Fitting Fundamental Factor Models: factorAnalytics vignette

Sangeetha Srinivasan

April 9, 2018

Abstract

The purpose of this vignette is to demonstrate the use of `fitFfm` and related control, analysis and plot functions in the `factorAnalytics` package.

Contents

1 Overview

1.1 Load Package

The latest version of the `factorAnalytics` package used in this vignette is hosted in the publicly available GitHub repository https://github.com/sangeeuw/factorAnalytics. There are plans for further updates to the package before its moved back to R-Forge and released on CRAN later this year.

The package can be installed from GitHub using `devtools` as follows.

```r
library(devtools)
install_github("sangeeuw/factorAnalytics")
```

```r
# load the package and its dependencies
library(factorAnalytics)
options(digits=3)
```

The focus of this vignette is on the `fitFfm` function and related methods. The original function was designed by Doug Martin and initially implemented in S-PLUS by a number of University of Washington Ph.D. students: Christopher Green, Eric Aldrich, and Yindeng Jiang. Guy Yollin ported the function to R and Yi-An Chen modified that code as part of Google Summer of Code (GSOC) 2013 . Sangeetha Srinivasan tested and expanded the functionalities and S3 methods as part of GSOC in 2014 and 2015. Doug Martin, Avinash Acharya, Lingjie Yi and Chindhanai Uthaisaad added options to fit EWMA or GARCH model for errors, enabled a market + industry and/or sector and/or country model specification, etc. as part of GSOC 2016 and 2017. Refer to the other fundamental factor model vignette by Avinash Acharya for more examples elaborating on these recent functionalities and reporting functions.

1.2 Summary of fitFfm function and related S3 methods

Here's a summary of the fit function and related S3 methods (generic accessor functions) demonstrated in this vignette:

- `fitFfm(data, asset.var, ret.var, date.var, exposure.vars, weight.var, fit.method, rob.stats, full.resid.cov, z.score, add.intercept, lag.exposures, resid.scale.type, lambda, GARCH.params, GARCH.MLE, std.return, analysis, target.vol, ...)`: Fits a fundamental factor model for one or more asset returns or excess returns using T cross-sectional regressions a.k.a. the "BARRA" approach (detailed in Grinold and Kahn (2000)), where T is the number of time periods. Available fit methods include Least squares (LS), weighted least squares (WLS), robust (rob) and weighted-robust regression (W-Rob). Options for computing residual variances include sample variance, EWMA, Robust EWMA and GARCH(1,1). An object of class "ffm" containing the fitted objects, factor exposures, factor returns, R^2, residual volatility, etc. is returned. A more detailed description is provided in Section 2.

- `coef(object, ...)`: Returns a data.frame containing the coefficients (intercept and factor exposures) for the last time period for all assets.
- `fitted(object, ...)`: Returns an "xts" data object of fitted asset returns from the factor model for all assets.
- `residuals(object, ...)`: Returns an "xts" data object of residuals from the fitted factor model for all assets.
- `fmCov(object, use, ...)`: Returns the `N x N` symmetric covariance matrix for asset returns based on the fitted factor model using exposures from the last time period.
- `fmSdDecomp(object, use, ...)`: Returns a list containing the standard deviation of asset returns based on the fitted factor model and the marginal, component and percentage component factor contributions estimated from the given sample. `"use"` specifies how missing values are to be handled.
- `fmVaRDecomp(object, factor.cov, p, type, use, ...)`: Returns a list containing the value-at-risk (VaR) for asset returns based on the fitted factor model and the estimated marginal, component and percentage component factor contributions. `factor.cov` allows for user-specified factor covariance matrix; defaults to the sample covariance of historical factor returns. `type` specifies if VaR computation should be non-parametric (sample quantile) or based on a Normal distribution. And, `"p"` specifies the confidence level.
- `fmEsDecomp(object, factor.cov, p, type, use, ...)`: Returns a list containing the expected shortfall (ES) for asset returns based on the fitted factor model and the estimated marginal, component and percentage component factor contributions. `factor.cov` allows for user-specified factor covariance matrix; defaults to the sample covariance of historical factor returns. `type` specifies if VaR computation should be non-parametric (sample quantile) or based on a Normal distribution. And, `"p"` specifies the confidence level.

- `plot(x)`: The `plot` method for class "ffm" can be used for plotting factor model characteristics of a group of assets (default) or an individual asset. The user can select the type of plot either from the menu prompt or directly via argument `which`. In case multiple plots are needed, the menu is repeated after each plot (enter 0 to exit). User can also input a numeric vector of plot options via `which`.
- `predict(object, newdata, pred.date, ...)`: The `predict` method for class "ffm" returns a vector or matrix of predicted returns for new or simulated values of the fundamental characteristics. `pred.date` allows user to choose the relevant date for the estimated factor exposures to be used in the prediction.
- `print(object, digits, ...)`: The `predict` method for class "ffm" prints the call, factor model dimension and summary statistics for the estimated factor returns, cross-sectional R^2 values and residual variances from the fitted object.
- `summary(object, ...)`: The `summary` method for class "ffm" returns an object of class `"summary.ffm"` containing the summaries of the fitted objects. Printing the factor model summary object outputs the call, estimated factor returns, R^2 and residual volatility for each time period.

A complete list of related methods is shown below.

```
methods(class="ffm")

##  [1] coef          fitted        fmCov         fmEsDecomp    fmRsq
##  [6] fmSdDecomp    fmTstats      fmVaRDecomp   plot          portEsDecomp
## [11] portSdDecomp  portVaRDecomp portVolDecomp predict       print
## [16] repRisk       residuals     riskDecomp    summary
## see '?methods' for accessing help and source code
```

1.3 Data

The following examples primarily use the `Stock.df` dataset. It contains fundamental and monthly return data for 447 stocks listed on the NYSE over a 8-year period. The dataset is balanced, i.e., every asset has a complete set of observations for all variables in each time period.

The following queries help understand key aspects of the dataset:

```r
# load the dataset into the environment
data(Stock.df)
# get a list of the variable names
colnames(stock)
```

```
##  [1] "DATE"              "RETURN"            "TICKER"
##  [4] "PRICE"             "VOLUME"            "SHARES.OUT"
##  [7] "MARKET.EQUITY"     "LTDEBT"            "NET.SALES"
## [10] "COMMON.EQUITY"     "NET.INCOME"        "STOCKHOLDERS.EQUITY"
## [13] "LOG.MARKETCAP"     "LOG.PRICE"         "BOOK2MARKET"
## [16] "GICS"              "GICS.INDUSTRY"     "GICS.SECTOR"
```

```r
# time period covered in the data
range(stock[,"DATE"])
```

```
## [1] "1996-02-29" "2003-12-31"
```

```r
# number of stocks
length(unique(stock[,"TICKER"]))
```

```
## [1] 447
```

```r
# count stocks by GICS sector as of the last time period
stocklist<-subset(stock,DATE=="2003-12-31")
table(stocklist$GICS.SECTOR)
```

```
##
##     Consumer Discretionary           Consumer Staples
##                         86                         30
##                     Energy                 Financials
##                         17                         55
##                Health Care                Industrials
##                         35                         89
##     Information Technology                  Materials
##                         57                         32
## Telecommunication Services                  Utilities
##                          6                         40
```

2 Fitting a fundamental factor model

A fundamental factor model uses observed cross-sectional asset characteristics such as dividend yield, earnings yield, book-to-market ratio, market capitalization, sector or industry classification, price volatility, price momemtum, leverage, etc. to determine common risk factors that contribute to asset returns. Chapter 15 from Zivot and Jia-hui (2006) serves as a good reference for a description of the different multi-factor models, estimation methods and relevant examples using S-PLUS.

There are 2 main approaches to estimating the fundamental factor model - the "BARRA" approach (detailed in Grinold and Kahn (2000)) and the "Fama-French" approach (introduced in Fama and French (1992)). In the "BARRA" approach, the obsered fundamental attributes are the factor betas and the unknown factor returns are estimated via cross-sectional regressions for each time period. Due to cross-sectional heteroskedasticity of asset returns, ordinary least squares (OLS) estimation of the factor returns is inefficient. So weighted least squares regression is performed as a second step to get efficient estimates, with the inverse of the estimated residual variances or market cap used as weights. In the "Fama-French" approach, the factor returns are the observed returns of a hypothetical hedge portfolio that's long/short the top/bottom quintile of stocks for a given attribute (ex: market cap for the size factor). After the factor returns are computed for each characterictic, each asset's factor exposures are estimated via a time series regression. `fitFfm` described in this vignette uses the "BARRA" approach.

Let's take a look at the arguments for `fitFfm`.

```
args(fitFfm)

## function (data, asset.var, ret.var, date.var, exposure.vars,
##     weight.var = NULL, fit.method = c("LS", "WLS", "Rob", "W-Rob"),
##     rob.stats = FALSE, full.resid.cov = FALSE, z.score = c("none",
##         "crossSection", "timeSeries"), add.intercept = FALSE,
##     lag.exposures = TRUE, resid.scale.type = c("stdDev", "EWMA",
##         "robEWMA", "GARCH"), GARCH.params = list(omega = 0.09,
##         alpha = 0.1, beta = 0.81), lambda = 0.9, GARCH.MLE = FALSE,
##     std.return = FALSE, analysis = c("none", "ISM", "NEW"), target.vol = 0.06,
##     ...)
## NULL
```

The default model fitting method is ordinary least squares (LS) regression, with the option to choose robust regression (Rob), weighted least squares (WLS) or weighted robust regression

(W-Rob). The different model fitting options are demonstrated in the following sections. If weighted regression (WLS or W-Rob) is chosen, inverse of the residual variances are used as weights. `resid.scale.type` allows the user to choose the method for computing residual variances - sample variance, EWMA, Robust EWMA and GARCH(1,1).

