As we see, both tests did not reject the null hypothesis and the suspicious value should not be removed. Further calculations should be done on the full sample.

Another example is testing a simple dataset for two opposite outliers, and two outliers at one tail:

```
> x = c(41.3,44.5,44.7,45.9,46.8,49.1)
> grubbs.test(x,type=11)

        Grubbs test for two opposite outliers

data:  x
G = 2.9908, U = 0.1025, p-value = 0.06497
alternative hypothesis: 41.3 and 49.1 are outliers

> x = c(45.1,45.9,46.1,46.2,49.1,49.2)
> grubbs.test(x,type=20)

        Grubbs test for two outliers

data:  x
U = 0.0482, p-value = 0.03984
alternative hypothesis: highest values 49.1 , 49.2 are outliers

>
```

In the first dataset, the smallest and greatest value should not be rejected. The second example rejects the null hypothesis: 49.1 and 49.2 are outliers, and calculations should be made without them.

The last example is testing for outlying variance. We have calculated variance in 8 groups (5 measurements in each group) of results and obtained: 1.2, 2.5, 2.9, 3.5, 3.6, 3.9, 4.0, 7.9. We must check now if the smallest or largest variance should be considered in calculations:

```
> v = c(1.2, 2.5, 2.9, 3.5, 3.6, 3.9, 4.0, 7.9)
> n = rep(5,8)
> cochran.test(v,n)

        Cochran test for outlying variance

data:  v
C = 0.2678, df = 5, k = 8, p-value = 0.3579
alternative hypothesis: Group 8 has outlying variance
sample estimates:
  1   2   3   4   5   6   7   8
1.2 2.5 2.9 3.5 3.6 3.9 4.0 7.9

> cochran.test(v,n,inlying=TRUE)

        Cochran test for inlying variance

data:  v
C = 0.0407, df = 5, k = 8, p-value < 2.2e-16
alternative hypothesis: Group 1 has inlying variance
sample estimates:
  1   2   3   4   5   6   7   8
1.2 2.5 2.9 3.5 3.6 3.9 4.0 7.9

>
```

The tests show that first group has inlying variance, significantly smaller than the others.

Bibliography

V. Barnett, T. Lewis. *Outliers in statistical data.* Wiley.

W.J. Dixon. Analysis of extreme values. *Ann. Math. Stat.*, 21(4):488-506, 1950.

W.J. Dixon. Ratios involving extreme values. *Ann. Math. Stat.*, 22(1):68-78, 1951.

D.B. Rorabacher. Statistical Treatment for Rejection of Deviant Values: Critical Values of Dixon Q Parameter and Related Subrange Ratios at the 95 percent Confidence Level. *Anal. Chem.* , 83(2):139-146, 1991.

F.E. Grubbs. Sample Criteria for testing outlying observations. *Ann. Math. Stat.* , 21(1):27-58, 1950.

E.S. Pearson, C.C. Chekar. The efficiency of statistical tools and a criterion for the rejection of outlying observations. *Biometrika* , 28(3):308-320, 1936.

H.A. David, H.O. Hartley, E.S. Pearson. The distribution of the ratio, in a single normal sample, of range to standard deviation. *Biometrika* , 41(3):482-493, 1954.

G.W. Snedecor, W.G. Cochran. Statistical Methods. *Iowa State University Press* , 1980.

Lukasz Komsta
Department of Medicinal Chemistry
Skubiszewski Medical University of Lublin, Poland
luke@novum.am.lublin.pl

Analysing equity portfolios in R

Using the portfolio package

by David Kane and Jeff Enos

Introduction

R is used by major financial institutions around the world to manage billions of dollars in equity (stock) portfolios. Unfortunately, there is no open source R package for facilitating this task. The **portfolio** package is meant to fill that gap. Using **portfolio**, an analyst can create an object of class `portfolioBasic` (weights only) or `portfolio` (weights and shares), examine its *exposures* to various factors, calculate its *performance* over time, and determine the *contributions* to performance from various categories of stocks. Exposures, performance and contributions are the basic building blocks of portfolio analysis.

