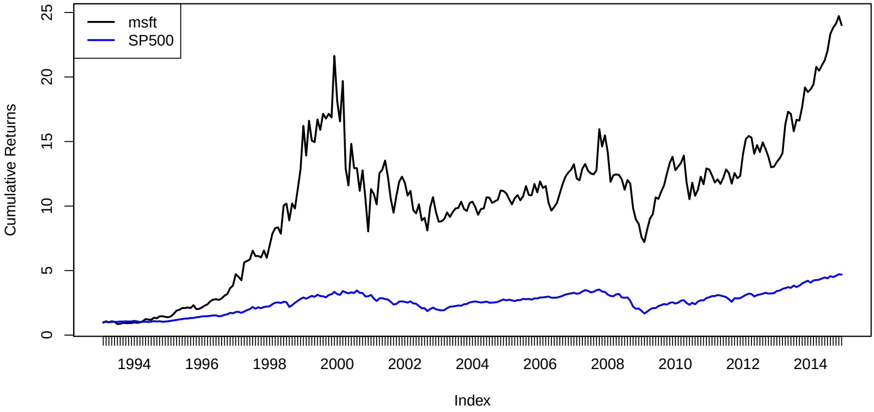


Plotting Daily Returns

```
plot.zoo(msftSp500DailyRetS, main="",
     panel=my.panel, col=c("black", "blue"))
```



Equity Curves for MSFT and SP500



Histograms

Goal: Describe the shape of the distribution of the data $\{x_t\}_{t=1}^{T}$

Hisogram Construction:

- Order data from smallest to largest values
- Divide range into N equally spaced bins

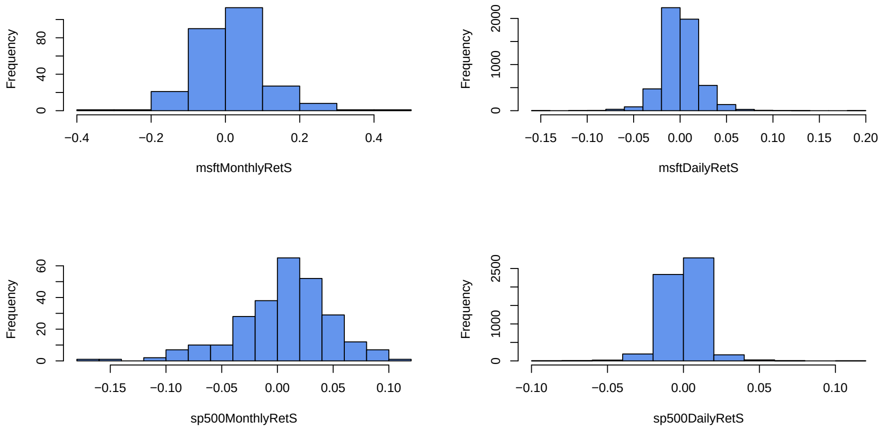
$$[-| - | - | \cdots | - | - |-]$$

- Count number of observations in each bin
- Create bar chart (optionally normalize area to equal 1)

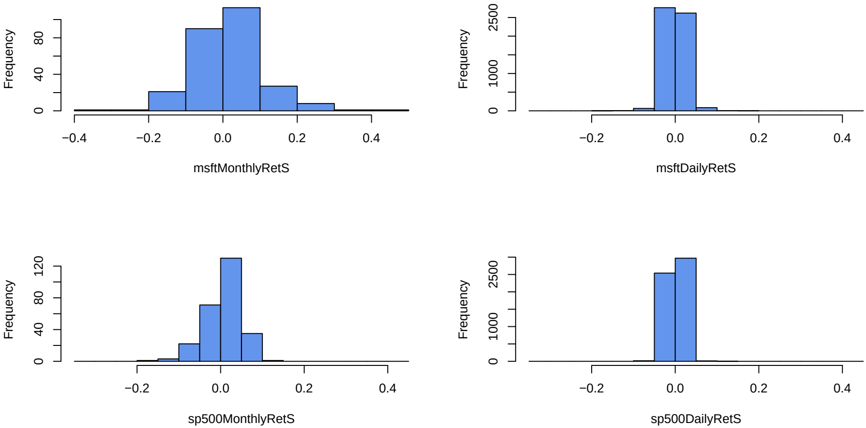
R Functions

Function	Description
`hist()`	compute histogram
`density()`	compute smoothed histogram

