Introduction to CRSP Stocks and SPGMI Factors in PCRA

February 20, 2023

Doug Martin and Jon Spinney

02/20/23

1 Introduction

This document introduces the `PCRA` package *data.table* objects *stocksCRSP* and *factorsSPGMI*, and the general function `selectCRSPandSPGMI` for extracting subsets of these two data sets, either separately or jointly, where the user can specify that the extracted subset will be either a data.table or an `xts` time series object. We also discuss use of the simple function `stocksCRSPxts` for extracting xts times series subsets of the `stocksCRSP` data.table. The stocksCRSP data.table contains monthly CRSP stocks data for 294 stocks from January 1993 to December 2015, and the factorSPGMI data.table contains 14 factor exposures for the same 294 stocks and same time period. CRSP is the *Center for Research in Security Prices, LLC*, and *SPGMI* is the *S&P Global Market Intelligence* publicly traded U.S. company, referred to as *S&P Global* for short. Unlike the `PCRA` package itself, the stocksCRSP and factorsSPGMI data contained in the package *are not open source*, and *may not be redistributed in any form*.

The motivations for using the data.table package and its functionality in the `PCRA` package, and and in the Cross-Section Factor Models companion package `facmodCS`, are: (1) To take advantage of the data.table performance efficiency for factor model fitting and portfolio optimization using large cross-sections of stock returns and factor exposures, and (2) To enable portfolio optimization with varying sets of stocks and other investible assets across time periods. The first of these properties, along with the extensive data manipulation capabilities of data.table, are reflected in the immense popularity of the package, whose number of monthly downloads is approaching one million. Complete details about the use of data.table are provided by the Reference Manual and Vignettes at `https://cran.r-project.org/web/packages/data.table/`, which the reader is encouraged to read.

We load the `PCRA` and `data.table` packages, and take a look at three basic characteristics of `stocksCRSP` `data,` namely its class, dimension and the names of the items in each column:

```r
library(PCRA)
library(data.table)
class(stocksCRSP)

## [1] "data.table" "data.frame"

dim(stocksCRSP)  # Number of rows and columns

## [1] 81144    14

names(stocksCRSP)  # Names of items in each column

##  [1] "Date"         "Ticker"       "TickerLast"   "Company"      "CapGroup"
##  [6] "CapGroupLast" "GICS"         "Sector"       "Return"       "RetExDiv"
## [11] "Price"        "PrcSplitAdj"  "Ret13WkBill"  "MktIndexCRSP"
```

The item TickerLast is the ticker for a given stock in December, 2015, and CapGroupLast is the market capitalization group (LargeCap, MidCap, SmallCap or MicroCap) in December, 2015. The market capitalization break points of four capitalization groups in CapGroupLast are \$15.6B, \$5.4B, \$600M. In general, the ticker of a company can change over time, and the item Ticker represents the ticker of a stock on a monthly basis. Likewise the capitalization group of a company can change over time, and the item CapGroup is the capitalization group on a monthly basis. CapGroup can be useful when you want to study the performance of a capitalization group portfolio, e.g., a portfolio consisting of all SmallCap stocks in stocksCRSP, or a subset thereof, over time.

Here are the class, dimension and item names of the `factorsSPGMI` data:

```r
class(factorsSPGMI)

## [1] "data.table" "data.frame"

dim(factorsSPGMI)

## [1] 81144    22

names(factorsSPGMI)

##  [1] "Date"           "Ticker"          "TickerLast"     "Company"
##  [5] "CapGroup"        "CapGroupLast"    "GICS"            "Sector"
##  [9] "AnnVol12M"       "Beta60M"         "BP"              "EP"
## [13] "LogMktCap"       "PM12M1M"         "AccrualRatioCF" "AstAdjChg1YOCF"
## [17] "CFROIC"          "Chg1YAstTo"      "EBITDAEV"        "FCFP"
## [21] "PM1M"            "SEV"
```

The first 8 items in `factorsSPGMI` are identical to the first 8 items of `stockSPGMI`. The last 14 items of `factorsSPGMI` are the exposures of 14 distinct SPGMI factors for each of the cross-section of 294 stocks from 1993 to 2015.