`z.score` provides the option to standardize factor exposures cross-sectionally across assets or across time periods. `weight.var` allows the user to give higher weight to some assets when estimating factor exposures; for example using the market cap of stocks as their weights. `add.intercept` gives the option to add an intercept term for fitting a Market + Sector or a Market + Sector + Country model. These models can simultaneously include other style factors. `lag.exposures` gives the option to use the factor exposures from the previous time period to estimate factor returns for the current period. `full.resid.cov` provides the option to choose between a diagonal vs. full residual covariance matrix. And, `rob.stats` allows for robust estimates of covariance, correlation, location and univariate scale.

These and other control parameters are demonstrated in the following sections.

2.1 Single Factor Model

Here's an example of a single factor model using the book-to-market ratio, a proxy for the value factor, as the explanatory variable for the returns of 447 stocks in the dataset.

```r
# Single Factor Model
fit.single <- fitFfm(data=stock, asset.var="TICKER", ret.var="RETURN",
                     date.var="DATE", exposure.vars="BOOK2MARKET")
```

The resulting object, `fit.single`, has the following attributes.

```r
class(fit.single)

## [1] "ffm"

names(fit.single)

##  [1] "factor.fit"      "beta"            "factor.returns"
##  [4] "residuals"       "r2"              "factor.cov"
##  [7] "g.cov"           "resid.cov"       "return.cov"
## [10] "restriction.mat" "resid.var"       "call"
## [13] "data"            "date.var"        "ret.var"
## [16] "asset.var"       "exposure.vars"   "weight.var"
```

```
## [19] "fit.method"      "asset.names"     "factor.names"
## [22] "time.periods"    "activeWeights"   "activeReturns"
## [25] "IR"
```

The component `factor.fit` contains a list of "lm" or "lmRob" objects, one for each time period. The fitted objects is of class "lm" if `fit.method="LS" or "WLS"`, or class "lmRob", if `fit.method="Rob" or "W-Rob"`. The component `factor.returns` contains the estimated factor returns and `beta` contains the factor exposures from the last time period. While, `r2` and `resid.var` denote the regression R^2 and estimated residual variance respectively. The estimated covariance matrices of factor returns, residuals and asset returns are given by `factor.cov`, `resid.cov` and `return.cov` respectively. The remaining components contain the input choices and the data.

The `print` method displays a summary of the T cross-sectional regressions, where T is the number of time periods.

```
# print the fitted "ffm" object
fit.single

##
## Call:
## fitFfm(data = stock, asset.var = "TICKER", ret.var = "RETURN",
##     date.var = "DATE", exposure.vars = "BOOK2MARKET")
##
## Model dimensions:
## Factors  Assets Periods
##       1     447      94
##
## Factor returns across periods:
##   BOOK2MARKET
##  Min.   :-0.0332
##  1st Qu.:-0.0053
##  Median : 0.0045
##  Mean   : 0.0048
##  3rd Qu.: 0.0139
##  Max.   : 0.0446
##
## R-squared values across periods:
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0010  0.0043  0.0076  0.0122  0.0475
##
## Residual Variances across assets:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0036  0.0141  0.0209  0.0294  0.0347  0.1590
```

Figure 1 shows a scatter plot of residuals for the 1st 6 stocks in the last time period, including histograms, density overlays, correlations and significance stars. (A detailed list of plot options is provided later in Section 4.) Note the high residual correlation between MSFT and ORCL; this might be due to their exposure to other omitted factors such as a sector/industry risk factor for "Software & Services". The next section demonstrates fitting an industry/sector factor model for these stocks.

```
# plot residual correlations for the single factor model
# default is to plot the 1st 6 assets
plot(fit.single, which=6, f.sub=1)
```

```
# GICS industry/sector classification (1st 6 stocks; penultimate time period)
subset(stock,DATE=="2003-11-28")[1:6,c("TICKER","GICS.INDUSTRY","GICS.SECTOR")]

##     TICKER                      GICS.INDUSTRY            GICS.SECTOR
## 94    JJSF         Food, Beverage & Tobacco       Consumer Staples
## 189   PLXS Technology Hardware & Equipment Information Technology
## 284   SUNW Technology Hardware & Equipment Information Technology
## 379   ORCL               Software & Services Information Technology
## 474   MSFT               Software & Services Information Technology
## 569    SDS               Software & Services Information Technology
```

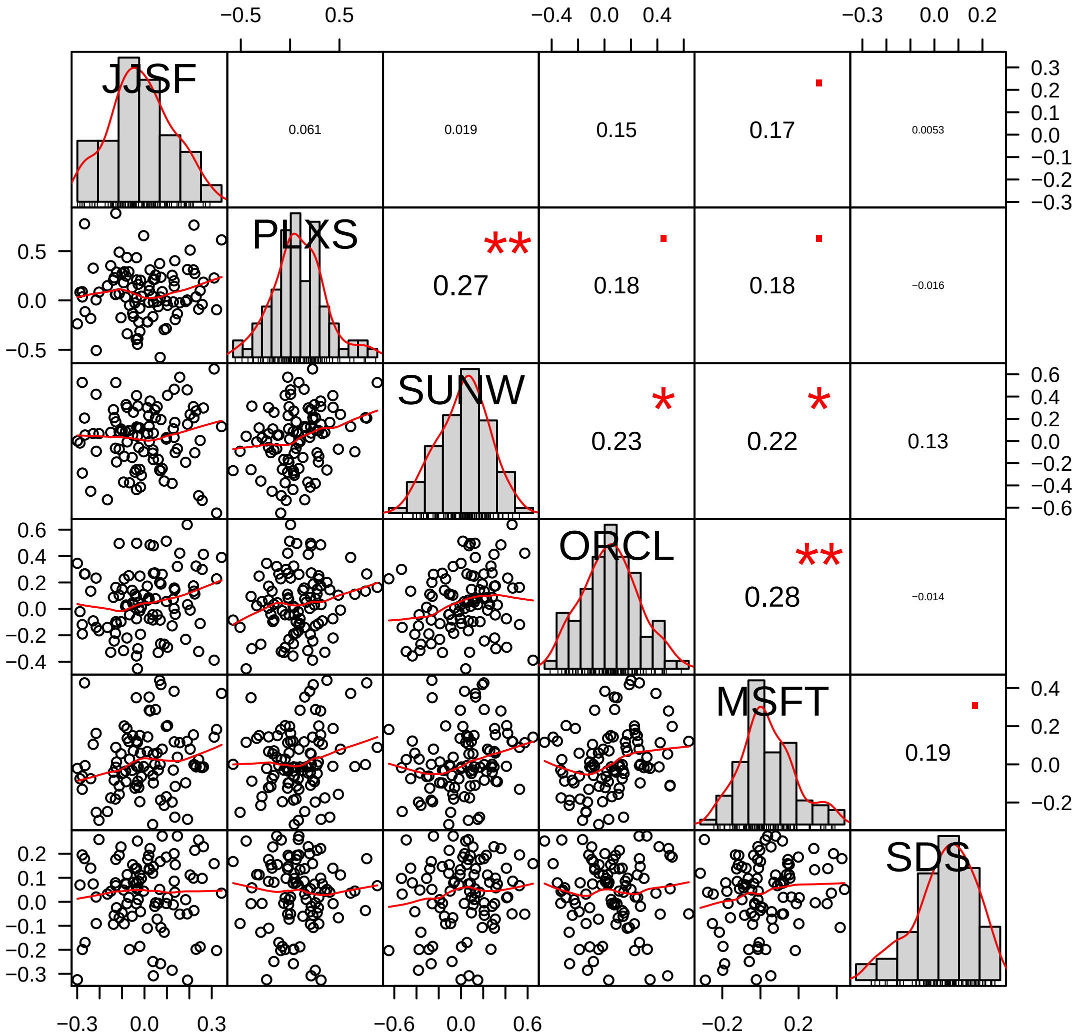

Figure 1: Single factor model: Residual correlations

2.2 BARRA-type Industry Factor Model

A BARRA-type industry (sector) factor model is a fundamental factor model with multiple factors. Here is a demonstration using the 447 NYSE stocks in our dataset; where the 10 mutually exclusive GICS sector classifications are the 10 factors. The factor exposures will be dummy variables that indicate if a given stock belongs to a particular sector or not. Mutually exclusive sectors means that each stock belongs to a unique sector in any given time period. Notice that the average R^2 from the sector model is significantly higher (and average residual correlations are lower) than the single factor model.

```
# Sector Factor Model
fit.sector <- fitFfm(data=stock, asset.var="TICKER", ret.var="RETURN",
                     date.var="DATE", exposure.vars="GICS.SECTOR")
# compare r2: single factor vs. sector model
summary(fit.single$r2)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0010  0.0043  0.0076  0.0122  0.0475

summary(fit.sector$r2)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.023   0.060   0.113   0.137   0.195   0.519

# compare avg. non-diagonal correlations: single factor vs. sector model
mean(cor(residuals(fit.single))[cor(residuals(fit.single))!=1])

## [1] 0.0923

mean(cor(residuals(fit.sector))[cor(residuals(fit.sector))!=1])

## [1] -0.00121
```

Let's take a look at the fitted factor model from the last period in the data. We observe that Energy, Materials and Telecomm sectors had particularly strong returns, with estimated factor returns over 10% for that month[1].