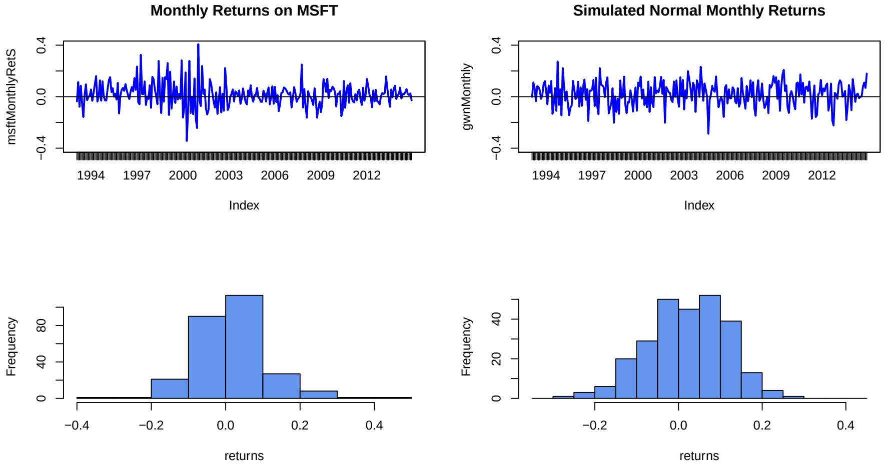
Histograms of Daily and Monthly Returns



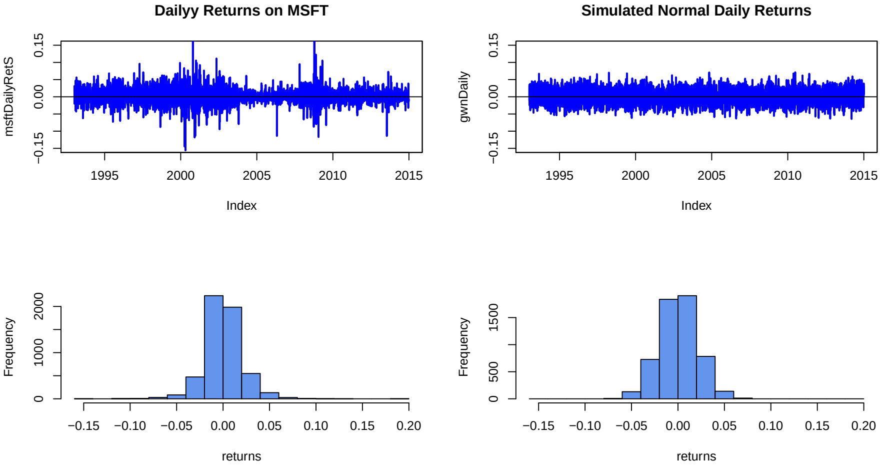
Histograms: Same Bins



MSFT Monthly Returns vs. Normal Distribution



MSFT Daily Returns vs. Normal Distribution



Smoothed Histogram

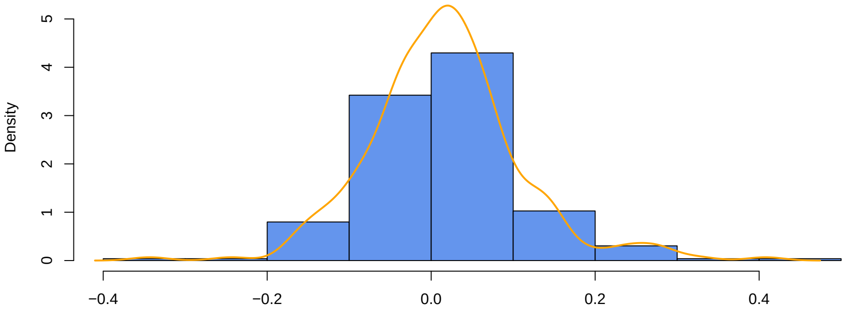
The `density()` function computes a smoothed (kernel density) estimate of the unknown pdf at the point x using the formula

$$
\begin{aligned}
\hat{f}(x) &= \frac{1}{Tb}\sum_{t=1}^{T} k\left(\frac{x - x_t}{b}\right) \\
k(\cdot) &= \text{kernel function} \\
b &= \text{ bandwidth (smoothing) parameter}
\end{aligned}
$$

where $k(\cdot)$ is a pdf symmetric about zero (typically the standard normal distribution). See Ruppert Chapter 4 for details.

Smoothed Histogram for MSFT Monthly returns

```
msft.density = density(msftMonthlyRetS)
hist(msftMonthlyRetS, main="", xlab="Monthly Returns",
     col="cornflowerblue", probability=TRUE, ylim=c(0,5.5))
points(msft.density,type="l", col="orange", lwd=2)
```



Empirical Quantiles/Percentiles

For $\alpha \in [0, 1]$, the $100 \times \alpha^{th}$ percentile (empirical quantile) of a sample of data is the data value $\hat{q}_\alpha$ such that $\alpha \cdot 100\%$ of the data are less than $\hat{q}_\alpha$.

Quartiles:

$$\begin{aligned}
\hat{q}_{.25} &= \text{ first quartile} \\
\hat{q}_{.50} &= \text{second quartile (median)} \\
\hat{q}_{.75} &= \text{third quartile} \\
\hat{q}_{.75} - \hat{q}_{.25} &= \text{interquartile range (IQR)}
\end{aligned}$$

R functions

- `sort()`: sort elements of data vector
- `min()`: compute minimum value of data vector
- `max()`: compute maximum value of data vector
- `range()`: compute min and max of a data vector
- `quantile()`: compute empirical quantiles
- `median()`: compute median
- `IQR()`: compute inter-quartile range
- `summary()`: compute summary statistics

Empirical Quantiles for Monthly Returns

```r
apply(msftSp500MonthlyRetS, 2, quantile)
```

```
##          MSFT   SP500
## 0%   -0.3434 -0.1694
## 25%  -0.0379 -0.0174
## 50%   0.0146  0.0121
## 75%   0.0629  0.0345
## 100%  0.4076  0.1077
```

```r
apply(msftSp500MonthlyRetS, 2, quantile,
      probs = c(0.01, 0.025, 0.05))
```

```
##         MSFT   SP500
## 1%   -0.165 -0.1100
## 2.5% -0.154 -0.0858
## 5%   -0.127 -0.0724
```

Empirical Quantiles for Daily Returns

```
apply(msftSp500DailyRetS, 2, quantile)
```

```
##             MSFT     SP500
## 0%   -0.15598 -0.090350
## 25%  -0.00956 -0.004704
## 50%   0.00000  0.000617
## 75%   0.01067  0.005739
## 100%  0.19555  0.115800
```

```
apply(msftSp500DailyRetS, 2, quantile,
      probs = c(0.01, 0.025, 0.05))
```

```
##            MSFT   SP500
## 1%   -0.0541 -0.0318
## 2.5% -0.0383 -0.0238
## 5%   -0.0297 -0.0180
```

Historical Value-at-Risk (VaR)

Let $\{R_t\}_{t=1}^{T}$ denote a sample of T *simple* monthly returns on an investment, and let $\$W_0$ be the initial value of an investment. For $\alpha \in (0,1)$, the historical VaR_α is

$$\begin{aligned} &\$W_0 \times \hat{q}_\alpha^R \\ \hat{q}_\alpha^R &= \text{empirical } \alpha \cdot 100\% \text{ quantile of} \{R_t\}_{t=1}^{T} \end{aligned}$$

For *continuously compounded* returns $\{r_t\}_{t=1}^{T}$ use

$$\begin{aligned} &\$W_0 \times (\exp(\hat{q}_\alpha^r) - 1) \\ \hat{q}_\alpha^r &= \text{empirical } \alpha \cdot 100\% \text{ quantile of} \{r_t\}_{t=1}^{T} \end{aligned}$$

Historical VaR

Pros:

- By using the empirical quantiles $\hat{q}_{\alpha}^{R}$ Historical VaR does not assume any particular probability distribution for simples returns (i.e., does not assume returns are normally distributed)

Cons:

- You need a lot of data for $\hat{q}_{\alpha}^{R}$ to be a good estimate of q_{α}^{R}, especially for α close to 0 (e.g. 0.05, 0.025, 0.01)

Example: Historical 1% and 5% Monthly VaR

Consider a $100,000 investment for 1-Month in MSFT and SP500:

```
W = 100000
msftQuantiles = quantile(msftMonthlyRetS, probs=c(0.01, 0.05))
sp500Quantiles = quantile(sp500MonthlyRetS, probs=c(0.01, 0.05
msftVaR = W*msftQuantiles
sp500VaR = W*sp500Quantiles
cbind(msftVaR, sp500VaR)
```

```
##    msftVaR sp500VaR
## 1%  -16462   -10997
## 5%  -12729    -7239
```

Sample Statistics

Plug-In Principle: Estimate population quantities using sample statistics

Sample Average (Mean)

$$\frac{1}{T}\sum_{t=1}^{T} x_t = \bar{x} = \hat{\mu}_x$$

Sample Variance

$$\frac{1}{T-1}\sum_{t=1}^{T}(x_t - \bar{x})^2 = s_x^2 = \hat{\sigma}_x^2$$

Sample Standard Deviation

$$\sqrt{s_x^2} = s_x = \hat{\sigma}_x$$

Sample Statistics

Sample Skewness

$$\frac{1}{T-1}\sum_{t=1}^{T}(x_t-\bar{x})^3/s_x^3=\widehat{skew}$$

Sample Kurtosis

$$\frac{1}{T-1}\sum_{t=1}^{T}(x_t-\bar{x})^4/s_x^4=\widehat{kurt}$$

Sample Excess Kurtosis

$$\widehat{kurt}-3$$

R Functions

Function	Package	Description
`mean()`	**base**	compute sample mean
`colMeans()`	**base**	compute column means of matrix
`var()`	**stats**	compute sample variance
`sd()`	**stats**	compute sample standard deviation
`skewness()`	**PerformanceAnalytics**	compute sample skewness
`kurtosis()`	**PerformanceAnalytics**	compute sample excess kurtosis

- Use the R function `apply()`, to apply functions over rows or columns of a `matrix` or `data.frame`

Sample Shape Statistics for Monthly Returns

```
statsMonthly = rbind(apply(msftSp500MonthlyRetS, 2, mean),
                     apply(msftSp500MonthlyRetS, 2, var),
                     apply(msftSp500MonthlyRetS, 2, sd),
                     apply(msftSp500MonthlyRetS, 2, skewness),
                     apply(msftSp500MonthlyRetS, 2, kurtosis))
rownames(statsMonthly) = c("Mean", "Variance", "Std Dev",
                           "Skewness",  "Excess Kurtosis")
```

Sample Shape Statistics for Monthly returns

```
round(statsMonthly, digits=4)
```

```
##                     MSFT   SP500
## Mean              0.0164  0.0068
## Variance          0.0087  0.0018
## Std Dev           0.0933  0.0424
## Skewness          0.4553 -0.7169
## Excess Kurtosis 2.1962  1.2910
```

Sample Shape Statistics for Daily Returns

```
##                   MSFT   SP500
## Mean            0.0008  0.0003
## Variance        0.0004  0.0001
## Std Dev         0.0205  0.0118
## Skewness        0.2158 -0.0609
## Excess Kurtosis 6.6294  8.8279
```

Empirical Cumulative Distribution Function

Recall, the CDF of a random variable X is

$$F_X(x) = \Pr(X \leq x)$$

The empirical CDF of a random sample is

$$\begin{aligned}\hat{F}_X(x) &= \frac{1}{n}(\#x_i \leq x) \\ &= \frac{\text{number of } x_i \text{ values } \leq x}{\text{sample size}}\end{aligned}$$

Empirical Cumulative Distribution Function

How to compute and plot $\hat{F}_X(x)$ for a sample $\{x_1, \ldots, x_n\}$:

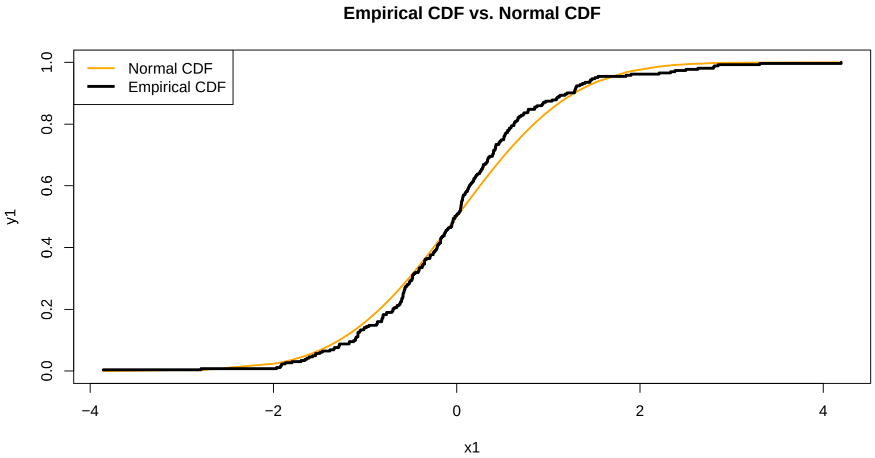
- Sort data from smallest to largest values: $\{x_{(1)}, \ldots, x_{(n)}\}$
- Compute $\hat{F}_X(x)$ at these points
- Plot $\hat{F}_X(x)$ against sorted data $\{x_{(1)}, \ldots, x_{(n)}\}$
- Alternatively, use the R function `ecdf()`

Note: $x_{(1)}, \ldots, x_{(n)}$ are called the *order statistics*. In particular, $x_{(1)} = \min(x_1, \ldots, x_n)$ and $x_{(n)} = \max(x_1, \ldots, x_n)$.