Detailed descriptions of the data items in each column of `stocksCRSP` and `factorsSPGMI` are provided in the Reference Manual PCRA.pdf, which is downloadable from the PCRA CRAN website, and you can view these descriptions individually using the R `help` function as follows:

```
help(stocksCRSP)
help(factorsSPGMI)
```

We recommend that the reader become familiar with the detailed descriptions of the data types in the columns of `stocksCRSP` and `factorsSPGMI`.

It is clear from the output of the above code that the number of columns in each of `stocksCRSP` and `factorsSPGMI` is equal to the number of names. If you count the number of months in the time period from January 1993 to December 2015, you will find that there are 276 months, and since the number of stocks is 294, the product of the number of stocks times the number of months is $276 \times 294 = 81,144$, which is the number of rows of `stockCRSP` and `factorsSPGMI`.

The rows of `stocksCRSP` and `factorsSPGMI` are structured by grouping all 294 stocks for a given date as 294 contiguous rows. You can see this character of the data structure by displaying the first 5 and the last 5 rows of the first 4 columns of `stockCRSP` with the code line:

```
stocksCRSP[, 1:4]

##               Date Ticker TickerLast                      Company
##     1: 1993-01-31  ARONA        AAN                   AARONS INC
##     2: 1993-01-31    ABM        ABM          A B M INDUSTRIES INC
##     3: 1993-01-31    ABT        ABT           ABBOTT LABORATORIES
##     4: 1993-01-31   ADBE       ADBE                    ADOBE INC
##     5: 1993-01-31    ADI        ADI            ANALOG DEVICES INC
##    ---
## 81140: 2015-12-31    WGO        WGO     WINNEBAGO INDUSTRIES INC
## 81141: 2015-12-31    WHR        WHR              WHIRLPOOL CORP
## 81142: 2015-12-31    WMT        WMT                 WALMART INC
## 81143: 2015-12-31    WTS        WTS WATTS WATER TECHNOLOGIES INC
## 81144: 2015-12-31    XOM        XOM            EXXON MOBIL CORP
```

It is a special feature of data.tables that the R print method for a data.table prints by default only the first 5 and last 5 rows of the table. Note, however, that you can print the first 3 and last 3 rows of the first 4 columns with `stocksCRSP[c(1:3,81142:81144),1:4]`, and can print the first row of all the columns with `stocksCRSP[1,]`.

The above simple method of sub-setting columns of a data.table with numeric column values are the same as for data.frames. But sub-setting columns by column names differs for data.tables and data frames. We illustrate this by first converting the stocksCRSP data.table to a data frame in preparation for comparing sub-setting by columns for data frames and data.tables:

```
dat.df <- data.frame(stocksCRSP)
class(dat.df)

## [1] "data.frame"

names(dat.df)

##  [1] "Date"          "Ticker"        "TickerLast"    "Company"       "CapGroup"
##  [6] "CapGroupLast" "GICS"          "Sector"        "Return"        "RetExDiv"
## [11] "Price"         "PrcSplitAdj"  "Ret13WkBill"  "MktIndexCRSP"
```

Recall that you can subset a data frame by using a character vector of the column names in quotes. If you subset by two or more columns the subset will also be a data frame. But if you just subset a single column the result is a numeric object instead of a data frame, and you can force getting a data frame by using the `drop = FALSE` argument when sub-setting a single column, as in the code example below, where it is also shown that if you subset a data frame by two or more columns you always get a data frame.

```
dat.Return <- dat.df[, "Return"]
class(dat.Return)

## [1] "numeric"

dat.Return <- dat.df[, "Return", drop = FALSE]
class(dat.Return)
```

```
## [1] "data.frame"

dat.DateAndReturn <- dat.df[, c("Date", "Return")]
class(dat.DateAndReturn)

## [1] "data.frame"
```