One Period, Long-Only

Consider a simple long-only portfolio formed from the universe of 30 stocks in the Dow Jones Industrial Average (DJIA) in January, 2005. Start by loading and examining the input data.

```
> library(portfolio)
> data(dow.jan.2005)
> summary(dow.jan.2005)

    symbol               name          
 Length:30          Length:30          
 Class :character   Class :character   
 Mode  :character   Mode  :character   




     price                    sector  
 Min.   : 21.0   Industrials   :6  
 1st Qu.: 32.1   Staples       :6  
 Median : 42.2   Cyclicals     :4  
 Mean   : 48.6   Financials    :4  
 3rd Qu.: 56.0   Technology    :4  
 Max.   :103.3   Communications:3  
                 (Other)       :3  
    cap.bil       month.ret       
 Min.   : 22.6   Min.   :-0.12726  
 1st Qu.: 53.9   1st Qu.:-0.05868  
 Median : 97.3   Median :-0.02758  
 Mean   :126.0   Mean   :-0.02914  
 3rd Qu.:169.3   3rd Qu.: 0.00874  
 Max.   :385.9   Max.   : 0.04468  

> head(dow.jan.2005)

     symbol                         name price
140      AA                    ALCOA INC 31.42
214      MO             ALTRIA GROUP INC 61.10
270     AXP          AMERICAN EXPRESS CO 56.37
294     AIG AMERICAN INTERNATIONAL GROUP 65.67
946      BA                   BOEING CO 51.77
1119    CAT              CATERPILLAR INC 97.51
          sector cap.bil month.ret
140    Materials   27.35 -0.060789
214      Staples  125.41  0.044681
270   Financials   70.75 -0.051488
294   Financials  171.04  0.009441
946  Industrials   43.47 -0.022600
1119 Industrials   33.27 -0.082199
```

The DJIA consists of exactly 30 large US stocks. We provide a minimal set of information for constructing a long-only portfolio. Note that `cap.bil` is market capitalization in billions of dollars, `price` is the per share closing price on December 31, 2004, and `month.ret` is the one month return from December 31, 2004 through January 31, 2005.

In order to create an object of class `portfolioBasic`, we can use this data and form a portfolio on the basis of a "nonsense" variable like `price`.

```
> p <- new("portfolioBasic",
+     date = as.Date("2004-12-31"),
+     id.var = "symbol", in.var = "price",
+     sides = "long", ret.var = "month.ret",
+     data = dow.jan.2005)

An object of class "portfolioBasic" with 6 positions

Selected slots:
name: Unnamed portfolio
date: 2004-12-31
in.var: price
ret.var: month.ret
type: equal
size: quintile

> summary(p)

Portfolio: Unnamed portfolio

       count      weight
Long:      6           1

Top/bottom positions by weight:
   id  pct
1 AIG 16.7
2 CAT 16.7
3 IBM 16.7
4 JNJ 16.7
5 MMM 16.7
6 UTX 16.7
```

In other words, we have formed a portfolio of the highest priced 6 stocks out of the 30 stocks in the DJIA. The `id.var` argument causes the portfolio to use `symbol` as the key for identifying securities throughout the analysis. `in.var` selects the variable by which stocks are ranked in terms of desirability. `sides` set to `long` specifies that we want a long-only portfolio. `ret.var` selects the return variable for measuring performance. In this case, we are interested in how the portfolio does for the month of January 2005.

The default arguments to `portfolioBasic` form equal-weighted positions; in this case, each of the 6 stocks has 16.67% of the resulting portfolio. The defaults also lead to a portfolio made up of the best 20% (or `quintile`) of the universe of stocks provided to the `data` argument. Since 20% of 30 is 6, there are 6 securities in the resulting portfolio.

Exposures

Once we have a portfolio, the next step is to analyse its exposures. These can be calculated with respect to both numeric or factor variables. The method `exposure` will accept a vector of variable names.

```
> exposure(p, exp.var = c("price", "sector"))

numeric
  variable exposure
1    price     85.1
```

```
sector
     variable exposure
2 Industrials    0.500
1  Financials    0.167
3     Staples    0.167
4  Technology    0.167
```

The weighted average price of the portfolio is 85. In other words, since the portfolio is equal-weighted, the average share price of AIG, CAT, IBM, JNJ, MMM, and UTX is 85. This is a relatively high price, but makes sense since we explicitly formed the portfolio by taking the 6 highest priced stocks out of the DJIA.

Each of the 6 stocks is assigned a sector and 3 of them are Industrials. This compares to the 20% of the entire universe of 30 stocks that are in this sector. In other words, the portfolio has a much higher exposure to Industrials than the universe as a whole. Similarly, the portfolio has no stocks from the Communications, Cyclicals or Energy sectors despite the fact that these make up almost 27% of the DJIA universe.

Performance

Time plays two roles in the `portfolio` class. First, there is the moment of portfolio formation. This is the instant when all of the data, except for future returns, is correct. After this moment, of course, things change. The price of AIG on December 31, 2004 was $65.67, but by January 5, 2005 it was $67.35.

The second role played by time on the `portfolio` class concerns future returns. `ret.var` specifies a return variable which measures the performance of individual stocks going forward. These returns can be of any duration — an hour, a day, a month, a year — but should generally start from the moment of portfolio formation. In this example, we are using one month forward returns. Now that we know the portfolio's exposures at the start of the time period, we can examine its performance for the month of January.

```
> performance(p)

Total return:  -1.71 %

Best/Worst performers:
   id weight      ret  contrib
2 CAT  0.167 -0.08220 -0.01370
3 IBM  0.167 -0.05234 -0.00872
6 UTX  0.167 -0.02583 -0.00431
1 AIG  0.167  0.00944  0.00157
4 JNJ  0.167  0.02018  0.00336
5 MMM  0.167  0.02790  0.00465
```

The portfolio lost 1.7% of its value in January. The worst performing stock was CAT (Caterpillar), down more than 8%. The best performing stock was MMM (3M), up almost 3%. The `contrib` (contribution) of each stock to the overall performance of the portfolio is simply its weight in the portfolio multiplied by its return. The sum of the 6 individual contributions yields -1.7%.