[1]Energy stocks rebounded in 2003 from the beating they took in 2002 following the Enron scandal. Telecomm stocks benefited from the increased spending by companies investing in internet-based phone systems during this period.

```r
# print the summary from the last period's fit
num.periods <- length(fit.sector$time.periods)
summary(fit.sector$factor.fit[[num.periods]])
```

```
##
## Call:
## FUN(formula = ..1, data = data[x, , drop = FALSE], na.action = ..3,
##     contrasts = ..2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.3984 -0.0806 -0.0067  0.0780  0.5362
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## Consumer Discretionary        0.0124     0.0154    0.80   0.4236
## Consumer Staples              0.0480     0.0261    1.84   0.0666 .
## Energy                        0.1131     0.0347    3.26   0.0012 **
## Financials                    0.0466     0.0193    2.41   0.0162 *
## Health Care                   0.0358     0.0242    1.48   0.1398
## Industrials                   0.0415     0.0152    2.74   0.0064 **
## Information Technology        0.0339     0.0190    1.79   0.0744 .
## Materials                     0.1146     0.0253    4.53  7.6e-06 ***
## Telecommunication Services    0.1025     0.0584    1.76   0.0799 .
## Utilities                     0.0684     0.0226    3.02   0.0027 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.143 on 437 degrees of freedom
## Multiple R-squared:  0.131,Adjusted R-squared:  0.112
## F-statistic: 6.61 on 10 and 437 DF,  p-value: 1.44e-09
```

Figure 2 shows the distribution of estimated monthly sector returns (from 1996 - 2003) in descending order of their mean. We find that the "Information Technology" sector had the highest average return (perhaps not suprising, given that the dataset covers the dot-com bubble).

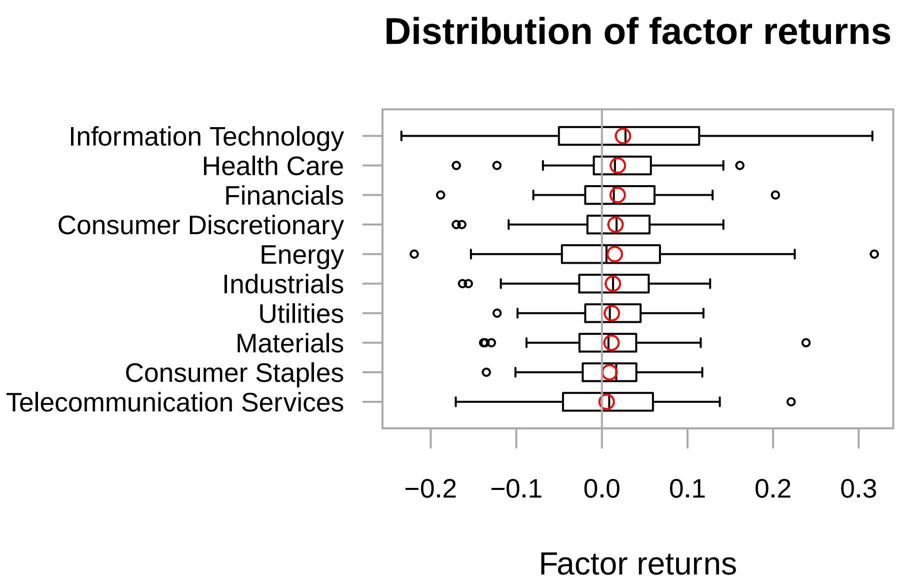

Figure 2: Sector model: Distribution of factor returns sorted by mean

```
# plot distribution of factor returns by sector sorted by means
plot(fit.sector, which=1, colorset="black", f.sub=1:10, lwd=1, sort.by="mean")
```

An extension of the above sector model is to isolate the market effect through the use of an intercept term and reparametrizing the sector exposures so that they are measured relative to the common market factor. Here, the intercept is interpreted as the return to the market factor (sum of all sectors), while the other factors are excess returns for the sector over the market. The methodology behind this model was introduced in the context of a common country effect in Heston and Rouwenhorst (1995) and also detailed in Menchero (2010). In `fitFfm` the market + sector model can be opted via the parameter `add.intercept` as shown below.

```
# Market + Sector Factor Model
fit.mkt.sector <- fitFfm(data=stock, asset.var="TICKER", ret.var="RETURN",
                         date.var="DATE", exposure.vars="GICS.SECTOR",
                         add.intercept=TRUE)

# coefficients (factor exposures) for first 10 assets
t(coef(fit.mkt.sector)[1:10,])
```

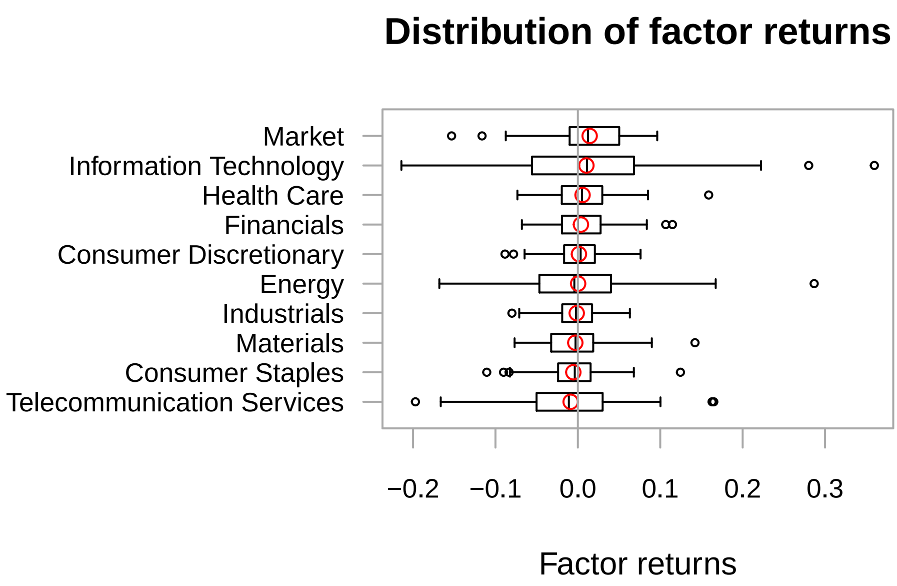

Figure 3: Market + Sector model: Distribution of factor returns sorted by mean

```
##                            JJSF PLXS SUNW ORCL MSFT SDS TROW HON EMC XRIT
## Market                        1    1    1    1    1   1    1   1   1    1
## Consumer Discretionary        0    0    0    0    0   0    0   0   0    0
## Consumer Staples              1    0    0    0    0   0    0   0   0    0
## Energy                        0    0    0    0    0   0    0   0   0    0
## Financials                    0    0    0    0    0   0    1   0   0    0
## Health Care                   0    0    0    0    0   0    0   0   0    0
## Industrials                   0    0    0    0    0   0    0   1   0    0
## Information Technology        0    1    1    1    1   1    0   0   1    1
## Materials                     0    0    0    0    0   0    0   0   0    0
## Telecommunication Services    0    0    0    0    0   0    0   0   0    0
## Utilities                     0    0    0    0    0   0    0   0   0    0
```

```
# plot distribution of factor returns by sector sorted by means
plot(fit.mkt.sector, which=1, colorset="black", f.sub=1:10, lwd=1, sort.by="mean")
```

The reparametrization of the market factor hasn't changed the order of sectors by mean factor return. The reader can verify that R^2 and other fit statistics haven't changed either.

2.3 Multi-factor Model with Sector and Style Characteristics

A fundamental factor model can simultaneously include both quantitative style factors, such as size (market cap), value (book-to-price ratio), price momentum etc., as well as sector/industry classifications. The next example demonstrates fitting a multi-factor model including 2 style factors, size and value, in addition to the sector model. Note that the adjusted-R^2 has improved.

```
# Market + Sector Factor Model
fit.style.sector <- fitFfm(data=stock, asset.var="TICKER", ret.var="RETURN",
  date.var="DATE", exposure.vars=c("GICS.SECTOR","LOG.MARKETCAP","BOOK2MARKET"))

# check if average adjusted R-squared improved vs. pure sector model
# adjusted r2 = 1 - ((n-1)*(1-r2)/(n-p-1))
print(adj.r2_style.sector <- 1-((447-1)*(1-mean(fit.style.sector$r2))/(447-12-1)))

## [1] 0.126

print(adj.r2_sector <- 1-((447-1)*(1-mean(fit.sector$r2))/(447-10-1)))

## [1] 0.117
```

Figures 4, 5 and 6 below show some properties (such as last period's factor exposures, time series of R^2 values and factor returns) of the fitted factor model. Figures 7 and 8 compares the kernel density of residuals for "MSFT" vs. normal and skew-t fits.

```
plot(fit.style.sector, which=2, f.sub=1:12, a.sub=1:10)
```

```
plot(fit.style.sector, which=4, las=2)
```

```
plot(fit.style.sector, which=12, f.sub=1:3, las=2, legend.loc="bottom", cex.legend=0.75)
```

```
plot(fit.style.sector, plot.single=TRUE, which=10, asset.name="MSFT")
```

```
plot(fit.style.sector, plot.single=TRUE, which=11, asset.name="MSFT")
```