In the case of a data.table you can subset a single column, say the Return column, by using the column name `Return` without quotes, but the result is a numeric object instead of a data.table. The way to return a data.table when sub-setting on one or more columns is to subset with a list version, e.g., `list(Return)` instead of `Return`, as in the code below, where we also show that use of .(`Return)` is a shorthand for `list(Return)`, and show that this shorthand works for selecting two or more columns of a data.table.

```
dat <- stocksCRSP  # This is a data.table
dat.Return <- dat[, Return]
class(dat.Return)

## [1] "numeric"

dat.Return <- dat[, list(Return)]
class(dat.Return)

## [1] "data.table" "data.frame"

dat.Return <- dat[, .(Return)]
class(dat.Return)

## [1] "data.table" "data.frame"

dat.DateAndReturn <- dat[, .(Date, Return)]
class(dat.DateAndReturn)

## [1] "data.table" "data.frame"
```

It turns out that there is quite a bit to learn about manipulating data.tables, and we suggest that the reader begin to do so by reading the data.table introduction document `https://cran.r-project.org/web/packages/data.table/vignettes/datatable-intro.html`. Meanwhile, the code below illustrates a very simple use of data.table sub-setting to compute the number of rows of `stocksCRSP`, and the first and last dates of `stocksCRSP.` We note that the function `unique` is used in the first line below because there are 294 occurrences of each `Date` value, and is used in the second line because `TickerLast` occurs for all 276 `Date` values for the 276 months of data.

```
nMonths <- length(unique(stocksCRSP[, Date]))
nStocks <- length(unique(stocksCRSP[, TickerLast]))
nMonths * nStocks  # Number of rows

## [1] 81144

range(stocksCRSP[, Date])  # First and last date

## [1] "1993-01-31" "2015-12-31"
```

Exercise. Create the data set datBP as follows: `datBP <- factorsSPGMI[ , .(Date, TickerLast, BP)]`. Without looking datBP, what do you think its dimension is? Then confirm whether or not your guess is correct.

2 Selecting stocksCRSP and factorsSPGMI

Now we discuss use of the function `selectCRSPandSPGMI` function for selecting quite general types of subsets of CRSP stocks or SPGMI factors, or merged versions of those subsets, and returning either a data.table or a time series xts object. This capability will be particularly useful for creating cross-section factor model data sets for use in the cross-section factor models package `facmodCS` currently under development. The default arguments of `selectCRSPandSPGMI` are:

```
args(selectCRSPandSPGMI)

## function (periodicity = "monthly", dateRange = c("1993-01-31",
##     "2015-12-31"), stockItems = c("Date", "TickerLast", "CapGroupLast",
##     "Sector", "Return", "Ret13WkBill", "MktIndexCRSP"), factorItems = c("BP",
##     "LogMktCap", "SEV"), subsetType = NULL, subsetValues = NULL,
##     outputType = "xts")
## NULL
```

At this point you should use

```r
help(selectCRSPandSPGMI)
```

and carefully read the Arguments and Details sections. In general `stockItems` can be any subset of the 14 `stocksCRSP` items, and `factorItems` can be any subset of the 22 factorsSPGMI items. Note that there is a duplication of 8 items in stocksCRSP and factorsSPGMI, and the subset of those items desired only needs to be included in either `stockItems` or `factorItems`, not both.

Example 1. In the code example below we select only 6 of 14 stocksCRSP items and select no factorsSPGMI items.

```r
stockItems1 <- c("Date", "TickerLast", "CapGroupLast",
    "Return", "MktIndexCRSP", "Ret13WkBill")
dateRange <- c("1997-01-31", "2010-12-31")
stocksSmall <- selectCRSPandSPGMI("monthly", dateRange = dateRange,
    stockItems = stockItems1, factorItems = NULL, subsetType = "CapGroupLast",
    subsetValues = "SmallCap", outputType = "data.table")
length(unique(stocksSmall[, TickerLast]))

## [1] 106

dim(stocksSmall)

## [1] 17808     6

names(stocksSmall)

## [1] "TickerLast"   "Date"         "CapGroupLast" "Return"       "MktIndexCRSP"
## [6] "Ret13WkBill"

range(stocksSmall[, Date])

## [1] "1997-01-31" "2010-12-31"
```

Exercise. Show that the number rows of `stocksSmall` is equal to the number of months times the number of stocks.