Compare Empirical CDF with Normal CDF for MSFT Monthly Returns

```
# standardize returns to have mean zero and sd 1
z1 = scale(coredata(msftMonthlyRetS))
n1 = length(msftMonthlyRetS)
# compute empirical CDF
F.hat = 1:n1/n1
# sort from smallest to largest
x1 = sort(z1)
# compute standard normal cdf at x1
y1 = pnorm(x1)
```

Compare Empirical CDF with Normal CDF for MSFT



Quantile-Quantile (QQ) Plots

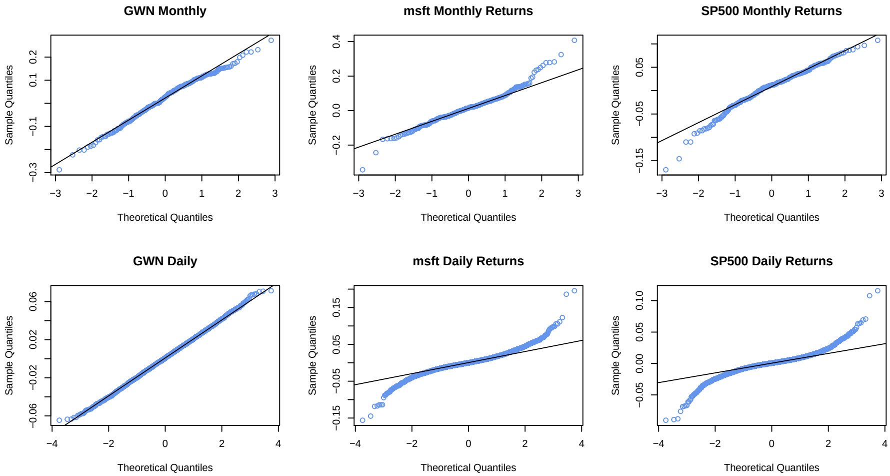
A *QQ plot* is useful for comparing your data with the quantiles of a distribution (usually the normal distribution) that you think is appropriate for your data. You interpret the QQ plot in the following way:

- If the points fall close to a straight line, your conjectured distribution is appropriate
- If the points do not fall close to a straight line, your conjectured distribution is not appropriate and you should consider a different distribution

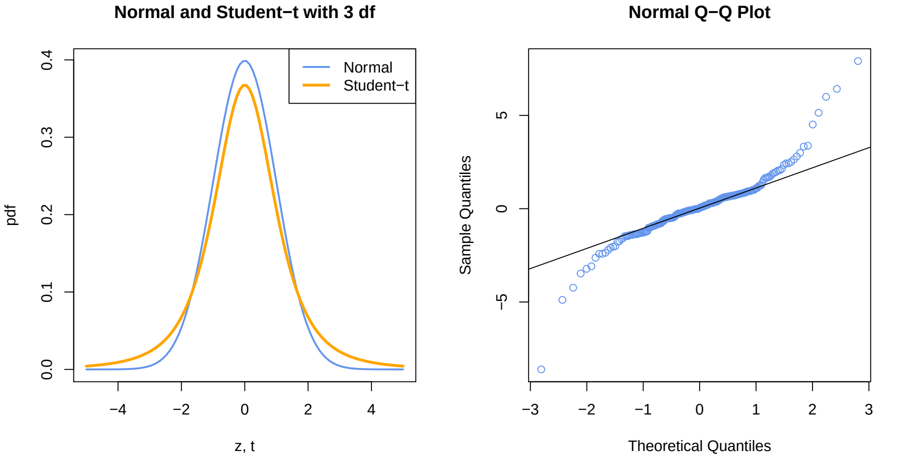
R functions

Function	Package	Description
`qqnorm()`	**stats**	QQ-plotagainst normal distribution
`qqline()`	**stats**	draw straight line on QQ-plot
`qqPlot()`	**car**	QQ-plot against specified distribution

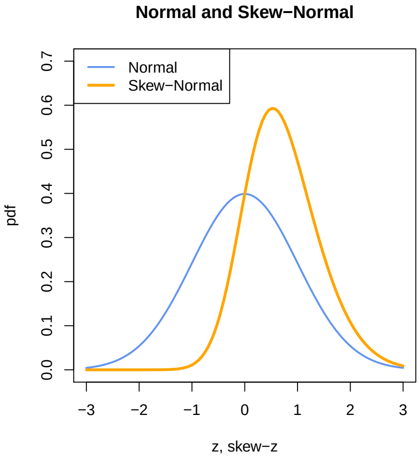
Normal QQ-plots for GWN, Microsoft and S&P 500 returns