Contributions

The contributions of individual stocks are not that interesting in and of themselves. More important is to examine summary statistics of the contributions across different categories of stocks. Consider the use of the `contribution` method:

```
> contribution(p, contrib.var = c("sector"))

sector
          variable weight  contrib     roic
5   Communications  0.000  0.00000  0.00000
6    Conglomerates  0.000  0.00000  0.00000
7        Cyclicals  0.000  0.00000  0.00000
8           Energy  0.000  0.00000  0.00000
1       Financials  0.167  0.00157  0.00944
2      Industrials  0.500 -0.01336 -0.02671
9        Materials  0.000  0.00000  0.00000
3          Staples  0.167  0.00336  0.02018
4       Technology  0.167 -0.00872 -0.05234
10       Utilities  0.000  0.00000  0.00000
```

`contribution`, like `exposure`, accepts a vector of variable names. In the case of `sector`, the contribution object displays the 10 possible values, the total weight of the stocks for each level and the sum of the contributions for those stocks. Only 4 of the 10 levels are represented by the stocks in the portfolio. The other 6 levels have zero weight and, therefore, zero contributions.

The sector with the biggest weight is Industrials, with half of the portfolio. Those 3 stocks did poorly, on average, in January and are therefore responsible for -1.3% in total losses. There is only a single Technology stock in the portfolio, IBM. Because it was down 5% for the month, its contribution was -0.87%. Since IBM is the only stock in its sector in the portfolio, the contribution for the sector is the same as the contribution for IBM.

The last column in the display shows the `roic` — the return on invested capital — for each level. This captures the fact that raw contributions are a function of both the total size of positions and their return. For example, the reason that the total contribution for Industrials is so large is mostly because they account for such a large proportion of the portfolio. The individual stocks performed poorly, on average, but not that poorly. Much worse was the performance of the Technology stock. Although the total contribution from Technology was only 60% of that of Industrials, this was on a total position weight only 1/3 as large. In other words, the return on total capital was *much worse* in Technology than in Industrials even though Industrials accounted for a larger share of the total losses.

Think about roic as useful in determining contributions on the margin. Imagine that you have the chance to move $1 out of one sector and into another. Where should than initial dollar come from? Not from the sector with the worst total contribution. Instead, the marginal dollar should come from the sector with the worst roic and should be placed into the sector with the best roic. In this example, we should move money out of Technology and into Staples.

contribution can also work with numeric variables.

```
> contribution(p, contrib.var = c("cap.bil"))

cap.bil
      rank     variable weight   contrib    roic
1  1 - low (22.6,50.9]  0.167 -0.013700 -0.08220
2        2 (50.9,71.1]  0.333  0.000345  0.00103
4        3  (71.1,131]  0.000  0.000000  0.00000
3        4   (131,191]  0.500 -0.003787 -0.00757
5 5 - high   (191,386]  0.000  0.000000  0.00000
```

Analysing contributions only makes sense in the context of categories into which each position can be placed. So, we need to break up a numeric variable like cap into discrete, exhaustive categories. The contribution function provides various options for doing so, but most users will be satisfied with the default behavior of forming 5 equal sized quintiles based on the distribution of the variable in the entire universe.

In this example, we see that there are no portfolio holdings among the biggest 20% of stocks in the DJIA. Half the portfolio comes from stocks in the second largest quintile. Within each category, the analysis is the same as that above for sectors. The worst performing category in terms of total contribution is the smallest quintile. This category also has the lowest roic.

One Period Long-Short

Having examined a very simple long-only portfolio in order to explain the concepts behind *exposures*, *performance* and *contributions*, it is time to consider a more complex case, a long-short portfolio which uses the same underlying data.

```
> p <- new("portfolioBasic",
+     date = as.Date("2004-12-31"),
+     id.var = "symbol", in.var = "price",
+     type = "linear", sides = c("long",
+         "short"), ret.var = "month.ret",
+     data = dow.jan.2005)

An object of class "portfolioBasic" with
12 positions

Selected slots:
name: Unnamed portfolio
date: 2004-12-31
in.var: price
ret.var: month.ret
type: linear
size: quintile

> summary(p)

Portfolio: Unnamed portfolio

       count      weight
Long:      6           1
Short:     6          -1

Top/bottom positions by weight:
    id    pct
12  UTX  28.57
5   IBM  23.81
2   CAT  19.05
8   MMM  14.29
1   AIG   9.52
10  PFE  -9.52
9  MSFT -14.29
11  SBC -19.05
6  INTC -23.81
4   HPQ -28.57
```