Example 2. In the code example below, for which we have in mind studying the relationship between smallcap stock returns and the Size, Beta and BM factors, we select smallcap stocks over the same time range as in the example above, but without the MktIndexCRSP and Ret13WkBill `stockItems`, and with the LogMktCap, Beta60M and EP `factorItems`.

```
stockItems2 <- c("Date", "TickerLast", "CapGroupLast",
    "Return")
factorItems <- c("LogMktCap", "Beta60M", "EP")
dateRange <- c("1997-01-31", "2010-12-31")
stocksSmall3Fac <- selectCRSPandSPGMI("monthly", dateRange = dateRange,
    stockItems = stockItems2, factorItems = factorItems,
    subsetType = "CapGroupLast", subsetValues = "SmallCap",
    outputType = "data.table")
length(unique(stocksSmall3Fac[, TickerLast]))

## [1] 106

dim(stocksSmall3Fac)

## [1] 17808     7

names(stocksSmall3Fac)

## [1] "TickerLast"   "Date"         "CapGroupLast" "Return"       "LogMktCap"
## [6] "Beta60M"      "EP"
```

Example 3. Here we modify the code of Example 1 by changing `subsetValues = “SmallCap”` to `subsetValues = “MicroCap”`, and changing `outputType = “data.table”` to `outputType = “xts”`, in order to create an xts time series object `stocksMicro`, which consists of the MicroCap stocks in stocksCRSP, along with the market returns ("MktIndexCRSP"), and the risk-free rate ("Ret13WkBill").

```
dateRange <- c("1997-01-31", "2010-12-31")
stocksMicro <- selectCRSPandSPGMI("monthly", dateRange = dateRange,
    stockItems = stockItems1, factorItems = NULL, subsetType = "CapGroupLast",
    subsetValues = "MicroCap", outputType = "xts")
class(stocksMicro)

## [1] "xts" "zoo"

dim(stocksMicro)
```

```
## [1] 168  36
```

```
names(stocksMicro)
```

```
##  [1] "AE"            "ALCO"          "ALOT"          "AMOT"          "ARKR"
##  [6] "ASEI"          "AVD"           "AVHI"          "BOOM"          "CMTL"
## [11] "COHU"          "CTS"           "DXYN"          "ESIO"          "FLXS"
## [16] "GHM"           "GPX"           "HNGR"          "HWKN"          "JOUT"
## [21] "LDR"           "MCS"           "MERC"          "MOD"           "MTRN"
## [26] "MYE"           "PIR"           "PKE"           "POWL"          "SPXC"
## [31] "SUP"           "TG"            "TTI"           "WGO"           "MktIndexCRSP"
## [36] "Ret13WkBill"
```

We see that stocksMicro is indeed an xts time series object, with 168 rows and 36 columns, where the names of the first 34 columns are the tickers of the microcap stocks, and the last two columns contain"MktIndexCRSP" and "Ret13WkBill", respectively.[1] For labeling convenience we replace "MktIndexCRSP" with “Market” and replace "Ret13WkBill" with “Risk-Free”, and display in Figure 2 the returns time series of the first 4 microcap stocks, the Market, and the Risk-Free rate.

```
names(stocksMicro)[c(35, 36)] <- c("Market", "Risk-Free")
```

[1]You get an xts object without the market returns and the risk-free rates by using `stockItems = stockItems2` instead of `stockItems = stockItems1` in the above code.