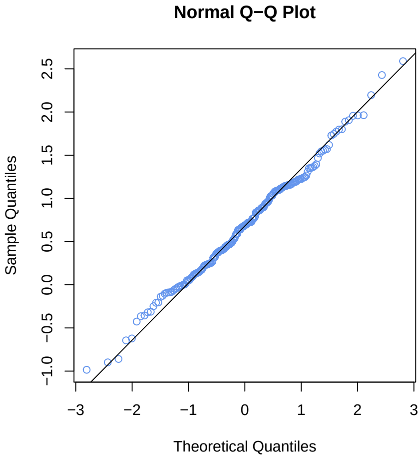


QQ-Plot for Fat-Tailed Data

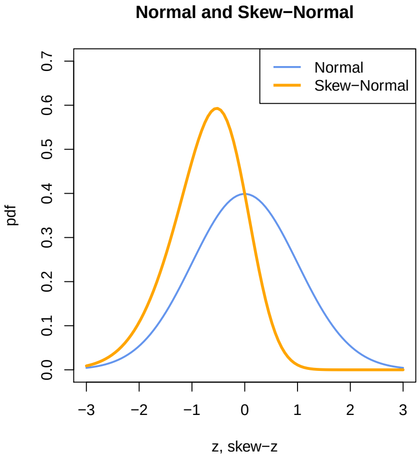


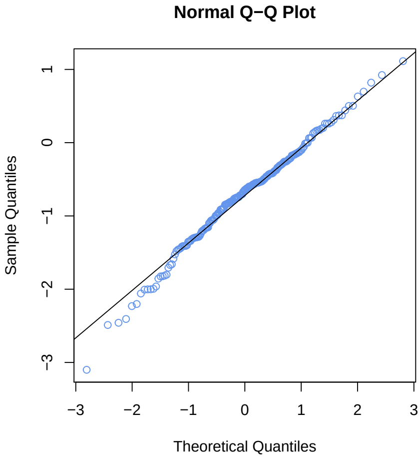
QQ-Plot for Right Skewed Data





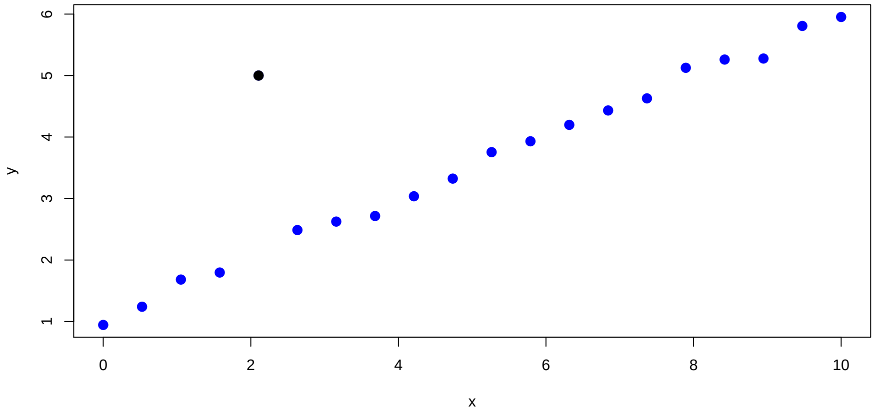
QQ-Plot for Left Skewed Data





Outliers

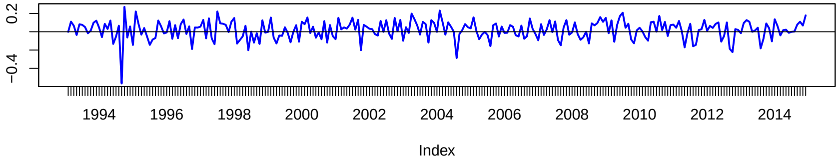
Can you spot the outlier?

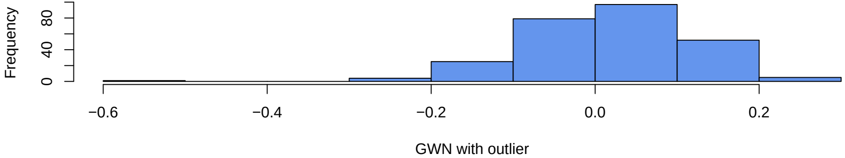


Outliers

- Extremely large, small or unusual values are called *outliers*
- Outliers can greatly influence the values of common descriptive statistics. In particular, the sample mean, variance, standard deviation, skewness and kurtosis
- Percentile measures are more robust to outliers: outliers do not greatly influence these measures (e.g. median instead of mean; IQR instead of SD)
- Bad rule-of-thumb: outlier $= \hat{\mu} \pm 3 \times \hat{\sigma}$. Why?

Effect of outliers on sample statistics





Effect of outliers on sample statistics

```
##                 GWN GWN.Outlier pctchange
## Mean         0.0203      0.0186   -0.0854
## Var          0.0089      0.0101    0.1398
## SD           0.0943      0.1006    0.0676
## skewness -0.3097     -0.9489    2.0640
## kurtosis  2.8895      6.5464    1.2656
## median    0.0276      0.0276    0.0000
## IQR       0.1292      0.1292    0.0000
```

Defining Outliers

Recall, the IQR is a robust measure of spread:

$$IQR = \hat{q}_{.75} - \hat{q}_{.25}$$

Moderate Outlier

$$\begin{gathered}\hat{q}_{.75} + 1.5 \cdot IQR < x < \hat{q}_{.75} + 3 \cdot IQR \\ \hat{q}_{.25} - 3 \cdot IQR < x < \hat{q}_{.25} - 1.5 \cdot IQR\end{gathered}$$

Extreme Outlier

$$\begin{gathered}x > \hat{q}_{.75} + 3 \cdot IQR \\ x < \hat{q}_{.25} - 3 \cdot IQR\end{gathered}$$

Boxplots

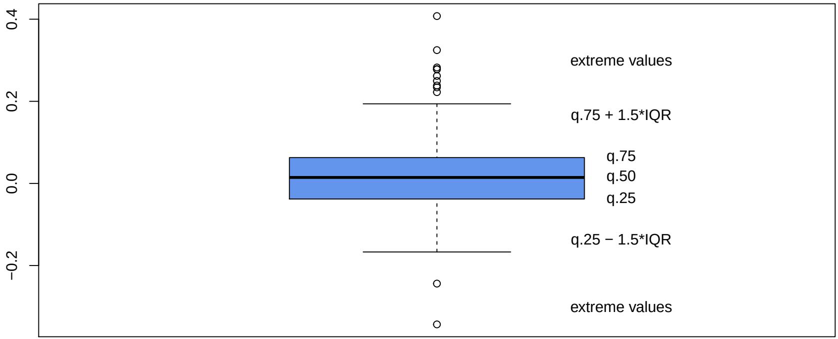
A *box plot* displays the locations of the basic features of the distribution of one-dimensional data

- The *box* shows the median, the upper and lower quartiles,
- The outer fences that indicate the extent of your data beyond the quartiles, and outliers, if any.
- Gives a graphical summary of a data distribution using outlier robust statistics

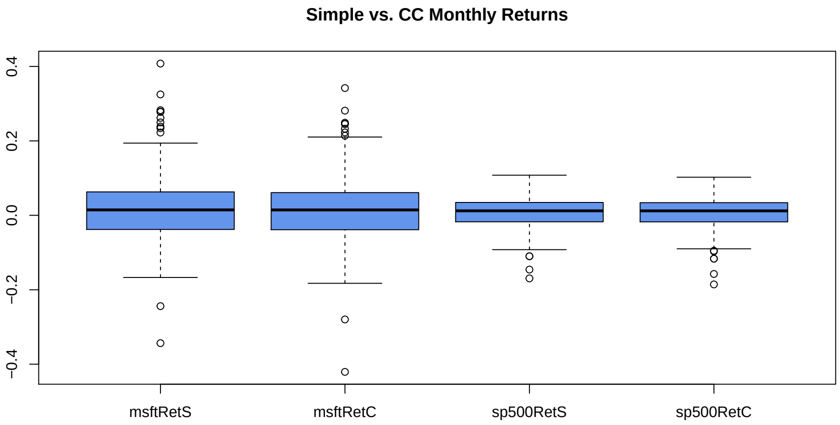
R functions

Function	Package	Description
`boxplot()`	**graphics**	box plots for multiple series
`chart.Boxplot()`	**PerformanceAnalytics**	box plots for asset returns