Besides changing to a long-short portfolio, we have also provided a value of "linear" to the type argument. As the summary above shows, this yields a portfolio in which the weights on the individual positions are (linearly) proportional to their share prices. At the end of 2004, the lowest priced stock in the DJIA was HPQ (Hewlett-Packard) at $20.97. Since we are using price as our measure of desirability, HPQ is the biggest short position. Since we have not changed the default value for size from "quintile," there are still 6 positions per side.

```
> plot(p)
```

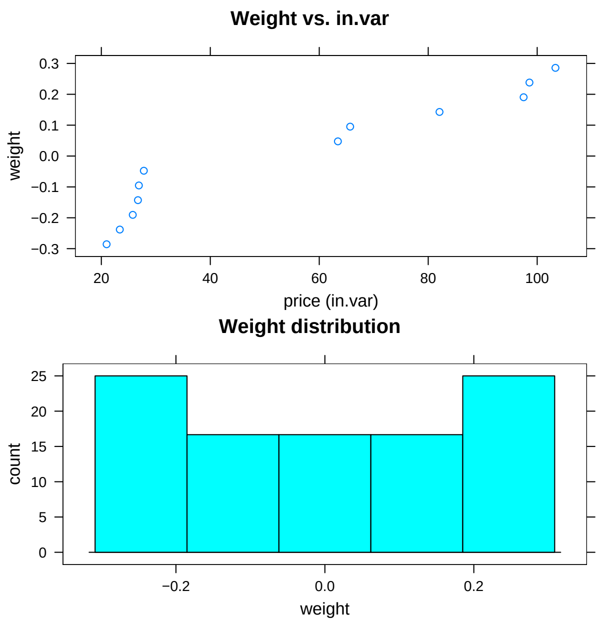

Figure 1: Plot of a portfolioBasic object.

Figure 1 shows the result of calling `plot` on this portfolio object. The top plot displays the relationship between position weights and their corresponding `in.var` values, while the bottom plot shows a histogram of position weights.

The display for the exposure object is now somewhat different to accommodate the structure of a long-short portfolio.

```
> exposure(p, exp.var = c("price", "sector"))

numeric
  variable long short exposure
1    price 92.6 -24.2     68.4

sector
        variable   long   short exposure
2    Industrials 0.6190  0.0000   0.6190
1     Financials 0.0952  0.0000   0.0952
3        Staples 0.0476 -0.0952  -0.0476
5 Communications 0.0000 -0.2381  -0.2381
4     Technology 0.2381 -0.6667  -0.4286
```

The `long` exposure to price is simply the weighted average of price on the long side of the portfolio, where the weighting is done in proportion to the size of the position in each stock. The same is true on the short side. Since a linear weighting used here emphasises the tail of the distribution, the long exposure is greater than the long exposure of the equal weighted portfolio considered above, $93 versus $85.

Since the weights on the short side are actually negative — in the sense that we are negatively exposed to positive price changes in these stocks — the weighted average on the short side for a positive variable like price is also negative. Another way to read this is to note that the weighted average price on the short side is about $24 but that the portfolio has a negative exposure to this number because these positions are all on the short side.

One reason for the convention of using a negative sign for short side exposures is that doing so makes the overall exposure of the portfolio into a simple summation of the long and short exposures. (Note the assumption that the weights on both sides are equal. In future versions of the **portfolio** package, we hope to weaken these and other requirements.) For this portfolio, the overall exposure is 68. Because the portfolio is long the high priced stocks and short the low priced ones, the portfolio has a positive exposure to the price factor. Colloquially, we are "long price."

A similar analysis applies to sector exposures. We have 62% of our long holdings in Industrials but zero of our short holdings. We are, therefore, 62% long Industrials. We have 24% of the longs holdings in Technology, but 67% of the short holdings; so we are 43% short Technology.

Calling `plot` on the exposure object produces a bar chart for each exposure category, as seen in Figure 2. Since all numeric exposures are grouped together, a plot of numeric exposure may only make sense if all numeric variables are on the same scale. In this example, the only numeric exposure we have calculated is that to `price`.

```
> plot(exposure(p, exp.var = c("price",
+     "sector")))
```

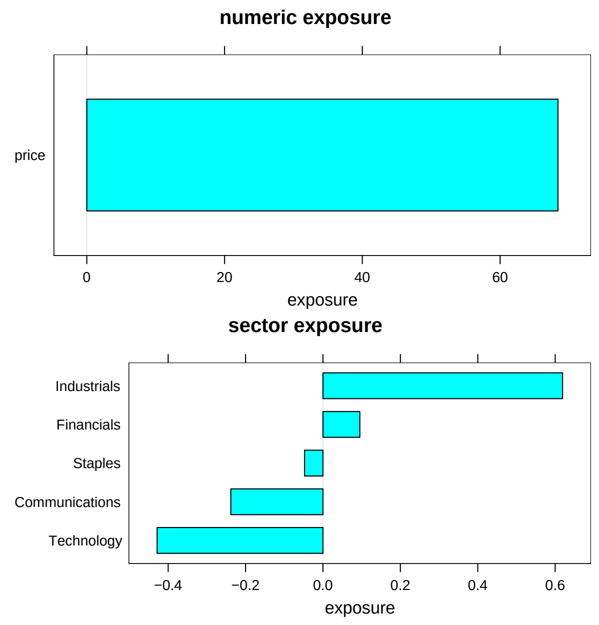

Figure 2: Plot of an exposure object for a single period. Bar charts are sorted by exposure.