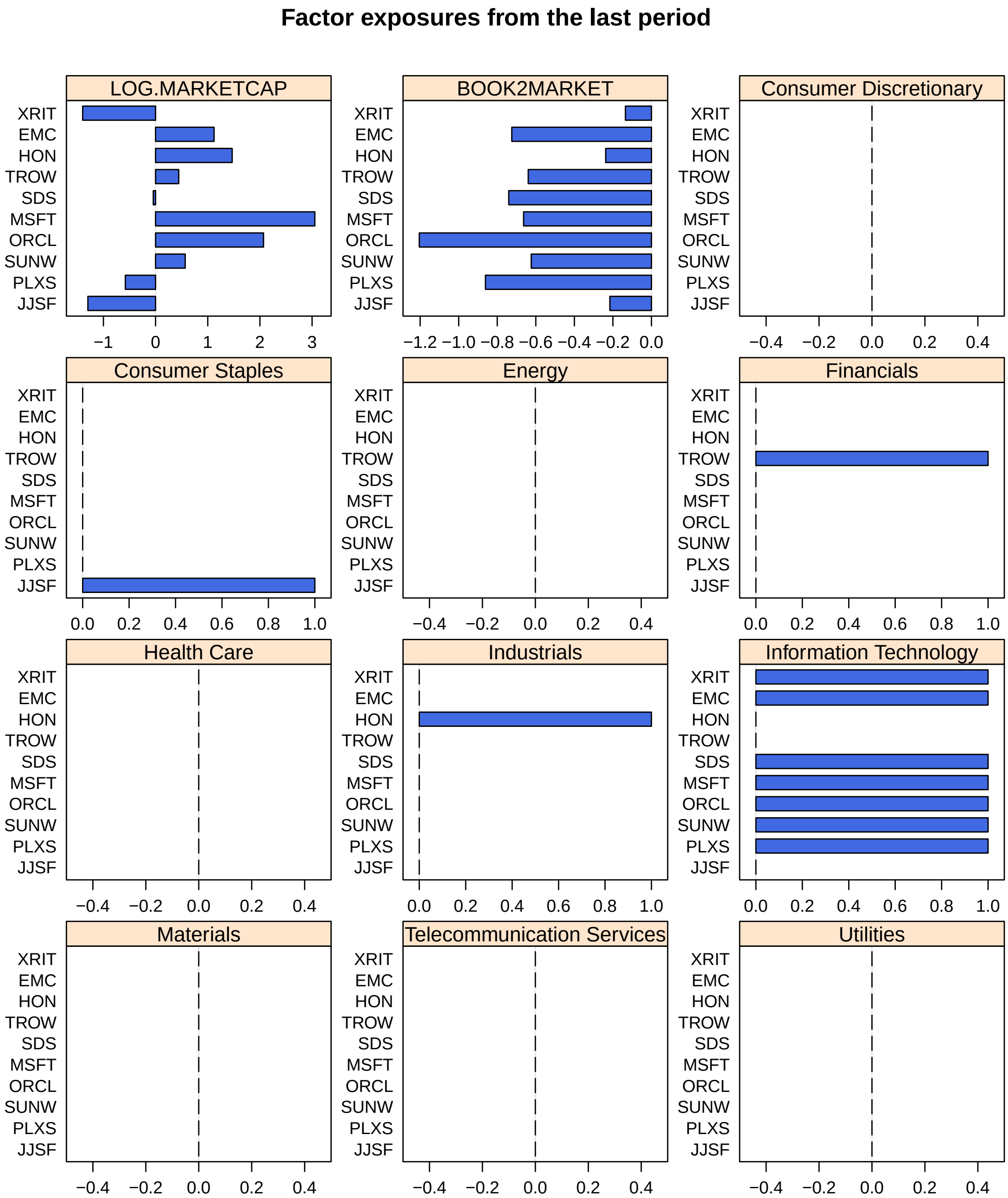

Figure 4: Factor exposures from the last time period (1st 10 assets)

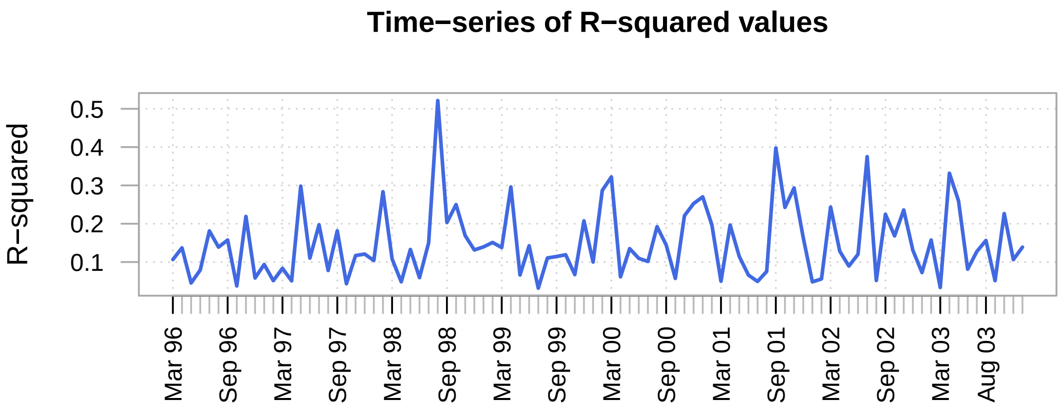

Figure 5: Time series of R-squared values

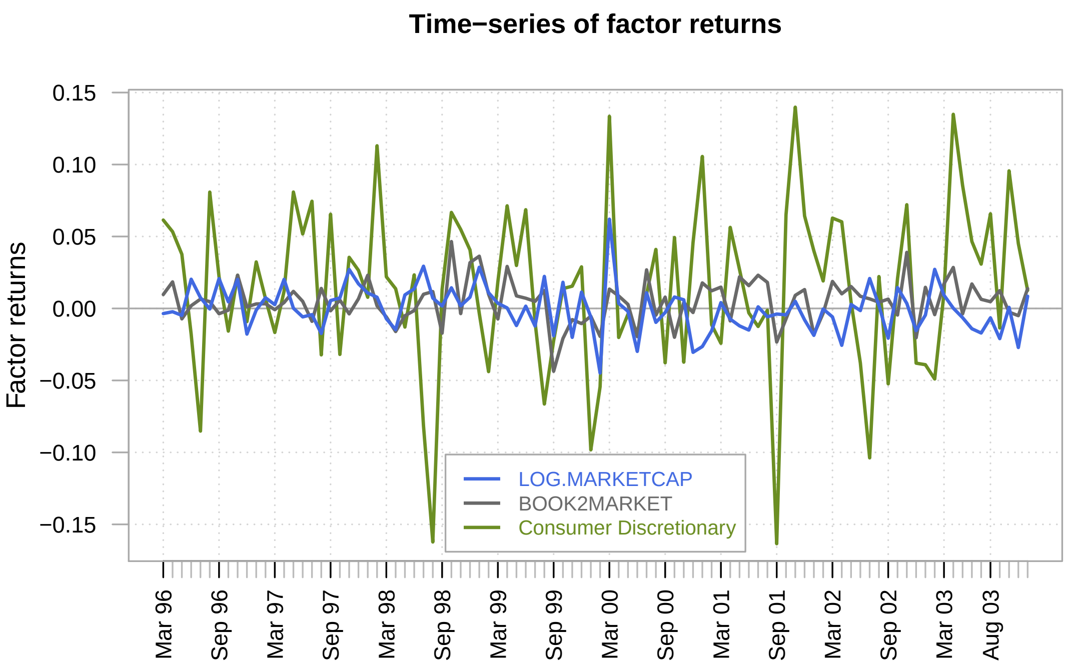

Figure 6: Time series of factor returns (2 style and 1 sector factors)