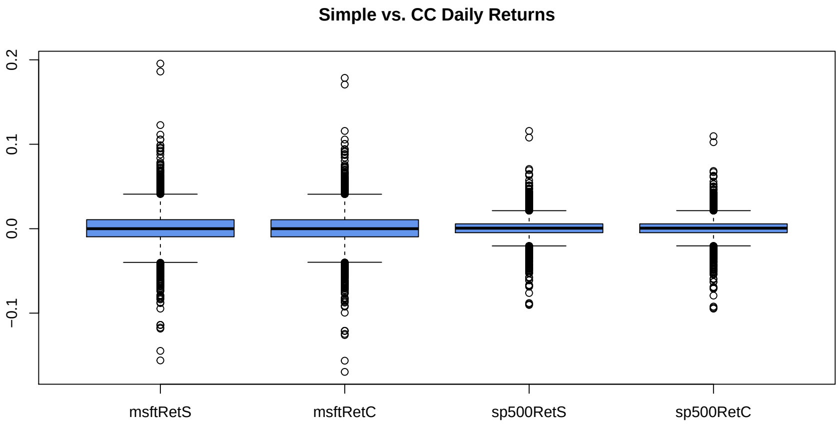
Example Boxplot for MSFT Monthly Returns



Boxplots: Comparing Return Distributions



Boxplots: Comparing Return Distributions



Bivariate Descriptive Statistics

$$\{\ldots,(X_1,Y_1),(X_2,Y_2),\ldots(X_T,Y_T),\ldots\}=\{(X_t,Y_t)\}$$

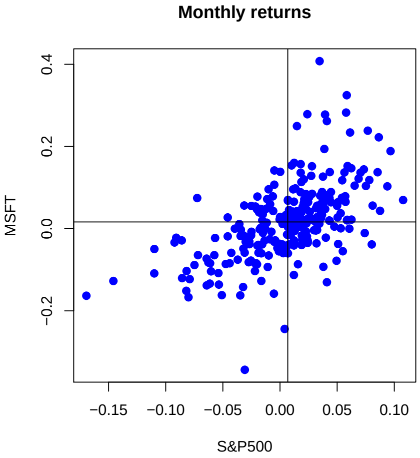
covariance stationary bivariate stochastic process with realized values

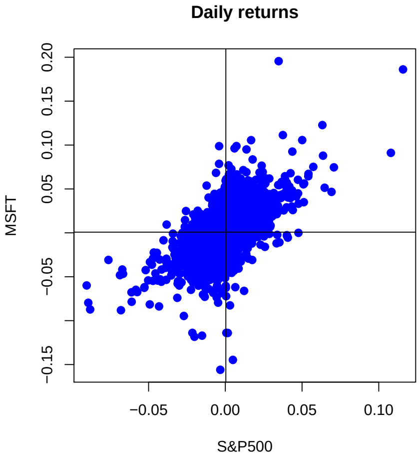
$$\{(x_1,y_1),(x_2,y_2),\ldots(x_T,y_T)\}=\{(x_t,y_t)\}_{t=1}^{T}$$

Scatterplot

- XY plot of bivariate data
- R functions: `plot()`, `pairs()`

Scatterplots of Returns





Sample Covariance and Correlation

Sample Covariance

$$\frac{1}{T-1}\sum_{t=1}^{T}(x_t - \bar{x})(y_t - \bar{y}) = s_{xy} = \hat{\sigma}_{xy}$$

Sample Correlation

$$\frac{\hat{\sigma}_{xy}}{\hat{\sigma}_x \hat{\sigma}_y} = \hat{\rho}_{xy} = \frac{s_{xy}}{s_x s_y} = r_{xy}$$

Sample Covariance and Correlation Matrices

For a data set of T observations on N asset returns

$$\hat{\Sigma} = \begin{bmatrix} \hat{\sigma}_1^2 & \hat{\sigma}_{12} & \cdots & \hat{\sigma}_{1N} \\ \hat{\sigma}_{12} & \hat{\sigma}_2^2 & \cdots & \hat{\sigma}_{2N} \\ \vdots & \vdots & \ldots & \vdots \\ \hat{\sigma}_{1N} & \hat{\sigma}_{2N} & \cdots & \hat{\sigma}_N^2 \end{bmatrix}, \hat{R} = \begin{bmatrix} 1 & \hat{\rho}_{12} & \cdots & \hat{\rho}_{1N} \\ \hat{\rho}_{12} & 1 & \cdots & \hat{\rho}_{2N} \\ \vdots & \vdots & \ldots & \vdots \\ \hat{\rho}_{1N} & \hat{\rho}_{2N} & \cdots & 1 \end{bmatrix}$$

R functions

Function	Package	Description
`var()`	**stats**	compute sample cov matrix
`cov()`	**stats**	compute sample cov matrix
`cor()`	**stats**	compute sample cor matrix
`corrplot()`	**corrplot**	visualize sample cor matrix

Sample Covariance and Correlation Matrices of Returns

Covariance matrix of monthly returns:

```
cov(msftSp500MonthlyRetS)
```

```
##          MSFT   SP500
## MSFT  0.00870 0.00228
## SP500 0.00228 0.00179
```

Correlation matrix of monthly returns

```
cor(msftSp500MonthlyRetS)
```

```
##        MSFT SP500
## MSFT  1.000 0.577
## SP500 0.577 1.000
```

Sample Covariance and Correlation Matrices of Returns

Covariance matrix of Daily returns:

```
cov(msftSp500DailyRetS)
```

```
##           MSFT    SP500
## MSFT  0.000420 0.000152
## SP500 0.000152 0.000138
```

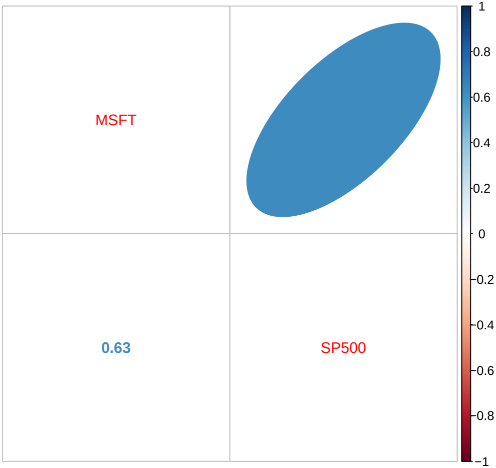
Correlation matrix of Daily returns

```
cor(msftSp500DailyRetS)
```

```
##       MSFT SP500
## MSFT  1.00  0.63
## SP500 0.63  1.00
```