The performance object is similar for a long-short portfolio.

```
> performance(p)

Total return:  2.19 %

Best/Worst performers:
     id  weight      ret  contrib
2   CAT  0.1905 -0.08220 -0.01566
5   IBM  0.2381 -0.05234 -0.01246
12  UTX  0.2857 -0.02583 -0.00738
3   DIS -0.0476  0.02986 -0.00142
1   AIG  0.0952  0.00944  0.00090
8   MMM  0.1429  0.02790  0.00399
6  INTC -0.2381 -0.04019  0.00957
10  PFE -0.0952 -0.10152  0.00967
11  SBC -0.1905 -0.06617  0.01260
4   HPQ -0.2857 -0.06581  0.01880
```

The portfolio was up 2.2% in January. By default, the summary of the `performance` object only provides the 5 worst and best contributions to return. HPQ was the biggest winner because, though it was only down 7% for the month, its large weighting caused it to contribute almost 2% to overall performance. The 19% weight of CAT in the portfolio placed it as only the third largest position on the long side, but its -8% return for the month made it the biggest drag on performance.

The contribution function provides similar output for a long-short portfolio.

```
> contribution(p, contrib.var = c("cap.bil",
+     "sector"))

cap.bil
      rank     variable weight  contrib     roic
1  1 - low (22.6,50.9] 0.0952 -0.01566 -0.16440
2        2 (50.9,71.1] 0.3810  0.01399  0.03671
3        3  (71.1,131] 0.0952  0.01260  0.13235
4        4   (131,191] 0.3095 -0.00103 -0.00334
5 5 - high   (191,386] 0.1190  0.01202  0.10098

sector
         variable weight contrib    roic
1  Communications 0.1190  0.0112  0.0939
6   Conglomerates 0.0000  0.0000  0.0000
7       Cyclicals 0.0000  0.0000  0.0000
8          Energy 0.0000  0.0000  0.0000
2      Financials 0.0476  0.0009  0.0189
3     Industrials 0.3095 -0.0191 -0.0616
9       Materials 0.0000  0.0000  0.0000
4         Staples 0.0714  0.0106  0.1488
5      Technology 0.4524  0.0183  0.0404
10      Utilities 0.0000  0.0000  0.0000
```

As in the last example, the weight column reflects the proportion of the portfolio invested in each category. In this case, we have 45% of the total capital (or weight) of the long and short sides *considered together* invested in the Technology sector. We know from the exposure results above that most of this is invested on the short side, but in the context of contributions, it does not matter on which side the capital is deployed.

```
> plot(contribution(p, contrib.var = c("cap.bil",
+     "sector")))
```

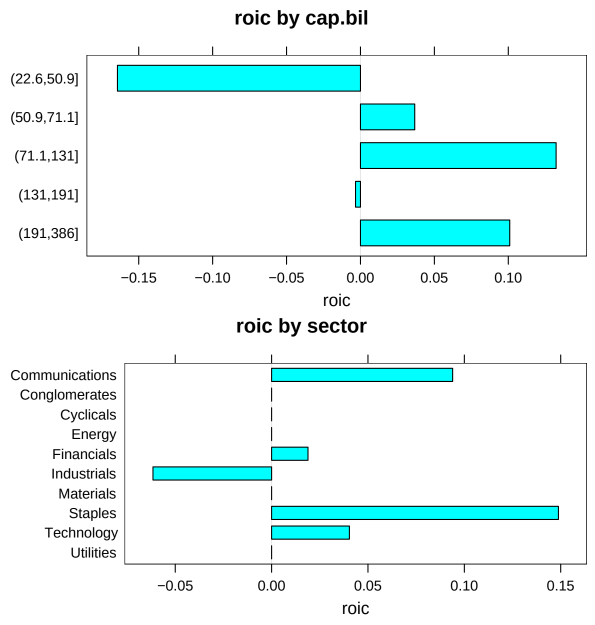

Figure 3: Plot of a contribution object for a single period.

Plotting objects of class contribution produces output similar to plotting exposures, as can be seen in Figure 3. This time, however, we see values of roic plotted against the categories that make up each contrib.var. For numeric variables, roic for each interval is shown; numeric intervals will always appear in order.