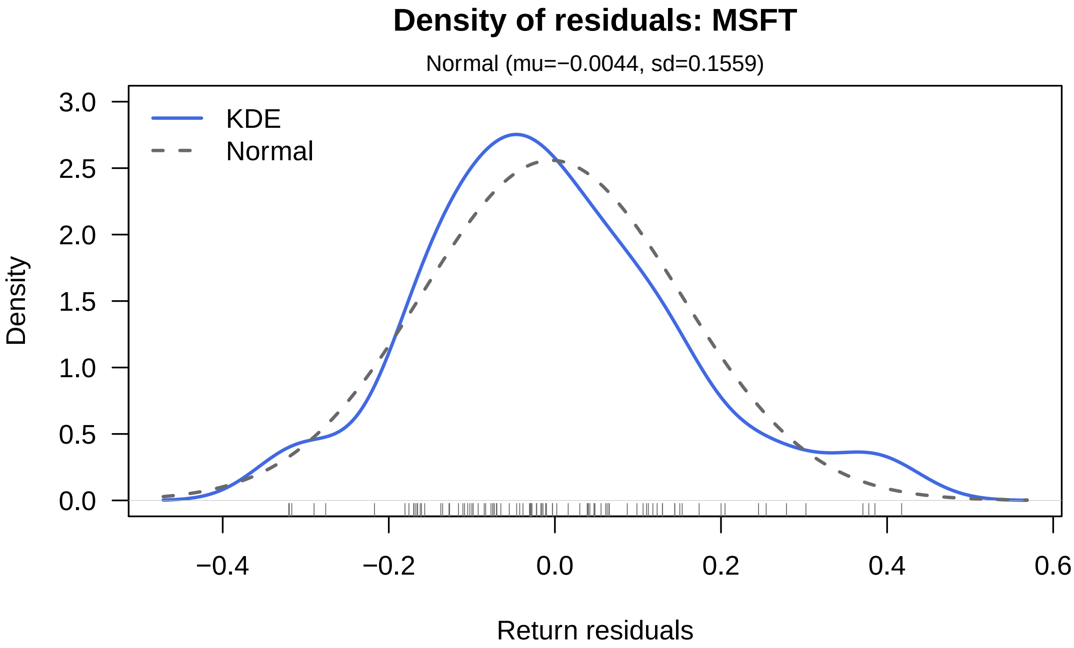

Figure 7: Non-parametric density of residuals with normal overlaid for MSFT

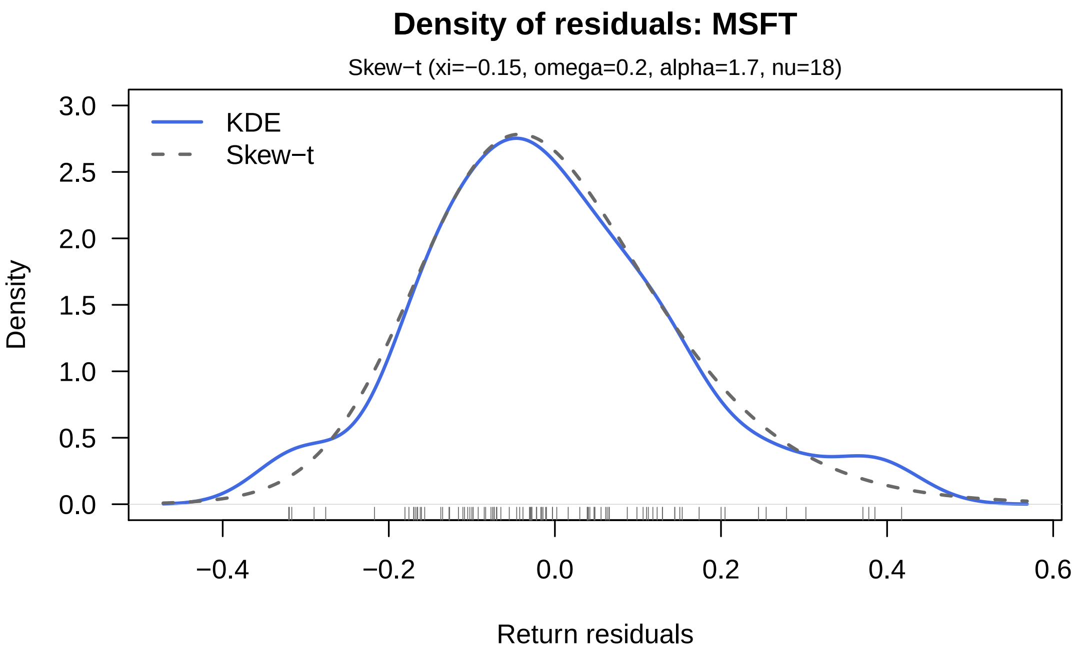

Figure 8: Non-parametric density of residuals with skew-t overlaid for MSFT

3 Factor Model Covariance & Risk Decomposition

3.1 Factor model covariance

Following Zivot and Jia-hui (2006), $R_{i,t}$, the return on asset i ($i = 1, ..., N$) at time t ($t = 1, ..., T$), is fitted with a factor model of the form,

$$R_{i,t} = \alpha_i + \boldsymbol{\beta}_i' \, \mathbf{f_t} + \epsilon_{i,t} \tag{1}$$

where, α_i is the intercept, $\mathbf{f_t}$ is a $K \times 1$ vector of factor returns at time t, $\boldsymbol{\beta}_i$ is a $K \times 1$ vector of factor exposures for asset i and the error terms $\epsilon_{i,t}$ are serially uncorrelated across time and contemporaneously uncorrelated across assets so that $\epsilon_{i,t} \sim iid(0, \sigma_i^2)$. Thus, the variance of asset i's return is given by

$$var(R_{i,t}) = \boldsymbol{\beta}_i' \, var(\mathbf{f_t}) \, \boldsymbol{\beta}_i + \sigma_i^2 \tag{2}$$

And, the $N \times N$ covariance matrix of asset returns is

$$var(\mathbf{R}) = \boldsymbol{\Omega} = \mathbf{B} \, var(\mathbf{F}) \, \mathbf{B}' + \mathbf{D} \tag{3}$$

where, R is the $N \times T$ matrix of asset returns, B is the $N \times K$ matrix of factor betas, $\mathbf{F}$ is a $K \times T$ matrix of factor returns and D is a diagonal matrix with σ_i^2 along the diagonal.

`fmCov()` computes the factor model covariance from a fitted factor model. The covariance of factor returns is the sample covariance matrix by default, but the option exists for the user to specify their own. Options for handling missing observations include "pairwise.complete.obs" (default), "everything", "all.obs", "complete.obs" and "na.or.complete".

```
fmCov(fit.style.sector)[1:6,1:6]

##           JJSF      PLXS      SUNW     ORCL     MSFT       SDS
## JJSF  0.031186  0.003423 -0.004958 -0.01453 -0.02224 -0.000505
## PLXS  0.003423  0.068127  0.000755 -0.00183 -0.00491  0.002341
## SUNW -0.004958  0.000755  0.057259  0.01107  0.01489  0.002837
## ORCL -0.014532 -0.001825  0.011074  0.07685  0.03911  0.004194
## MSFT -0.022237 -0.004907  0.014887  0.03911  0.08100  0.004351
## SDS  -0.000505  0.002341  0.002837  0.00419  0.00435  0.030825

# factor model return correlation plot (for 1st 6 assets by default)
plot(fit.style.sector, which=8)
```

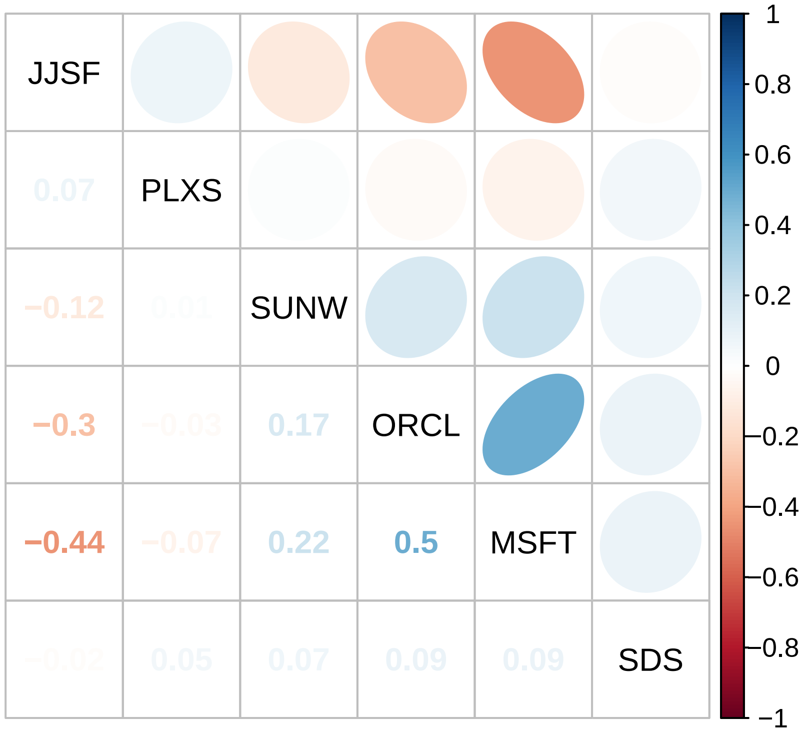

Figure 9: Factor model return correlation (pairwise complete obs)

3.2 Standard deviation decomposition

Given the factor model in equation 1, the standard deviation of the asset i's return can be decomposed as follows (based on Meucci (2007)):

$$
\begin{aligned}
R_{i,t} &= \alpha_i + \boldsymbol{\beta}_i' \, \mathbf{f_t} + \epsilon_{i,t} && (4)\\
&= \boldsymbol{\beta}_i^{*'} \, \mathbf{f_t^*} && (5)
\end{aligned}
$$

where, $\boldsymbol{\beta}_i^{*'} = (\boldsymbol{\beta}_i' \, \sigma_i)$ and $\mathbf{f_t^{*'}} = (\mathbf{f_t'} \, z_t)$, with $z_t \sim iid(0,1)$ and σ_i is asset i's residual standard deviation.

By Euler's theorem, the standard deviation of asset i's return is:

$$
Sd.fm_i = \sum_{k=1}^{K+1} cSd_{i,k} = \sum_{k=1}^{K+1} \beta_{i,k}^* \, mSd_{i,k} \tag{6}
$$

where, summation is across the K factors and the residual, $\mathbf{cSd_i}$ and $\mathbf{mSd_i}$ are the component and marginal contributions to $Sd.fm_i$ respectively. Computing $Sd.fm_i$ and $\mathbf{mSd_i}$ is straight forward. The formulas are given below and details are in Meucci (2007). The covariance term is approximated by the sample covariance and $\odot$ represents element-wise multiplication.

$$
\begin{aligned}
Sd.fm_i &= \sqrt{\boldsymbol{\beta}_i^{*'} \, cov(\mathbf{F}^*) \, \boldsymbol{\beta}_i^*} && (7)\\
\mathbf{mSd_i} &= \frac{cov(\mathbf{F}^*) \, \boldsymbol{\beta}_i^*}{Sd.fm_i} && (8)\\
\mathbf{cSd_i} &= \boldsymbol{\beta}_i^* \, \odot \, \mathbf{mSd_i} && (9)
\end{aligned}
$$