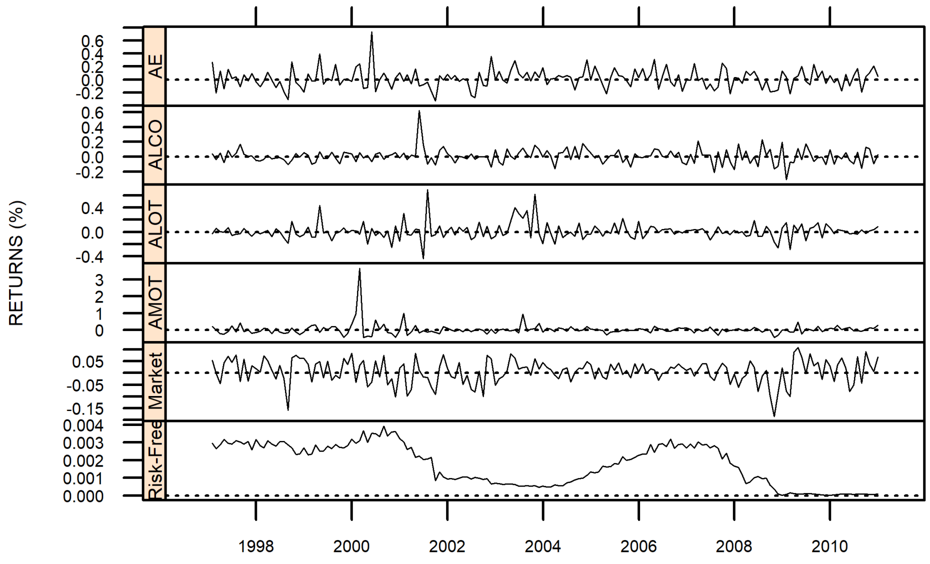

Figure 1: Returns of the first 4 CRSP microcap stocks, the Market, and the Risk-Free rate from 1997 to 2010

2.1 Manipulation of xts Objects

The package `xts` contains a large set of capabilities of manipulating and plotting xts objects, and you need to have `xts` loaded in order to take advantage of those capabilities. We illustrate here just one such capability here, namely use of the `index` and `range` functions as applied to the `stocksMicro` xts object created just above.

```
library(xts)
datesIndex <- index(stocksMicro)
class(datesIndex)

## [1] "Date"

length(datesIndex)  # Recall 14 years x 12 months

## [1] 168

head(datesIndex, 3)
```

```
## [1] "1997-01-31" "1997-02-28" "1997-03-31"

tail(datesIndex, 3)

## [1] "2010-10-31" "2010-11-30" "2010-12-31"

range(datesIndex)

## [1] "1997-01-31" "2010-12-31"
```

To find out more about working with xts objects, it is recommended to read the Vignettes "xts: Extensible Time Series" and "xts FAQ", available at `https://cran.r-project.org/web/packages/xts/index.html`. There are also a number of online tutorials for using xts.

2.2 A Simple Selection Function for stocksCRSP Data

As we have seen, the function `selectCRSPandSPGMI` has very general capabilities for selection subsets of `stockCRSP` and `factorsSPGMI`, either separately or jointly. But sometimes users of PCRA may wish to choose very simple types of subsets of stocks from `stocksCRSP`, e.g., by specifying only a time interval and a subset of the tickers in `TickerLast`, and creating an xts time series object. The function `stocksCRSPxts` serves this purpose, and its arguments are:

```
args(stocksCRSPxts)

## function (data, dateRange = c("1993-01-31", "2015-12-31"), tickerSet = NULL)
## NULL
```

Use of `stocksCRSPxts` with the default arguments creates an xts object that contains the entire cross-section of CRSP stocks for the entire time interval 1993 - 2015:

```
stocksAll <- stocksCRSPxts(stocksCRSP)
class(stocksAll)

## [1] "xts" "zoo"

dim(stocksAll)

## [1] 276 294
```

```r
library(xts)
range(index(stocksAll))

## [1] "1993-01-31" "2015-12-31"
```