Multi-Period, Long-Short

Analysing a portfolio for a single time period is a useful starting point, but any serious research will require considering a collection of portfolios over time. The help pages of the **portfolio** package provide detailed instructions on how to construct a portfolioHistory object. Here, we will just load up the example object provided with the package and consider the associated methods for working with it.

```
> data(global.2004.history)
> global.2004.history

An object of class "portfolioHistory"
Contains:

Object of class "exposureHistory"
Number of periods: 12
Period range: 2003-12-31 -- 2004-11-30
Data class: exposure
Variables: numeric currency sector

Object of class "performanceHistory"
Number of periods: 12
Period range: 2003-12-31 -- 2004-11-30
Data class: performance

Object of class "contributionHistory"
Number of periods: 12
Period range: 2003-12-31 -- 2004-11-30
Data class: contribution
Variables: currency sector liq.w
```

Note that the portfolios analysed in this object were created on a monthly frequency for 2004 using a universe of the 500 largest global stocks each month. Market capitalization in billions of dollars (cap.bil) was used as a measure of desirability, so the portfolio is long the mega-large caps and short the merely typical large caps.

We have designed the architecture of the **portfolio** package so that working with multi-period portfolios is similar to working with single period portfolios. For starters, we can call the exposure method on objects of class portfolioHistory just as we called them on objects of class portfolio. An added call to summary prints a summary of the returned object.

```
> summary(exposure(global.2004.history))

Mean exposure:
numeric
```

```
   variable   long   short exposure
11  cap.bil 109.20 -10.483    98.71
1     liq.w   1.22   0.822     2.04

currency
   variable      long     short  exposure
6       JPY 0.062978 -0.158931 -0.095952
8       SEK 0.000948 -0.040208 -0.039260
5       HKD 0.000000 -0.017283 -0.017283
1       AUD 0.001242 -0.005734 -0.004491
7       NOK 0.000000 -0.003949 -0.003949
9       SGD 0.000000 -0.000438 -0.000438
3       EUR 0.179902 -0.176904  0.002998
4       GBP 0.113589 -0.109119  0.004470
2       CHF 0.042340 -0.019232  0.023108
10      USD 0.598999 -0.468202  0.130797

sector
         variable    long    short  exposure
10      Utilities 0.00803 -0.08832 -0.080290
3       Cyclicals 0.04980 -0.12475 -0.074945
7       Materials 0.00112 -0.05876 -0.057636
6     Industrials 0.04147 -0.09762 -0.056152
2   Conglomerates 0.00000 -0.00356 -0.003563
9      Technology 0.08784 -0.08730  0.000545
8         Staples 0.24521 -0.23168  0.013532
5      Financials 0.24544 -0.19245  0.052982
4          Energy 0.10328 -0.03456  0.068719
1  Communications 0.19144 -0.05385  0.137588
```

The difficulty in considering exposures — basically a static concept about how the portfolio looks at this moment — in the context of a time series of `portfolio` objects is that it is not obvious how to summarize the exposures over time, especially in the context of a portfolio whose overall size is changing. In this release of the **portfolio** package, we will simply report the average exposure for all time periods.

For this example portfolio, it is easy to see that the portfolio is long the biggest stocks. The average market capitalization is over $100 billion on the long side versus only $10 billion on the short side. This cap exposure causes the portfolio to take a similar "bet" on liquidity, since larger stocks are, on average, more liquid than smaller stocks. The `liq.w` variable is our measure of liquidity — basically how much stock is traded on a "typical" day — scaled to have mean 0 and variance 1 for the universe.

The `currency` and `sector` exposures are, likewise, simple averages of the of the 12 monthly exposures. Even though the portfolio is constructed without reference to `currency` or `sector`, it still ends up with non-trivial exposures to these variables. In a typical month, the portfolio is 14% long Communications and 10% short Japan.

Plotting the `exposureHistory` object yields box and whisker plots of net exposure for each exposure variable, as in Figure 4.

```
> plot(exposure(global.2004.history))
```

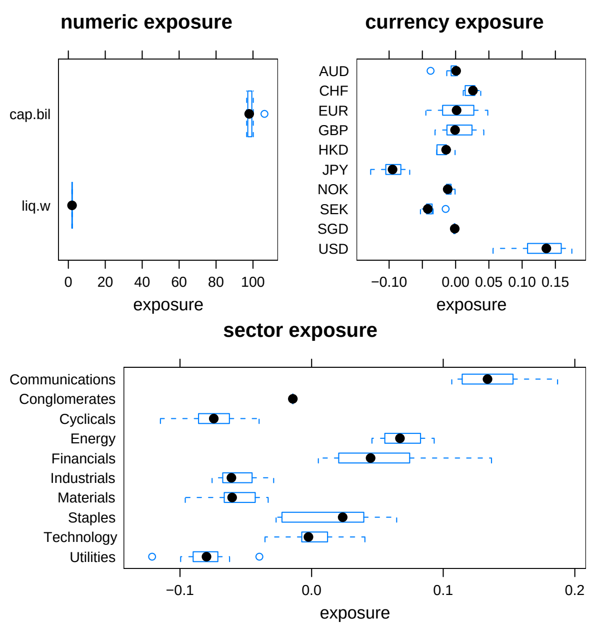

Figure 4: Plot of an exposureHistory object.