`fmSdDecomp` performs this decomposition for all assets in the given factor model fit object as shown below. The total standard deviation and component, marginal and percentage component contributions for each asset are returned.

```
decomp <- fmSdDecomp(fit.style.sector)
names(decomp)

## [1] "Sd.fm" "mSd"   "cSd"   "pcSd"

# get the factor model standard deviation for 1st 6 assets
decomp$Sd.fm[1:6]

##  JJSF  PLXS  SUNW  ORCL  MSFT   SDS
## 0.155 0.279 0.255 0.249 0.197 0.203

# get the component contributions to Sd for (1st 6 assets, relevant factors)
decomp$cSd[1:6, c(1,2,4,9)]
```

```
##      LOG.MARKETCAP BOOK2MARKET Consumer Staples Information Technology
## JJSF      2.36e-03   -7.16e-05           0.0128                 0.0000
## PLXS      6.75e-04    1.88e-04           0.0000                 0.0469
## SUNW     -5.62e-05   -1.38e-04           0.0000                 0.0509
## ORCL      2.62e-03   -8.79e-05           0.0000                 0.0509
## MSFT      9.18e-03   -6.46e-04           0.0000                 0.0643
## SDS       3.99e-05    1.46e-05           0.0000                 0.0641

# get the marginal factor contributions to Sd (1st 6 assets, relevant factors)
decomp$mSd[1:6, c(1,2,4,9)]

##      LOG.MARKETCAP BOOK2MARKET Consumer Staples Information Technology
## JJSF     -0.001820    3.32e-04          0.01282                0.00796
## PLXS     -0.001165   -2.19e-04          0.00328                0.04695
## SUNW     -0.000099    2.21e-04          0.00399                0.05090
## ORCL      0.001265    7.30e-05          0.00401                0.05093
## MSFT      0.003006    9.74e-04          0.00583                0.06433
## SDS      -0.000907   -1.97e-05          0.00475                0.06414

# get the % component contributions to Sd (1st 6 assets, relevant factors)
decomp$pcSd[1:6, c(1,2,4,9)]

##      LOG.MARKETCAP BOOK2MARKET Consumer Staples Information Technology
## JJSF        1.5283    -0.04632              8.3                    0.0
## PLXS        0.2423     0.06760              0.0                   16.8
## SUNW       -0.0221    -0.05418              0.0                   20.0
## ORCL        1.0508    -0.03531              0.0                   20.5
## MSFT        4.6697    -0.32874              0.0                   32.7
## SDS         0.0196     0.00718              0.0                   31.6

# plot the % component contributions to Sd (1st 6 assets, relevant factors)
plot(fit.style.sector, which=9, f.sub=c(1,2,4,9))
```

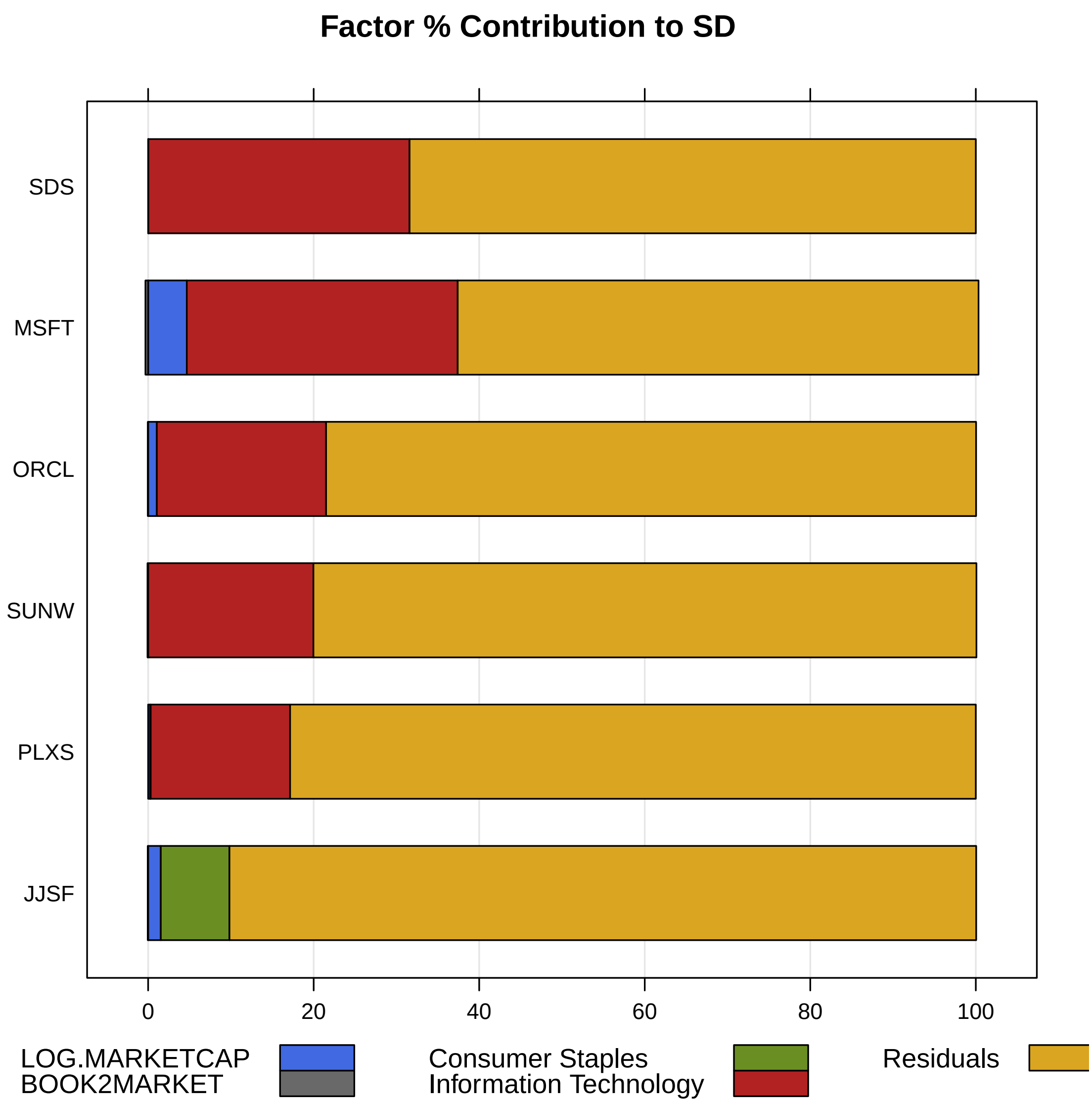

Figure 10: Percentage factor contribution to SD

3.3 Value-at-Risk decomposition

The VaR version of equation 6 is given below. By Euler's theorem, the value-at-risk of asset i's return is:

$$VaR.fm_i = \sum_{k=1}^{K+1} cVaR_{i,k} = \sum_{k=1}^{K+1} \beta^*_{i,k} \, mVaR_{i,k} \tag{10}$$

The marginal contribution to $VaR.fm$ is defined as the expectation of $F.star$, conditional on the loss being equal to $VaR.fm$. This is approximated as described in Epperlein and Smillie (2006) using a triangular smoothing kernel. `type` gives the option to estimate $VaR.fm$ non-parametrically using the sample quantile (default) or assuming a normal distribution.

`fmVaRDecomp` performs this decomposition for all assets in the given factor model fit object as shown below. The total VaR and component, marginal and percentage component contributions for each asset are returned.

```
decomp1 <- fmVaRDecomp(fit.style.sector, type="normal", p=0.10)
names(decomp1)

## [1] "VaR.fm"     "n.exceed"   "idx.exceed" "mVaR"       "cVaR"
## [6] "pcVaR"

# get the factor model value-at-risk for 1st 6 assets
decomp1$VaR.fm[1:6]

##   JJSF   PLXS   SUNW   ORCL   MSFT    SDS
## -0.190 -0.335 -0.304 -0.299 -0.229 -0.238

# print the number of VaR exceedences for 1st 6 assets
decomp1$n.exceed[1:6]

## JJSF PLXS SUNW ORCL MSFT  SDS
##   11    6   12    9    7    6

# plot the % component contributions to VaR (1st 6 assets, relevant factors)
plot(fit.style.sector, which=11, f.sub=c(1,2,4,9))
```