The following is a typical application example where having the simple function `stocksCRSPxts` is quite convenient. The application goal is to compute the excess Kurtosis (eKR) of the cross-section of CRSP stocks monthly returns for the 6 contiguous time intervals "1993-1995", "1996-1999", "2000-2003", "2004-2007", "2008-2011", "2012-2015", and display the cross-section distributions of the eKR values with 6 boxplots. Here is the code, followed by the Figure 2 display of the boxplots.

```r
# Extract cross-sections of stocksCRSP monthly
# returns on 6 time intervals
dates1 <- c("1993-01-31", "1995-12-31")
dates2 <- c("1996-01-31", "1999-12-31")
dates3 <- c("2000-01-31", "2003-12-31")
dates4 <- c("2004-01-31", "2007-12-31")
dates5 <- c("2008-01-31", "2011-12-31")
dates6 <- c("2012-01-31", "2015-12-31")
ret1 <- stocksCRSPxts(stocksCRSP, dateRange = dates1)
ret2 <- stocksCRSPxts(stocksCRSP, dateRange = dates2)
ret3 <- stocksCRSPxts(stocksCRSP, dateRange = dates3)
ret4 <- stocksCRSPxts(stocksCRSP, dateRange = dates4)
ret5 <- stocksCRSPxts(stocksCRSP, dateRange = dates5)
ret6 <- stocksCRSPxts(stocksCRSP, dateRange = dates6)
# Compute cross-section of excess kurtosis values
# on those time intervals
eKR1 <- apply(coredata(ret1), 2, KRest)
eKR2 <- apply(coredata(ret2), 2, KRest)
eKR3 <- apply(coredata(ret3), 2, KRest)
eKR4 <- apply(coredata(ret4), 2, KRest)
eKR5 <- apply(coredata(ret5), 2, KRest)
eKR6 <- apply(coredata(ret6), 2, KRest)
eKR <- cbind(eKR1, eKR2, eKR3, eKR4, eKR5, eKR6)
times <- c("1993-1995", "1996-1999", "2000-2003", "2004-2007",
    "2008-2011", "2012-2015")
```

```r
boxplot(eKR, xaxt = "n", ylim = c(-2, 12), main = "Excess Kutosis stocksCRSP",
    cex.main = 1.5, col = "cyan")
axis(1, at = 1:6, labels = times, cex.axis = 1.1)
abline(h = 0, lty = "dotted")
```

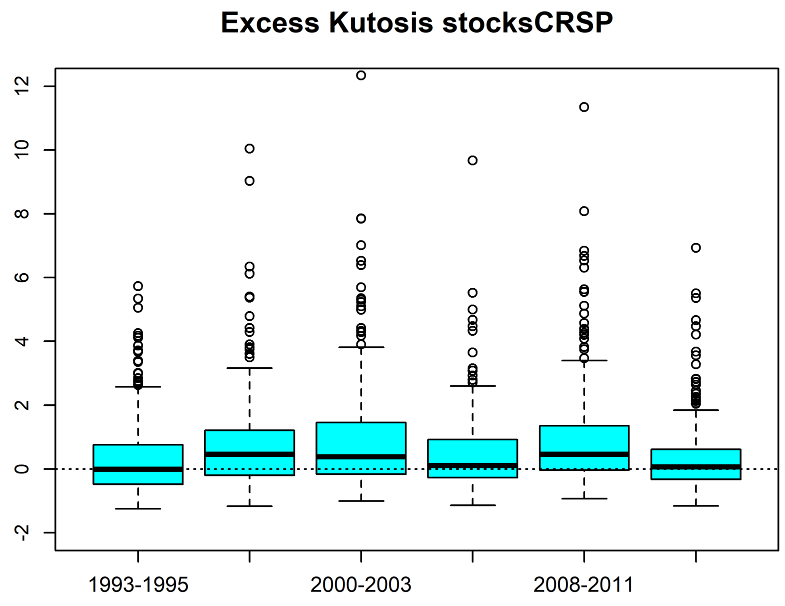

Figure 2: Boxplots of cross-sections of CRSP stock returns excess kurtosis (eKR) for 6 contiguous time intervals

2.3 Using the Weekly and Daily stocksCRSP Data

Here we show how to access and use weekly CRSP stock returns, which and contained in the data.table `stocksCRSPweekly`.

```r
stocksCRSPweekly <- getPCRAData(dataset = "stocksCRSPweekly")
dateRange <- c("2000-01-7", "2000-12-31")
stocksMicroWeekly <- selectCRSPandSPGMI("weekly", dateRange = dateRange,
    stockItems = stockItems1, factorItems = NULL, subsetType = "CapGroupLast",
    subsetValues = "MicroCap", outputType = "xts")
dim(stocksMicroWeekly)

## [1] 52 36

names(stocksMicroWeekly)[35] <- "Market"
```

```r
tsPlotMP(stocksMicroWeekly[, c(1:5, 35)], scaleType = "free",
    stripText.cex = 0.45, yname = "RETURNS", layout = c(1,
        6), type = "b", axis.cex = 0.7, lwd = 0.6)
dev.off()
```