Visualize Sample Correlation Matrix

```
corrplot.mixed(cor.mat, lower="number", upper="ellipse")
```



Time Series Descriptive Statistics

Sample Autocovariance

$$\hat{\gamma}_j = \frac{1}{T-1} \sum_{t=j+1}^{T} (x_t - \bar{x})(x_{t-j} - \bar{x}), j = 1, 2, \ldots$$

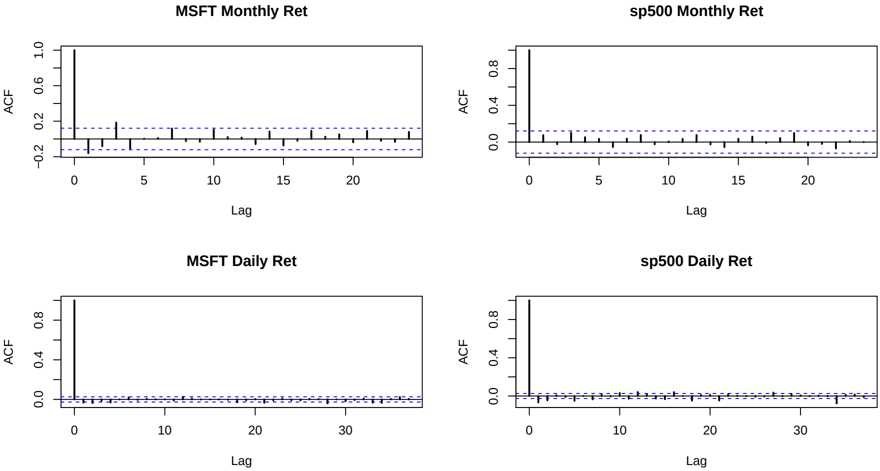
Sample Autocorrelation

$$\hat{\rho}_j = \frac{\hat{\gamma}_j}{\hat{\sigma}^2},\ j = 1, 2, \ldots$$

R functions

Function	Package	Description
`acf()`	**stats**	plot $\hat{\rho}_j$
`chart.ACF()`	**PerformanceAnalytics**	plot $\hat{\rho}_j$
`chart.ACFplus()`	**PerformanceAnalytics**	plot $\hat{\rho}_j$

Sample Autocorrelation Function (SACF) of Returns



Rolling Means

Idea: compute estimate of μ_i over rolling windows of length $n < T$:

$$\hat{\mu}_{it}(n) = \frac{1}{n}\sum_{j=0}^{n-1} R_{it-j}$$
$$= \frac{1}{n}(R_{it} + R_{it-1} + \cdots + R_{it-n+1})$$

for $t = n, \cdots, T$.

$\hat{\mu}_{in}(n)$ is the sample mean of the returns $\{R_{it}\}_{t=1}^{n}$ over the first sub-sample window of size n.

Similarly, $\hat{\mu}_{in+1}(n)$ is the sample mean of the returns $\{R_{it}\}_{t=2}^{n+1}$.

Rolling estimates: $\{\hat{\mu}_{in}(n), \hat{\mu}_{in+1}(n), \cdots, \hat{\mu}_{iT}(n)\}$

Rolling Means Example

```
roll.data = merge(msftMonthlyRetS,sp500MonthlyRetS,gwnMonthly)
colnames(roll.data) = c("MSFT", "SP500", "GWN")
roll.muhat = rollapply(roll.data, width=24, by=1,
                       by.column=TRUE, FUN=mean, align="right")
class(roll.muhat)
```

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

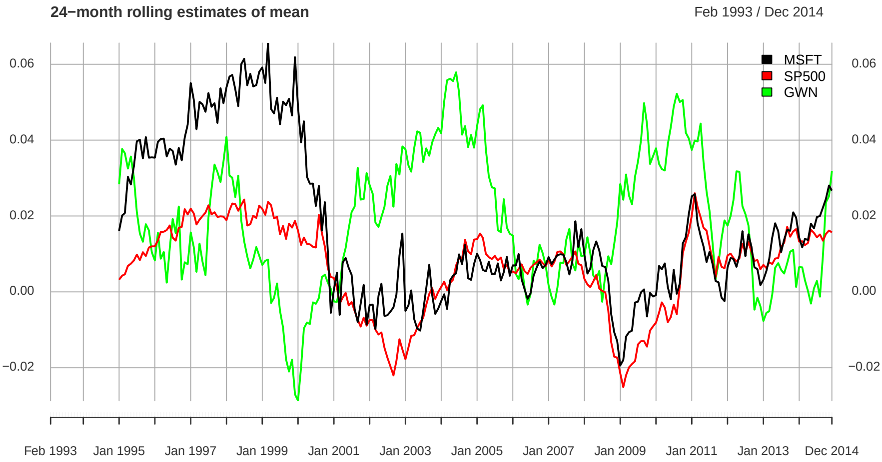
```
head(roll.muhat, n=1)
```

```
##          MSFT SP500 GWN
## Feb 1993   NA    NA  NA
```

```
head(na.omit(roll.muhat), n=1)
```

```
##            MSFT  SP500    GWN
## Jan 1995 0.0161 0.0032 0.0284
```

Rolling Means Example Cont.