The `performance` method aggregates performance for each individual time period into an overall figure and provides a listing of the largest contributions.

```
> summary(performance(global.2004.history))

Performance summary (frequency = 12):

              ret
mean      -0.0081
mean (ann) -0.0971
sd         0.0152
sd (ann)   0.0527
sharpe    -1.8439

Best period:
                date    ret
2004-03-31 2004-03-31 0.0149


Worst period:
                date     ret
2004-10-31 2004-10-31 -0.0292

Top/Bottom performers (total contribution):
         id  contrib
140      EBAY  0.00423
231    MRW.LN  0.00387
234        MU  0.00357
338       XOM  0.00337
196       JNJ  0.00333
211       LTR -0.00416
323  VOLVB.SS -0.00453
329       WLP -0.00474
227       MON -0.00494
11       5201 -0.00555
```

The best month for the portfolio was April 2004 when it earned 1.5%. The worst month was a loss of almost 3% in November. The biggest contributor on the positive side was EBAY. The mean annualized return of the portfolio is almost -10%; the annualized volatility is 5.3%. Sharpe ratios — defined[1] as the mean return divided by the standard deviation of return — are generally not provided for unprofitable portfolios, but the ratio here is -1.8.

Note that the `portfolioHistory` object does not keep a record of every position for each time period. By design, it only retains summary information, albeit on a stock-by-stock basis. We want to be able to use this package for high frequency data — say, a decade worth of daily portfolios — so it is not easy (or even possible) to manipulate an object which maintains information for every stock for each time period.

As with the `exposure` method, the `contribution` method works with `portfolioHistory` objects in the same way that it works with `portfolio` objects.

```
> summary(contribution(global.2004.history))

Mean contribution:
currency
   variable    weight     contrib     roic
1       AUD 0.003488 -0.00014179 -0.02622
2       CHF 0.030786  0.00000868 -0.00291
11      DKK 0.000000  0.00000000  0.00000
3       EUR 0.178403 -0.00104765 -0.00620
4       GBP 0.111354 -0.00048120 -0.00409
5       HKD 0.008641  0.00011748 -0.00720
6       JPY 0.110954  0.00005111 -0.00273
21      NOK 0.001974 -0.00012765 -0.03745
7       SEK 0.020578 -0.00111390 -0.04674
31      SGD 0.000219 -0.00000963 -0.01264
8       USD 0.533601 -0.00534746 -0.00996

sector
         variable  weight    contrib     roic
1  Communications 0.12610  0.0008478  0.00814
11  Conglomerates 0.00179 -0.0000746 -0.01039
2       Cyclicals 0.08961 -0.0025430 -0.02863
3          Energy 0.07077  0.0008145  0.01384
4      Financials 0.22484 -0.0010191 -0.00416
5     Industrials 0.07153 -0.0014918 -0.01990
6       Materials 0.03067 -0.0002470 -0.01246
7         Staples 0.24511 -0.0036175 -0.01483
8      Technology 0.09007  0.0002594  0.00231
9       Utilities 0.04950 -0.0011930 -0.02607

liq.w
  variable weight  contrib    roic
1  1 - low 0.2628 -0.00390 -0.0140
2        2 0.1607 -0.00545 -0.0349
3        3 0.0668 -0.00136 -0.0197
4        4 0.0952 -0.00167 -0.0156
5 5 - high 0.4144  0.00384  0.0090
```

Again, the basic approach consists of calculating the contributions for each time period separately, saving those results, and then presenting an average for the entire period.

Figure 5 shows the plot of the resulting `contributionHistory` object. As with exposure history, the box and whisker plot shows the median and quartile range of variable values over the given time period.

```
> plot(contribution(global.2004.history))
```

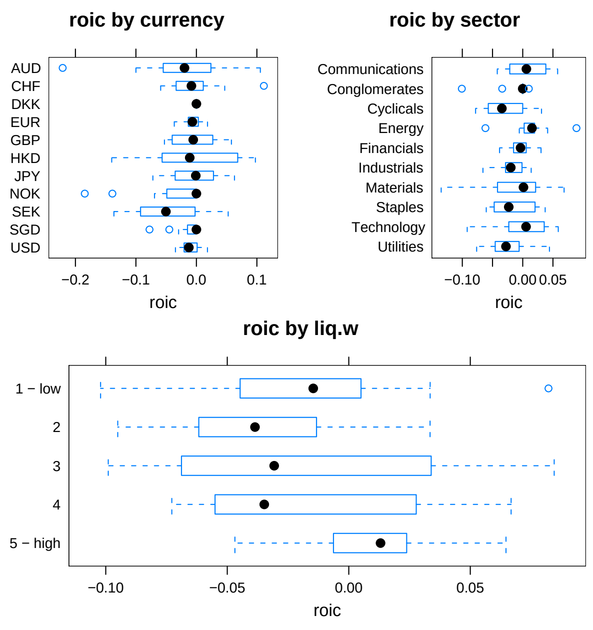

Figure 5: Plot of a contributionHistory object.