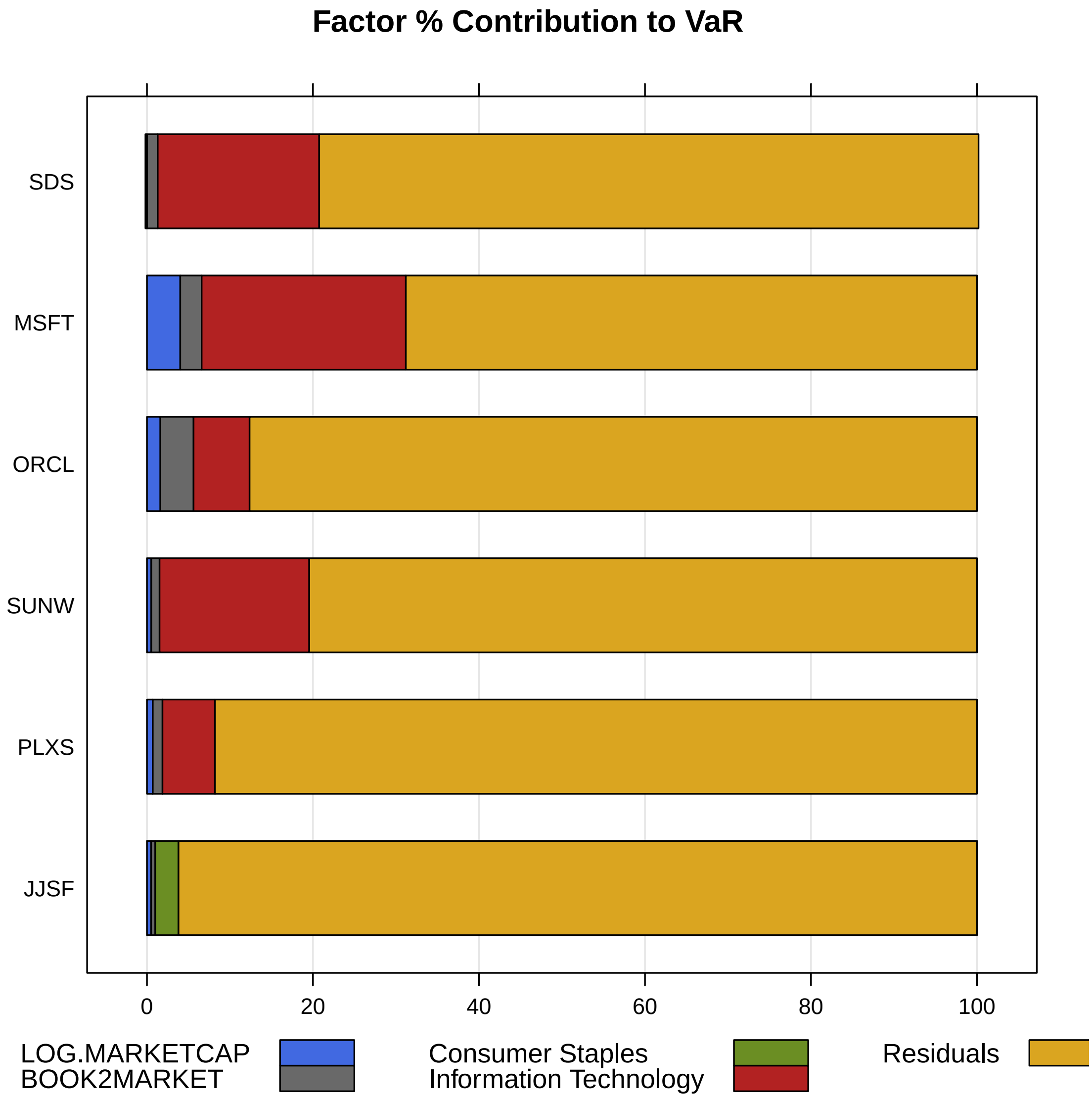

Figure 11: Percentage factor contribution to VaR

3.4 Expected Shortfall decomposition

The Expected Shortfall (ES) version of equation 6 is given below. By Euler's theorem, the expected shortfall of asset i's return is:

$$ES.fm_i = \sum_{k=1}^{K+1} cES_{i,k} = \sum_{k=1}^{K+1} \beta^*_{i,k}\, mES_{i,k} \tag{11}$$

The marginal contribution to $ES.fm$ is defined as the expectation of $F.star$, conditional on the loss being less than or equal to $ES.fm$. This is estimated as a sample average of the observations in that data window. Once again, input variable `type` gives the option to estimate $ES.fm$ non-parametrically using the sample quantile (default) or assuming a normal distribution.

`fmEsDecomp` performs this decomposition for all assets in the given factor model fit object as shown below. The total ES and component, marginal and percentage component contributions for each asset are returned.

```
decomp2 <- fmEsDecomp(fit.style.sector, type="normal")
names(decomp2)

## [1] "ES.fm" "mES"   "cES"   "pcES"

# get the factor model expected shortfall for 1st 6 assets
decomp2$ES.fm[1:6]

##   JJSF   PLXS   SUNW   ORCL   MSFT    SDS
## -0.327 -0.597 -0.549 -0.534 -0.428 -0.442

# get the component contributions to ES for (1st 6 assets, relevant factors)
decomp2$cES[1:6, c(1,2,4,9)]

##      LOG.MARKETCAP BOOK2MARKET Consumer Staples Information Technology
## JJSF     -4.87e-03     0.00115          -0.0357                  0.000
## PLXS     -1.39e-03     0.00364           0.0000                 -0.122
## SUNW      1.15e-04     0.00351           0.0000                 -0.133
## ORCL     -5.41e-03     0.00568           0.0000                 -0.130
## MSFT     -1.91e-02     0.00409           0.0000                 -0.156
## SDS      -8.35e-05     0.00255           0.0000                 -0.153

# plot the % component contributions to ES (1st 6 assets, relevant factors)
plot(fit.style.sector, which=10, f.sub=c(1,2,4,9))
```

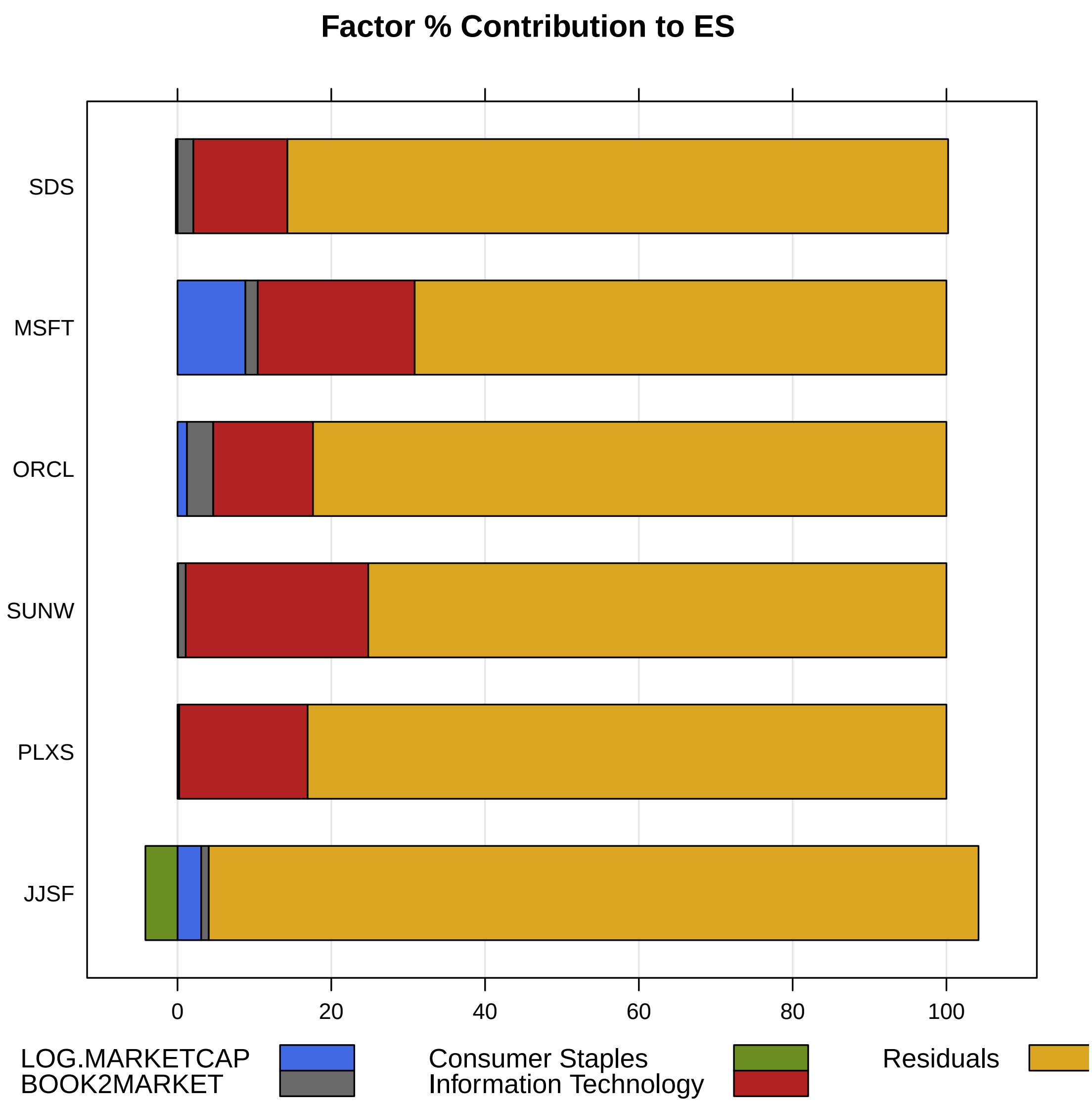

Figure 12: Percentage factor contribution to ES

4 Plot

Some types of group plots (Figures 1-6,9-12) and individual asset plots (Figure 7,8) have already been demonstrated. Let's take a look at all available arguments for plotting a "ffm" object.

```
## S3 method for class "ffm"
plot (x, which=NULL, f.sub=1:2, a.sub=1:6, plot.single=FALSE, asset.name,
      colorset=c("royalblue","dimgray","olivedrab","firebrick", "goldenrod",
                 "mediumorchid","deepskyblue","chocolate","darkslategray"),
      legend.loc="topleft", las=1, lwd=2, maxlag=15, ...)
```

4.1 Group plots

This is the default option for plotting. Simply running `plot(fit)`, where `fit` is any "ffm" object, will bring up a menu (shown below) for group plots.

```
plot(fit.sector)

# Make a plot selection (or 0 to exit):
#
#  1: Distribution of factor returns
#  2: Factor exposures from the last period
#  3: Actual and Fitted asset returns
#  4: Time-series of R-squared values
#  5: Residual variance across assets
#  6: Scatterplot matrix of residuals, with histograms, density overlays,
#     correlations and significance stars
#  7: Factor Model Residual Correlation
#  8: Factor Model Return Correlation
#  9: Factor Contribution to SD
# 10: Factor Contribution to ES
# 11: Factor Contribution to VaR
# 12: Time series of factor returns
#
# Selection:
```

Figure 13: Actual (blue) and fitted (grey) factor model returns for the 1st 3 assets

Remarks: Only a subset of assets and factors selected by `a.sub` and `f.sub` are plotted. The first 2 factors (or just the solitary factor for a single factpr model) and first 6 assets are shown by default.

```
# Examples of group plots: looping disabled & no. of assets displayed = 4.
plot(fit.style.sector, which=3, a.sub=1:3, legend.loc=NULL, lwd=1)
```

4.2 Menu and looping

If the plot type argument `which` is not specified, a menu prompts for user input. In case multiple plots are needed, the menu is repeated after each plot (enter 0 to exit). User can also input a numeric vector of plot options via `which`.