Rolling Volatility

Idea: Compute estimates of σ_i^2 and σ_i over rolling windows of length $n < T$

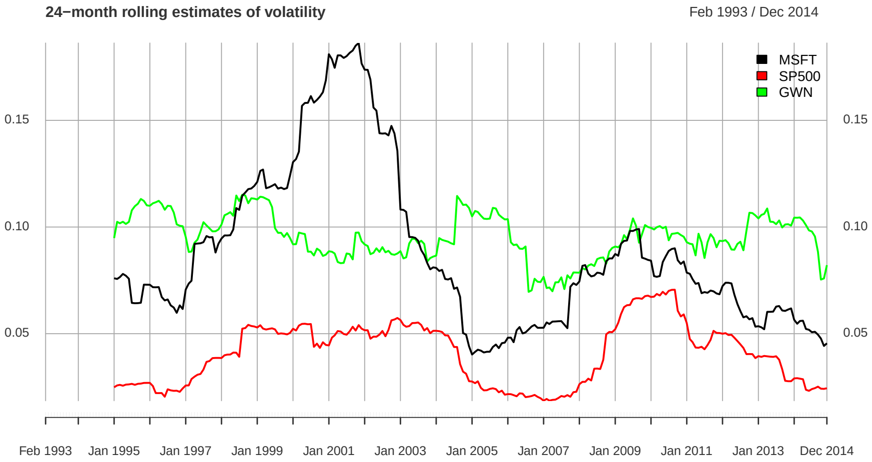
$$\hat{\sigma}_{it}^2(n) = \frac{1}{n-1}\sum_{j=0}^{n-1}(R_{it-j} - \hat{\mu}_{it}(n))^2$$

$$\hat{\sigma}_{it}(n) = \sqrt{\hat{\sigma}_{it}^2(n)}$$

for $t = n, \cdots, T$.

Rolling estimates: $\{\hat{\sigma}_{in}^2(n), \hat{\sigma}_{in+1}^2(n), \cdots, \hat{\sigma}_{iT}^2(n)\}$

Rolling Volatility Example



Rolling Covariances and Correlations

Idea: Compute estimates of σ_{jk} and ρ_{jk} over rolling windows of length $n < T$

$$\hat{\sigma}_{jk,t}(n) = \frac{1}{n-1}\sum_{i=0}^{n-1}(r_{jt-i} - \hat{\mu}_j(n))(r_{kt-i} - \hat{\mu}_k(n))$$

$$\hat{\rho}_{jk,t}(n) = \frac{\hat{\sigma}_{jk,t}(n)}{\hat{\sigma}_{jt}(n)\hat{\sigma}_{kt}(n)}$$

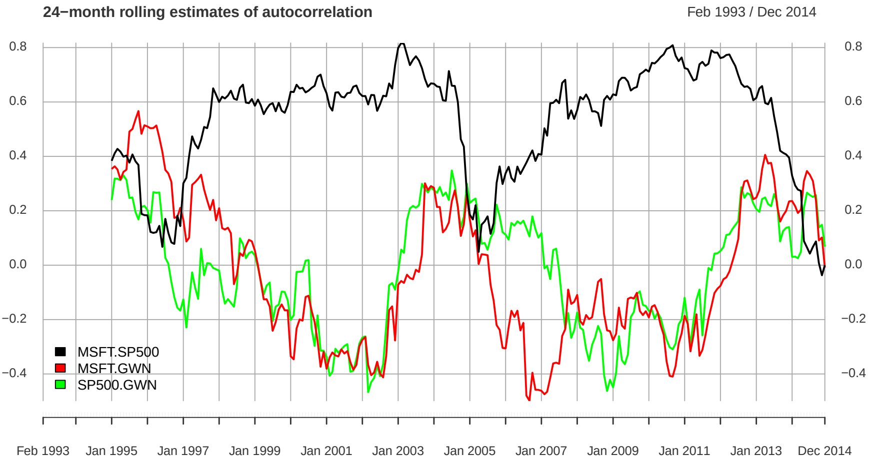
Rolling Covariances and Correlations Example

```
rhohat = function(x) {
   corhat = cor(x)
   corhat.vals = corhat[lower.tri(corhat)]
   names(corhat.vals) = c("MSFT.SP500", "MSFT.GWN", "SP500.GWN
   corhat.vals
}
```

```
roll.rhohat = rollapply(roll.data, width=24, FUN=rhohat,
   by=1, by.column=FALSE, align="right")
head(na.omit(roll.rhohat),n=3)
```

```
##          MSFT.SP500 MSFT.GWN SP500.GWN
## Jan 1995      0.384    0.354     0.239
## Feb 1995      0.411    0.363     0.318
## Mar 1995      0.428    0.352     0.318
```

Rolling Covariances and Correlations Example Cont.



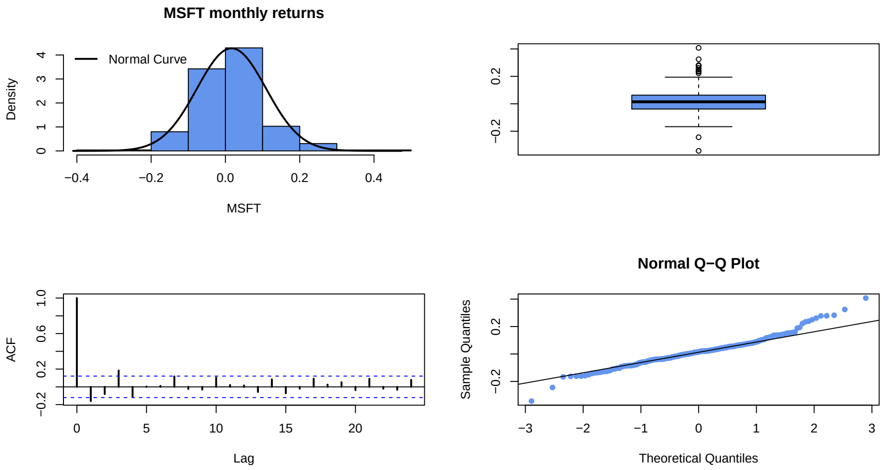
Four Panel Plot

The **IntroCompFinR** package has the function `fourPanelPlot()` which can be used to graphically summarize the stylized facts of a single time series of returns:

- Histogram with fitted normal curve overlaid
- Boxplot
- SACF
- Normal QQ-Plot

Four Panel Plot - Example

```
fourPanelPlot(msftMonthlyRetS)
```



Stylized Facts of Monthly Returns

- Returns appear to be approximately normally distributed
- Individual asset returns have higher volatility than diversified portfolios (diversification effect)
- Many assets are contemporaneously positively correlated (unusual to find negatively correlated assets)
- Asset returns are approximately uncorrelated over time (no serial correlation)

Stylized Facts of Daily Returns

- Returns are not normally distributed - distributions have fatter tails than normal
- Returns are approximately uncorrelated over time
- Returns are not independent over time
  - Squared and absolute returns are positively autocorrelated
  - Volatility appears to be serially correlated