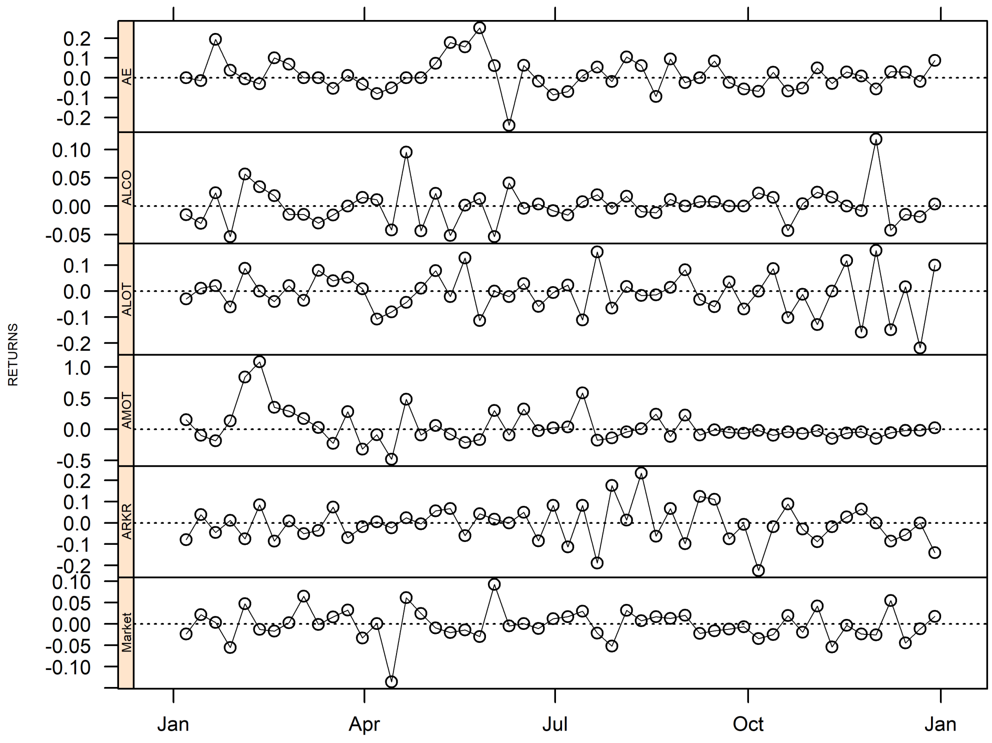

Figure 3: Time series of weekly returns for the first 5 of the 34 MicroCap CRSP stocks, and the CRSP Market returns

For daily CRSP stocks, one replaces `dataset = "stocksCRSPweekly"` with `dataset = "stocksCRSPdaily"` in the `getPCRAData` function, and replace the argument “`weekly`” with “`daily`” in the `selectCRSPandSPGMI function`.

3 Basic Data.Table Manipulations

The data.table package provides a variety of data manipulation and analysis tools, some of which are quite simple, and in this section we demonstrate a few of these.

3.1 Sector and Capitalization Group Counts

You will have noticed that stocksCRSP and factorsSPGMI data.tables contain Sector, and CapGroupLast variables. The following code shows a simple way to determine the the number of stocks in each sector and each capgroup.

```
lastDate <- tail(factorsSPGMI[, Date], 1)
factorsSPGMILast <- factorsSPGMI[Date == lastDate]
factorsSPGMILast[, .N, by = Sector]

##                    Sector  N
## 1: Information Technology 41
## 2:            Industrials 92
## 3:            Health Care 31
## 4:       Consumer Staples 29
## 5:                 Energy 20
## 6:              Materials 26
## 7: Consumer Discretionary 46
## 8: Communication Services  9

factorsSPGMILast[, .N, by = CapGroup]

##    CapGroup   N
## 1: SmallCap 106
## 2: LargeCap  87
## 3: MicroCap  34
## 4:   MidCap  67
```

Exercise. Modify the above code to check that the stocks sectors are the same in the first month as in the last month of the data range from January 1993 to December 2015, but their capgroups are different in the first month than in the last month.