4.3 Individual plots

Setting `plot.single=TRUE` enables individual asset plots. If there is more than one asset fit by the fitted object `x`, `asset.name` is also necessary. In case the `ffm` object `x` contains only a single asset's fit, plot.ffm can infer `asset.name` without user input.

Here's the individual plot menu.

```
plot(fit.style.sector, plot.single=TRUE, asset.name="MSFT")

# Make a plot selection (or 0 to exit):
#
#  1: Actual and fitted asset returns
#  2: Actual vs. fitted asset returns
#  3: Residuals vs. fitted asset returns
#  4: Residuals with standard error bands
#  5: Time series of squared residuals
#  6: Time series of absolute residuals
#  7: SACF and PACF of residuals
#  8: SACF and PACF of squared residuals
#  9: SACF and PACF of absolute residuals
# 10: Non-parametric density of residuals with normal overlaid
# 11: Non-parametric density of residuals with skew-t overlaid
# 12: Histogram of residuals with non-parametric density and normal overlaid
# 13: QQ-plot of residuals
#
# Selection:
```

Here are a few more examples which don't need interactive user input.

```
plot(fit.style.sector, plot.single=TRUE, asset.name="MSFT", which=4)
```

```
plot(fit.style.sector, plot.single=TRUE, asset.name="MSFT", which=9)
```

```
plot(fit.style.sector, plot.single=TRUE, asset.name="MSFT", which=13)
grid()
```

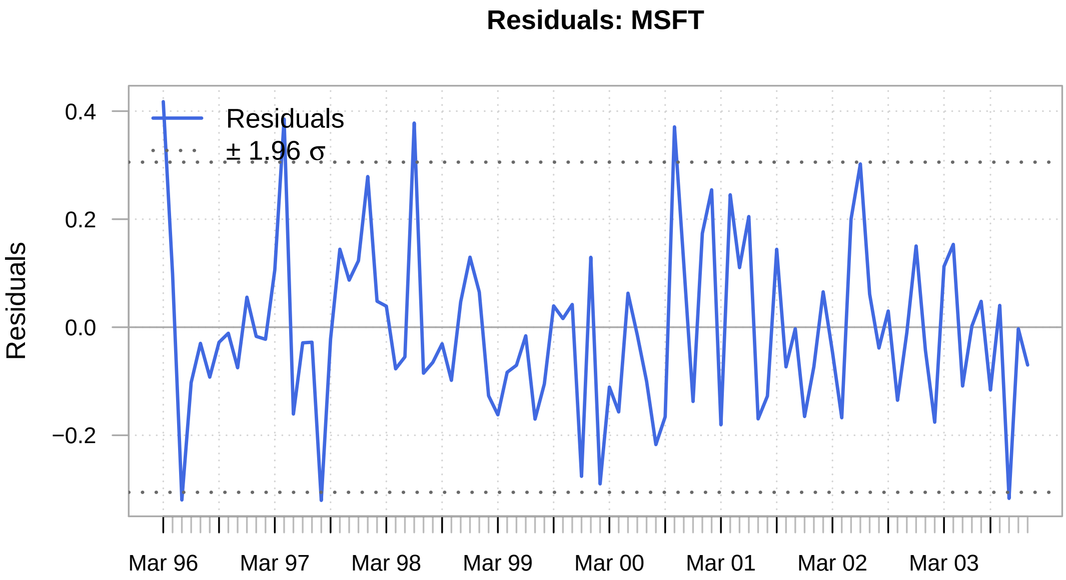

Figure 14: Time series plot of residuals with standard error bands: MSFT

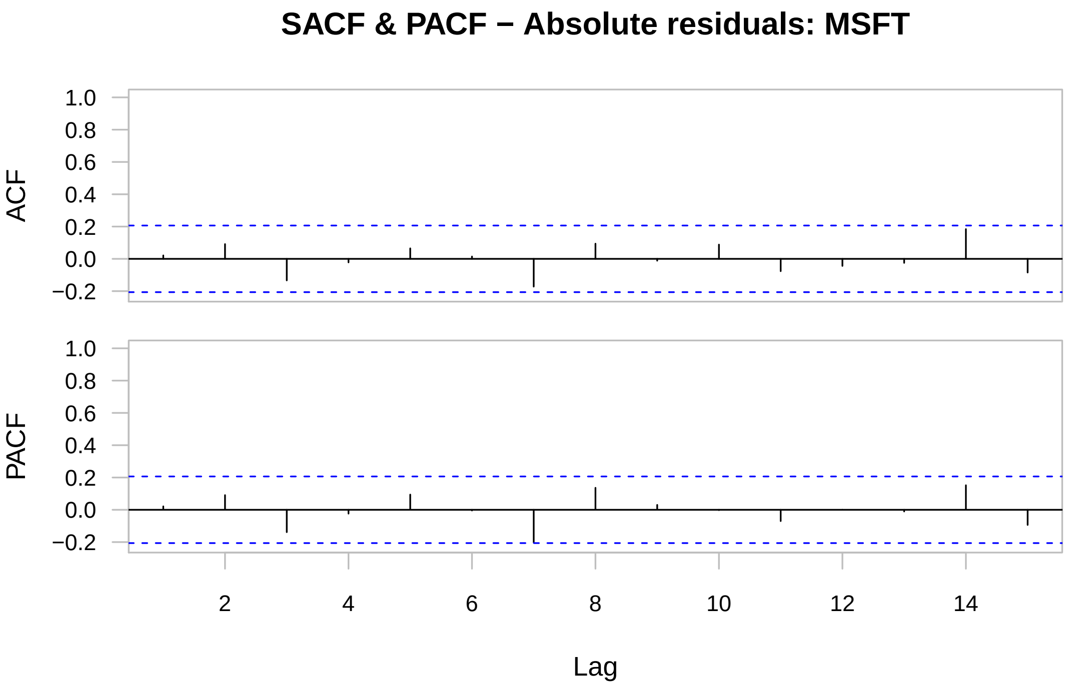

Figure 15: SACF and PACF of absolute residuals: MSFT

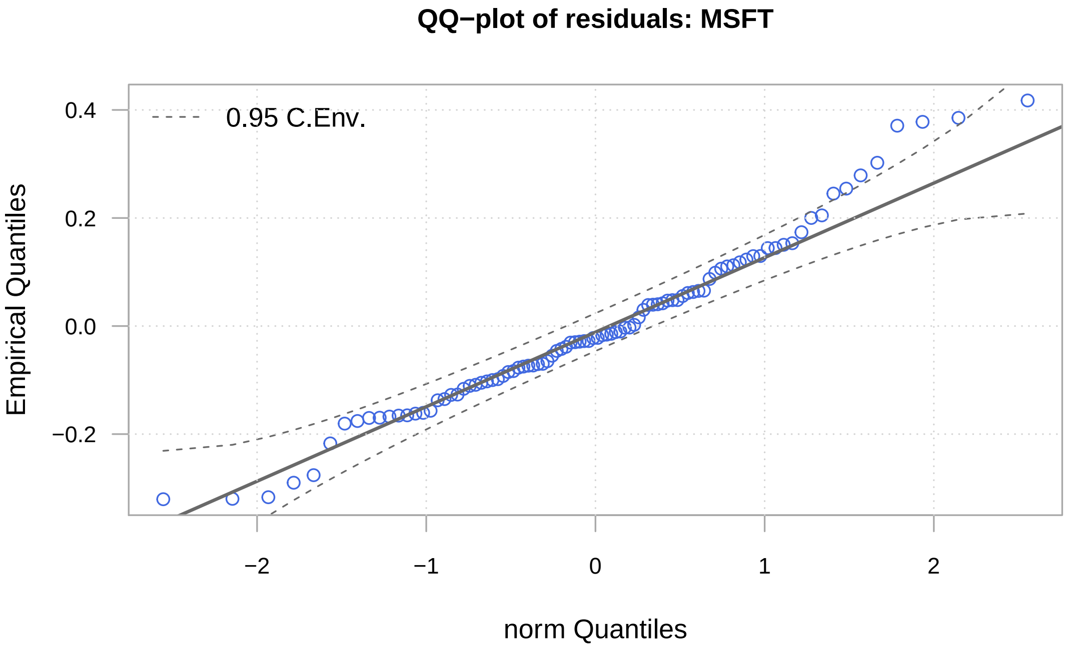

Figure 16: QQ-plot of residuals: MSFT

References

E. Epperlein and A. Smillie. Portfolio risk analysis Cracking VAR with kernels. *RISK-LONDON-RISK MAGAZINE LIMITED-*, 19(8):70, 2006.

E. F. Fama and K. R. French. The cross-section of expected stock returns. *the Journal of Finance*, 47(2):427–465, 1992.

R. C. Grinold and R. N. Kahn. Active portfolio management. 2000.

S. L. Heston and K. G. Rouwenhorst. Industry and country effects in international stock returns. *Journal of Portfolio Management*, 21:53–53, 1995.

J. Menchero. Characteristics of factor portfolios. *Journal of Performance Measurement*, 2010.

A. Meucci. Risk contributions from generic user-defined factors. *RISK-LONDON-RISK MAGAZINE LIMITED-*, 20(6):84, 2007.

E. Zivot and W. Jia-hui. Modeling Financial Time Series with S-Plus Springer-Verlag. 2006.