Conclusion

The current release of the portfolio package is meant to serve as a proof-of-concept. Relatively sophisticated portfolio analytics are possible using an open source package. Although money management is largely a zero-sum game — every dollar that I make is a dollar that someone else has lost — there is no reason why we might not cooperate on the tools that we all use.

David Kane and Jeff Enos
Kane Capital Management, Cambridge, Massachusetts, USA
`david@kanecap.com` *and* `jeff@kanecap.com`

[1]The definition of the Sharpe ratio subtracts the risk free rate from the mean return in the numerator. But, in the context of a long-short portfolio, this is typically ignored.

GroupSeq: Designing clinical trials using group sequential designs

by Roman Pahl, Andreas Ziegler, and Inke R. König

Introduction

Clinical trials are the gold standard to prove superiority of new interventions. An attractive variant to these studies is to use group sequential designs (Ziegler et al., 2003). Widely used since their introduction in the 1970s and 1980s, many different versions of group sequential designs have been developed. Common to all these designs is their complex computation for which software tools are required. In this context, existing software tools like ADDPLAN (`http://www.addplan.com/`) often are not free of charge, making it valuable to consider freely distributed software alternatives.

GroupSeq provides a set of several methods for computing group sequential boundaries of well established group sequential designs, among others the designs by Pocock (1977) and O'Brien and Fleming (1979). More precisely, the original designs are approximated by the *α-spending* function approach which was introduced by Lan and DeMets (1983) and Kim and DeMets (1987), for which a Fortran program by Reboussin et al. (2003) has been available. For **GroupSeq**, the latter implementation was completely recoded in R, and, in addition, new features were added. For instance, **GroupSeq** allows one to calculate exact Pocock bounds beside the estimated ones obtained by the *α-spending* approach.

Furthermore, as an important feature, this application is embedded in a graphical user interface (GUI) using the Tcl/Tk interface in the R **tcltk** package. Note that there is also a graphical user interface by Reboussin et al. (2003) although it only runs under Windows. In contrast, the **GroupSeq** GUI is available on several platforms, and, moreover, provides customization possibilities; i.e., the user may create as many windows as desired. Thus, he or she may perform multiple tasks such as computing and comparing several designs at the same time.

The computations yield valid results for any test which is based on normally distributed test statistics with independent increments, survival studies, and certain longitudinal designs. Using the α-spending function approach, interim analyses need not be equally spaced, and their number need not be specified in advance.

Group sequential tests

This paper only gives a basic introduction to group sequential designs. For a comprehensive overview of this topic, see, e.g., Jennison and Turnbull (2000) and Wassmer (2001).

The classical theory of testing goes back to Neyman and Pearson (1928) and is based on a sample of fixed size. In this sample, the null hypothesis H_0 is tested against an alternative hypothesis H_1. A significance level α has to be defined *a priori*, which implies the probability of the null hypothesis being falsely rejected. Importantly, the evaluation always takes place at the end after *all* observations have been made.

By contrast, group sequential designs allow consecutive testing, with possible rejection of the null hypothesis after observation of each group in a sequence of groups. Therefore, the significance levels corresponding to each group have to be adjusted appropriately. Many different group sequential designs have been developed. The main difference among these is the manner of allocating the specific significance levels. A further distinction consists in equally sized groups as in classical group sequential testing (Pocock, 1977; O'Brien and Fleming, 1979) versus unequally sized groups (Kim and DeMets, 1987).

Determining critical regions

Normally one wants to compare two population mean values μ_1 and μ_2—i.e., one wants to tests the null hypothesis $H_0 : \mu_1 = \mu_2$ against the alternative hypothesis $H_1 : \mu_1 \neq \mu_2$. Consider a standard normally distributed Z statistic

$$Z = \frac{\overline{X}_1 - \overline{X}_2}{\sqrt{\sigma_1^2/n_1 + \sigma_2^2/n_2}} \qquad (1)$$

with $\overline{X}_1$, $\overline{X}_2$ being the means of two independent samples of sizes n_1, n_2 which are distributed as $N(\mu_1, \sigma_1^2)$ and $N(\mu_2, \sigma_2^2)$. Assuming $n_1 = n_2 = n$ and $\sigma_1^2 = \sigma_2^2 = \sigma^2$, analogously to (1), one can define a statistic *per group* k :

$$Z_k = \frac{\overline{X}_{1k} - \overline{X}_{2k}}{\sigma} \sqrt{\frac{n_k}{2}}$$

The standardized overall statistic until group k then is defined as

$$Z_k^* = \frac{\sum_{j=1}^{k} \sqrt{n_j} Z_j}{\sqrt{\sum_{j=1}^{k} n_j}}$$