3.2 Extract a Subset of the Variables

Typically you will be working with a subset of the variables in `stocksCRSP` or `factorsSPGMI`, or in their merged data.table. For example, see the column names of the data.table `stocksSmall3Fac` in Example 2 of Section 2. You often end up just wanting to work with a subset of those items. For example, suppose you become interested in only the Date, TickerLast, Return, Beta60M and EP items. Then you can create such a data.table which is also called `stocksSmall3Fac` with the following code:

```
colNames <- c("Date", "TickerLast", "Return", "Beta60M",
    "EP")
stocksSmall3Fac <- stocksSmall3Fac[, .SD, .SDcols = colNames]
names(stocksSmall3Fac)

## [1] "Date"       "TickerLast" "Return"     "Beta60M"    "EP"
```

In this code .SD represents the subset of columns that is specified by .SDcols. For further details on .SD see `https://cran.r-project.org/web/packages/data.table/vignettes/datatable-sd-usage.html`.

3.3 Renaming Variables

Some of the names of variables in stocksCRSP and factorsSPGMI are fairly long, and it is sometimes convenient to replace one or more of them with shorter names. For example, you may wish to replace the names "MktIndexCRSP" and “Ret13WkBill” in stocksDatSmallcap1 in Section 2 with the shorter common names “Market” and "RiskFree". This is easily done using the data.table `setnames` function:

```
setnames(stocksSmall, old = c("MktIndexCRSP", "Ret13WkBill"),
    new = c("Market", "RiskFree"))
names(stocksSmall)

## [1] "TickerLast"   "Date"         "CapGroupLast" "Return"       "Market"
## [6] "RiskFree"
```

3.4 Extracting a SmallCap Stocks Sector

One often wants to study portfolios consisting a single sector of smallcap stocks, e.g., the Information Technology sector of smallcap stocks, and this is easily done as follows by first extracting the set of all smallcap stocks for the default `dateRange` of 1993 to 2015, and then extracting the Information Technology sector of the smallcap stocks.

```
stockItems <- c("Date", "TickerLast", "CapGroupLast",
    "Return", "Sector")
stocksSmallCap <- selectCRSPandSPGMI("monthly", stockItems = stockItems,
    factorItems = NULL, subsetType = "CapGroupLast",
    subsetValues = "SmallCap", outputType = "data.table")
length(unique(stocksSmallCap[, TickerLast]))
```

```
## [1] 106

range(stocksSmallCap[, Date])

## [1] "1993-01-31" "2015-12-31"

stocksSmallCapIT <- stocksSmallCap[Sector == "Information Technology"]
unique(stocksSmallCapIT[, TickerLast])

##  [1] "AAN"  "AMD"  "AXE"  "BMI"  "CRUS" "CY"   "DBD"  "IDTI" "IIVI" "KLIC"
## [11] "MENT" "MTSC" "NEWP" "PLXS"
```

3.5 Extracting a Time Range of the Data

One often wants to look at stocks or factors, or merged stocks and factors data, over a specified time range subset of an initial long time range. For example, you extract the 5 years from 2010 to 2014 of the `stocksSmallCapIT` data with the first line below, and check the time range and number of months with the second and third lines:

```
stocksSmallCapIT5Year <- stocksSmallCapIT[stocksSmallCapIT[,
    Date] >= as.Date("2010-01-31") & stocksSmallCapIT[,
    Date] <= as.Date("2014-12-31"), ]
range(stocksSmallCapIT5Year[, Date])

## [1] "2010-01-31" "2014-12-31"

length(unique(stocksSmallCapIT5Year[, Date]))

## [1] 60
```